

Package ‘fitdistcp’

July 22, 2025

Type Package

Title Distribution Fitting with Calibrating Priors for Commonly Used Distributions

Version 0.1.1

Maintainer Stephen Jewson <stephen.jewson@gmail.com>

Imports stats, mev, extraDistr, gnorm, fdrtool, pracma, rust, actuar, fExtremes

Depends R (>= 3.5.0)

Description Generates predictive distributions based on calibrating priors for various commonly used statistical models, including models with predictors. Routines for densities, probabilities, quantiles, random deviates and the parameter posterior are provided. The predictions are generated from the Bayesian prediction integral, with priors chosen to give good reliability (also known as calibration). For homogeneous models, the prior is set to the right Haar prior, giving predictions which are exactly reliable. As a result, in repeated testing, the frequencies of out-of-sample outcomes and the probabilities from the predictions agree. For other models, the prior is chosen to give good reliability. Where possible, the Bayesian prediction integral is solved exactly. Where exact solutions are not possible, the Bayesian prediction integral is solved using the Datta-Mukerjee-Ghosh-Sweeting (DMGS) asymptotic expansion. Optionally, the prediction integral can also be solved using posterior samples generated using Paul Northrop's ratio of uniforms sampling package ('rust'). Results are also generated based on maximum likelihood, for comparison purposes. Various model selection diagnostics and testing routines are included. Based on ``Reducing reliability bias in assessments of extreme weather risk using calibrating priors'', Jewson, S., Sweeting, T. and Jewson, L. (2024); <doi:10.5194/ascmo-11-1-2025>.

License MIT + file LICENSE

BugReports <https://github.com/stephenjewson/fitdistcp/issues>

URL <https://fitdistcp.info>

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Suggests knitr, rmarkdown

NeedsCompilation no

Author Stephen Jewson [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-6011-6262>>)

Repository CRAN

Date/Publication 2025-04-23 09:40:02 UTC

Contents

adhoc_dmgs_cpmethod	26
analytic_cpmethod	26
bayesian_dq_4terms_v1	27
calc_revert2ml	27
cauchy_cp	28
cauchy_f1f	35
cauchy_f1fa	35
cauchy_f2f	36
cauchy_f2fa	36
cauchy_fd	37
cauchy_fdd	37
cauchy_ldd	38
cauchy_ldda	38
cauchy_lddd	39
cauchy_lddda	39
cauchy_lmn	40
cauchy_lmnp	40
cauchy_logf	41
cauchy_logfdd	42
cauchy_logfddd	42
cauchy_loglik	43
cauchy_logscores	43
cauchy_mu1f	44
cauchy_mu2f	44
cauchy_p1f	45
cauchy_p1_cp	45
cauchy_p1_f1f	53
cauchy_p1_f1fa	54
cauchy_p1_f2f	54
cauchy_p1_f2fa	55
cauchy_p1_fd	55
cauchy_p1_fdd	56
cauchy_p1_ldd	56
cauchy_p1_ldda	57
cauchy_p1_lddd	58

cauchy_p1_lddda	58
cauchy_p1_lmn	59
cauchy_p1_lmnp	60
cauchy_p1_logf	60
cauchy_p1_logfdd	61
cauchy_p1_logfddd	62
cauchy_p1_loglik	62
cauchy_p1_logscores	63
cauchy_p1_means	63
cauchy_p1_mu1f	64
cauchy_p1_mu2f	65
cauchy_p1_p1f	65
cauchy_p1_p2f	66
cauchy_p1_predictordata	67
cauchy_p1_waic	67
cauchy_p2f	68
cauchy_waic	69
crhpflat_dmgs_cpmethod	70
d100gamma_example_data_v1	70
d101invgamma_example_data_v1	70
d102invgauss_example_data_v1	70
d105burr_example_data_v1	71
d10exp_example_data_v1	71
d110gev_example_data_v1	71
d11pareto_k2_example_data_v1	71
d120gpd_k1_example_data_v1	71
d150gev_p1_example_data_v1_t	72
d150gev_p1_example_data_v1_x	72
d151gev_p12_example_data_v1_t	72
d151gev_p12_example_data_v1_x	72
d152gev_p123_example_data_v1_t	72
d152gev_p123_example_data_v1_x	73
d20halfnorm_example_data_v1	73
d25unif_example_data_v1	73
d30norm_example_data_v1	73
d31norm_dmgs_example_data_v1	73
d32gnorm_k3_example_data_v1	74
d35lnorm_example_data_v1	74
d36lnorm_dmgs_example_data_v1	74
d40logis_example_data_v1	74
d41lst_k3_example_data_v1	74
d42cauchy_example_data_v1	75
d50gumbel_example_data_v1	75
d51frechet_k1_example_data_v1	75
d52weibull_example_data_v1	75
d53gev_k3_example_data_v1	75
d55exp_p1_example_data_v1_t	76
d55exp_p1_example_data_v1_x	76

d56pareto_p1k2_example_data_v1_t	76
d56pareto_p1k2_example_data_v1_x	76
d60norm_p1_example_data_v1_t	76
d60norm_p1_example_data_v1_x	77
d61lnorm_p1_example_data_v1_t	77
d61lnorm_p1_example_data_v1_x	77
d62logis_p1_example_data_v1_t	77
d62logis_p1_example_data_v1_x	77
d63lst_p1k3_example_data_v1_t	78
d63lst_p1k3_example_data_v1_x	78
d64cauchy_p1_example_data_v1_t	78
d64cauchy_p1_example_data_v1_x	78
d70gumbel_p1_example_data_v1_t	78
d70gumbel_p1_example_data_v1_x	79
d71frechet_p2k1_example_data_v1_t	79
d71frechet_p2k1_example_data_v1_x	79
d72weibull_p1_example_data_v1_t	79
d72weibull_p1_example_data_v1_x	79
d73weibull_p2_example_data_v1_t	80
d73weibull_p2_example_data_v1_x	80
d74gev_p1k3_example_data_v1_t	80
d74gev_p1k3_example_data_v1_x	80
d80norm_p12_example_data_v1_t1	80
d80norm_p12_example_data_v1_t2	81
d80norm_p12_example_data_v1_x	81
d81lst_p12k3_example_data_v1_t1	81
d81lst_p12k3_example_data_v1_t2	81
d81lst_p12k3_example_data_v1_x	81
d82weibull_p12_example_data_v1_t1	82
d82weibull_p12_example_data_v1_t2	82
d82weibull_p12_example_data_v1_x	82
dcauchysub	82
dcauchy_p1	83
dcauchy_p1sub	83
deriv_copyfdd	84
deriv_copyld2	85
deriv_copyldd	85
deriv_copylddd	86
dexpsub	86
dexp_p1	87
dexp_p1sub	87
dfrechetsub	88
dfrechet_p2k1	88
dfrechet_p2k1sub	89
dgammausub	90
dgevsusub	90
dgev_k3sub	91
dgev_p1	92

dgev_p12	92
dgev_p123	93
dgev_p123sub	94
dgev_p12sub	95
dgev_p1k3	96
dgev_p1k3sub	97
dgev_p1sub	97
dgnorm_k3sub	99
dgpds	99
dgumbelsub	100
dgumbel_p1	101
dgumbel_p1sub	101
dhalfnormsub	102
dinvgamma	102
dinvgauss	103
dlnormsub	103
dlnorm_dmgssub	104
dlnorm_p1	104
dlnorm_p1sub	105
dlogis2sub	105
dlogis_p1	106
dlogis_p1sub	106
dlst_k3sub	107
dlst_p1k3	108
dlst_p1k3sub	108
dmgs	109
dnormsub	110
dnorm_dmgssub	110
dnorm_p1	111
dnorm_p1sub	111
dnorm_p1_formula	112
dpareto_k2_sub	112
dpareto_p1k2	113
dpareto_p1k2sub	113
dunif_formula	114
dweibullsub	114
dweibull_p2	115
dweibull_p2sub	115
exp_cp	116
exp_f1f	122
exp_f1fa	122
exp_f2f	123
exp_f2fa	123
exp_fd	124
exp_fdd	124
exp_l111	125
exp_ldd	125
exp_ldda	126

exp_lddd	126
exp_lddda	127
exp_logf	127
exp_logfdd	128
exp_logfddd	128
exp_logscores	129
exp_p1fa	129
exp_p1_cp	130
exp_p1_f1f	137
exp_p1_f1fa	138
exp_p1_f2f	138
exp_p1_f2fa	139
exp_p1_fd	139
exp_p1_fdd	140
exp_p1_ddd	140
exp_p1_ldda	141
exp_p1_lddd	141
exp_p1_lddda	142
exp_p1_lmn	142
exp_p1_lmnp	143
exp_p1_logf	143
exp_p1_logfdd	144
exp_p1_logfddd	145
exp_p1_loglik	145
exp_p1_logscores	146
exp_p1_means	146
exp_p1_mu1f	147
exp_p1_mu1fa	148
exp_p1_mu2f	148
exp_p1_mu2fa	149
exp_p1_p1f	149
exp_p1_p1fa	150
exp_p1_p2f	150
exp_p1_p2fa	151
exp_p1_pd	151
exp_p1_pdd	152
exp_p1_predictordata	152
exp_p1_waic	153
exp_p2fa	154
exp_pd	154
exp_pdd	155
exp_waic	155
fixgevrage	156
fixgpdrange	156
frechet_k1_cp	157
frechet_k1_f1f	164
frechet_k1_f1fa	164
frechet_k1_f2f	165

frechet_k1_f2fa	166
frechet_k1_fd	166
frechet_k1_fdd	167
frechet_k1_ldd	167
frechet_k1_ldda	168
frechet_k1_lddd	168
frechet_k1_lddda	169
frechet_k1_lmn	169
frechet_k1_lmnp	170
frechet_k1_logf	171
frechet_k1_logfdd	171
frechet_k1_logfddd	172
frechet_k1_mu1f	172
frechet_k1_mu1fa	173
frechet_k1_mu2f	173
frechet_k1_mu2fa	174
frechet_k1_p1f	174
frechet_k1_p1fa	175
frechet_k1_p2f	175
frechet_k1_p2fa	176
frechet_k1_pd	176
frechet_k1_pdd	177
frechet_k1_waic	177
frechet_loglik	178
frechet_logscores	179
frechet_means	179
frechet_p2k1_cp	180
frechet_p2k1_f1f	188
frechet_p2k1_f1fa	189
frechet_p2k1_f2f	189
frechet_p2k1_f2fa	190
frechet_p2k1_fd	191
frechet_p2k1_fdd	191
frechet_p2k1_ldd	192
frechet_p2k1_ldda	193
frechet_p2k1_lddd	193
frechet_p2k1_lddda	194
frechet_p2k1_lmn	195
frechet_p2k1_lmnp	195
frechet_p2k1_logf	196
frechet_p2k1_logfdd	197
frechet_p2k1_logfddd	197
frechet_p2k1_loglik	198
frechet_p2k1_logscores	199
frechet_p2k1_means	199
frechet_p2k1_mu1f	200
frechet_p2k1_mu1fa	201
frechet_p2k1_mu2f	202

frechet_p2k1_mu2fa	202
frechet_p2k1_p1f	203
frechet_p2k1_p1fa	204
frechet_p2k1_p2f	204
frechet_p2k1_p2fa	205
frechet_p2k1_pd	206
frechet_p2k1_pdd	206
frechet_p2k1_predictordata	207
frechet_p2k1_waic	208
gamma_cp	209
gamma_f1f	216
gamma_f1fa	216
gamma_f2f	217
gamma_f2fa	217
gamma_fd	218
gamma_fdd	218
gamma_gg	219
gamma_gmn	219
gamma_ddd	220
gamma_ddd	220
gamma_ddd	221
gamma_ddd	221
gamma_lmn	222
gamma_lmnp	222
gamma_logf	223
gamma_logfdd	224
gamma_logfddd	224
gamma_loglik	225
gamma_logscores	225
gamma_means	226
gamma_mu1f	226
gamma_mu2f	227
gamma_p1f	227
gamma_p2f	228
gamma_waic	229
gev_checkmle	229
gev_cp	230
gev_f1f	239
gev_f1fa	240
gev_f2f	240
gev_f2fa	241
gev_fd	241
gev_fdd	242
gev_ggd_mev	242
gev_ggid_mev	243
gev_k12_ppm_minusloglik	244
gev_k3_cp	245
gev_k3_f1f	253

gev_k3_f1fa	253
gev_k3_f2f	254
gev_k3_f2fa	254
gev_k3_fd	255
gev_k3_fdd	255
gev_k3_ddd	256
gev_k3_ldda	256
gev_k3_ddd	257
gev_k3_lddda	257
gev_k3_lmn	258
gev_k3_lmnp	258
gev_k3_logf	259
gev_k3_logfdd	260
gev_k3_logfddd	260
gev_k3_loglik	261
gev_k3_means	261
gev_k3_mu1f	262
gev_k3_mu1fa	262
gev_k3_mu2f	263
gev_k3_mu2fa	263
gev_k3_pd	264
gev_k3_pdd	264
gev_k3_waic	265
gev_ld12a	266
gev_lda	266
gev_ddd	267
gev_ldda	267
gev_ddd	268
gev_lddda	268
gev_lmn	269
gev_lmnp	270
gev_logf	270
gev_logfd	271
gev_logfdd	271
gev_logfddd	272
gev_loglik	273
gev_means	273
gev_mu1f	274
gev_mu1fa	275
gev_mu2f	275
gev_mu2fa	276
gev_p123_checkmle	276
gev_p123_cp	277
gev_p123_f1f	287
gev_p123_f1fa	288
gev_p123_f2f	288
gev_p123_f2fa	289
gev_p123_fd	290

gev_p123_fdd	291
gev_p123_ldd	291
gev_p123_ldda	292
gev_p123_lddd	293
gev_p123_lddda	294
gev_p123_lmn	294
gev_p123_lmnp	296
gev_p123_logf	297
gev_p123_logfdd	298
gev_p123_logfddd	298
gev_p123_loglik	299
gev_p123_means	300
gev_p123_mu1f	300
gev_p123_mu1fa	301
gev_p123_mu2f	302
gev_p123_mu2fa	303
gev_p123_pd	304
gev_p123_pdd	305
gev_p123_predictordata	305
gev_p123_setics	306
gev_p123_waic	307
gev_p12k3_f1f	308
gev_p12k3_f1fa	309
gev_p12k3_f2f	309
gev_p12k3_f2fa	310
gev_p12k3_fd	311
gev_p12k3_fdd	311
gev_p12k3_ldd	312
gev_p12k3_ldda	313
gev_p12k3_lddd	313
gev_p12k3_lddda	314
gev_p12k3_logfdd	315
gev_p12k3_logfddd	315
gev_p12k3_mu1f	316
gev_p12k3_mu1fa	317
gev_p12k3_mu2f	317
gev_p12k3_mu2fa	318
gev_p12k3_pd	319
gev_p12k3_pdd	319
gev_p12_checkmle	320
gev_p12_cp	321
gev_p12_f1f	330
gev_p12_f1fa	331
gev_p12_f2f	331
gev_p12_f2fa	332
gev_p12_fd	333
gev_p12_fdd	333
gev_p12_ggd	334

gev_p12_ldd	335
gev_p12_ldda	335
gev_p12_lddd	336
gev_p12_lddda	337
gev_p12_lmn	337
gev_p12_lmnp	338
gev_p12_logf	339
gev_p12_logfdd	340
gev_p12_logfddd	340
gev_p12_loglik	341
gev_p12_means	342
gev_p12_mu1f	342
gev_p12_mu1fa	343
gev_p12_mu2f	344
gev_p12_mu2fa	344
gev_p12_pd	345
gev_p12_pdd	346
gev_p12_predictordata	346
gev_p12_setics	347
gev_p12_waic	348
gev_p1k3_cp	349
gev_p1k3_f1f	357
gev_p1k3_f1fa	358
gev_p1k3_f2f	359
gev_p1k3_f2fa	359
gev_p1k3_fd	360
gev_p1k3_fdd	361
gev_p1k3_ldd	361
gev_p1k3_ldda	362
gev_p1k3_lddd	363
gev_p1k3_lddda	363
gev_p1k3_lmn	364
gev_p1k3_lmnp	365
gev_p1k3_logf	365
gev_p1k3_logfdd	366
gev_p1k3_logfddd	367
gev_p1k3_loglik	367
gev_p1k3_means	368
gev_p1k3_mu1f	368
gev_p1k3_mu1fa	369
gev_p1k3_mu2f	370
gev_p1k3_mu2fa	370
gev_p1k3_pd	371
gev_p1k3_pdd	372
gev_p1k3_predictordata	372
gev_p1k3_waic	373
gev_p1_checkmle	374
gev_p1_cp	374

gev_p1_f1f	384
gev_p1_f1fa	385
gev_p1_f2f	385
gev_p1_f2fa	386
gev_p1_fd	387
gev_p1_fdd	387
gev_p1_ggd	388
gev_p1_ddd	389
gev_p1_ldda	389
gev_p1_ddd	390
gev_p1_lddda	391
gev_p1_lmn	391
gev_p1_lmnp	392
gev_p1_logf	393
gev_p1_logfdd	393
gev_p1_logfddd	394
gev_p1_loglik	395
gev_p1_means	395
gev_p1_mu1f	396
gev_p1_mu1fa	397
gev_p1_mu2f	397
gev_p1_mu2fa	398
gev_p1_pd	399
gev_p1_pdd	399
gev_p1_predictordata	400
gev_p1_setics	401
gev_p1_waic	401
gev_pd	402
gev_pdd	403
gev_pwm_params	403
gev_setics	404
gev_waic	404
gnorm_k3_cp	405
gnorm_k3_f1f	412
gnorm_k3_f1fa	413
gnorm_k3_f2f	414
gnorm_k3_f2fa	414
gnorm_k3_fd	415
gnorm_k3_fdd	415
gnorm_k3_ddd	416
gnorm_k3_ldda	417
gnorm_k3_ddd	417
gnorm_k3_lddda	418
gnorm_k3_lmn	418
gnorm_k3_logf	419
gnorm_k3_logfdd	419
gnorm_k3_logfddd	420
gnorm_k3_loglik	420

gnorm_k3_logscores	421
gnorm_k3_mu1f	421
gnorm_k3_mu2f	422
gnorm_k3_p1f	423
gnorm_k3_p2f	423
gnorm_lmnp	424
gnorm_waic	425
gpd_k13_f1f	426
gpd_k13_f1fa	426
gpd_k13_f2f	427
gpd_k13_f2fa	427
gpd_k13_fd	428
gpd_k13_fdd	428
gpd_k13_l11	429
gpd_k13_l111	429
gpd_k13_ldd	430
gpd_k13_ldda	430
gpd_k13_lddd	431
gpd_k13_lddda	431
gpd_k13_logfdd	432
gpd_k13_logfddd	432
gpd_k13_mu1f	433
gpd_k13_mu1fa	433
gpd_k13_mu2f	434
gpd_k13_mu2fa	434
gpd_k13_p1f	435
gpd_k13_p2f	435
gpd_k13_pd	436
gpd_k13_pdd	436
gpd_k1_checkmle	437
gpd_k1_cp	437
gpd_k1_f1f	446
gpd_k1_f1fa	447
gpd_k1_f2f	447
gpd_k1_f2fa	448
gpd_k1_fd	448
gpd_k1_fdd	449
gpd_k1_ggd_mev	449
gpd_k1_ldd	450
gpd_k1_ldda	450
gpd_k1_lddd	451
gpd_k1_lddda	451
gpd_k1_lmn	452
gpd_k1_lmnp	452
gpd_k1_logf	453
gpd_k1_logfdd	454
gpd_k1_logfddd	454
gpd_k1_loglik	455

gpd_k1_means	455
gpd_k1_mu1f	456
gpd_k1_mu1fa	457
gpd_k1_mu2f	457
gpd_k1_mu2fa	458
gpd_k1_p1f	458
gpd_k1_p2f	459
gpd_k1_pd	459
gpd_k1_pdd	460
gpd_k1_setics	460
gpd_k1_waic	461
gumbel_cp	462
gumbel_f1f	469
gumbel_f1fa	469
gumbel_f2f	470
gumbel_f2fa	470
gumbel_fd	471
gumbel_fdd	471
gumbel_ddd	472
gumbel_ldda	472
gumbel_ddd	473
gumbel_ddd	473
gumbel_lmn	474
gumbel_lmnp	474
gumbel_logf	475
gumbel_logfdd	476
gumbel_logfddd	476
gumbel_loglik	477
gumbel_logscores	477
gumbel_means	478
gumbel_mu1f	478
gumbel_mu1fa	479
gumbel_mu2f	479
gumbel_mu2fa	480
gumbel_p1f	480
gumbel_p1fa	481
gumbel_p1_cp	481
gumbel_p1_f1f	489
gumbel_p1_f1fa	490
gumbel_p1_f2f	490
gumbel_p1_f2fa	491
gumbel_p1_fd	491
gumbel_p1_fdd	492
gumbel_p1_ddd	492
gumbel_p1_ldda	493
gumbel_p1_ddd	494
gumbel_p1_ddd	494
gumbel_p1_lmn	495

gumbel_p1_lmnf	496
gumbel_p1_logf	496
gumbel_p1_logfdd	497
gumbel_p1_logfddd	498
gumbel_p1_loglik	498
gumbel_p1_logscores	499
gumbel_p1_means	499
gumbel_p1_mu1f	500
gumbel_p1_mu1fa	501
gumbel_p1_mu2f	501
gumbel_p1_mu2fa	502
gumbel_p1_p1f	502
gumbel_p1_p1fa	503
gumbel_p1_p2f	504
gumbel_p1_p2fa	504
gumbel_p1_pd	505
gumbel_p1_pdd	505
gumbel_p1_predictordata	506
gumbel_p1_waic	507
gumbel_p2f	508
gumbel_p2fa	508
gumbel_pd	509
gumbel_pdd	509
gumbel_waic	510
halfnorm_cp	510
halfnorm_f1f	517
halfnorm_f1fa	518
halfnorm_f2f	518
halfnorm_f2fa	519
halfnorm_fd	519
halfnorm_fdd	520
halfnorm_gg	520
halfnorm_gg11	521
halfnorm_l111	521
halfnorm_ldd	522
halfnorm_ldda	522
halfnorm_lddd	523
halfnorm_lddd	523
halfnorm_logf	524
halfnorm_logfdd	524
halfnorm_logfddd	525
halfnorm_loglik	525
halfnorm_logscores	526
halfnorm_means	526
halfnorm_mu1f	527
halfnorm_mu2f	527
halfnorm_p1f	528
halfnorm_p2f	528

halfnorm_waic	529
invgamma_cp	529
invgamma_f1f	536
invgamma_f1fa	537
invgamma_f2f	538
invgamma_f2fa	538
invgamma_fd	539
invgamma_fdd	539
invgamma_ddd	540
invgamma_ldda	540
invgamma_ddd	541
invgamma_ddd	541
invgamma_lmn	542
invgamma_lmnp	542
invgamma_logf	543
invgamma_logfdd	544
invgamma_logfddd	544
invgamma_loglik	545
invgamma_logscores	545
invgamma_mu1f	546
invgamma_mu2f	546
invgamma_p1f	547
invgamma_p2f	547
invgamma_waic	548
invgauss_cp	549
invgauss_f1f	556
invgauss_f1fa	557
invgauss_f2f	557
invgauss_f2fa	558
invgauss_fd	558
invgauss_fdd	559
invgauss_ddd	559
invgauss_ldda	560
invgauss_ddd	560
invgauss_ddd	561
invgauss_lmn	561
invgauss_lmnp	562
invgauss_logf	562
invgauss_logfdd	563
invgauss_logfddd	563
invgauss_loglik	564
invgauss_logscores	564
invgauss_means	565
invgauss_mu1f	566
invgauss_mu2f	566
invgauss_p1f	567
invgauss_p2f	567
invgauss_waic	568

jpf2p	569
jpf3p	569
jpf4p	570
lnorm_cp	570
lnorm_dmgs_cp	577
lnorm_dmgs_gg11	583
lnorm_dmgs_gg12	584
lnorm_dmgs_gg22	584
lnorm_dmgs_loglik	585
lnorm_dmgs_logscores	585
lnorm_dmgs_means	586
lnorm_dmgs_mu1f	586
lnorm_dmgs_mu2f	587
lnorm_dmgs_p1f	587
lnorm_dmgs_p2f	588
lnorm_dmgs_waic	588
lnorm_f1f	589
lnorm_f1fa	590
lnorm_f2f	590
lnorm_f2fa	591
lnorm_fd	591
lnorm_fdd	592
lnorm_ldd	592
lnorm_ldda	593
lnorm_lddd	593
lnorm_lddda	594
lnorm_lmn	594
lnorm_lmnp	595
lnorm_logf	595
lnorm_logfdd	596
lnorm_logfddd	596
lnorm_logscores	597
lnorm_mu1fa	597
lnorm_mu2fa	598
lnorm_p1fa	598
lnorm_p1_cp	599
lnorm_p1_f1f	606
lnorm_p1_f1fa	607
lnorm_p1_f2f	607
lnorm_p1_f2fa	608
lnorm_p1_fd	608
lnorm_p1_fdd	609
lnorm_p1_ldd	609
lnorm_p1_ldda	610
lnorm_p1_lddd	611
lnorm_p1_lddda	611
lnorm_p1_lmn	612
lnorm_p1_lmnp	613

lnorm_p1_logf	613
lnorm_p1_logfdd	614
lnorm_p1_logfddd	615
lnorm_p1_loglik	615
lnorm_p1_logscores	616
lnorm_p1_mu1fa	616
lnorm_p1_mu2fa	617
lnorm_p1_p1fa	617
lnorm_p1_p2fa	618
lnorm_p1_pd	618
lnorm_p1_pdd	619
lnorm_p1_predictordata	619
lnorm_p1_waic	620
lnorm_p2fa	621
lnorm_pd	621
lnorm_pdd	622
lnorm_waic	622
logis_cp	623
logis_f1f	630
logis_f1fa	630
logis_f2f	631
logis_f2fa	631
logis_fd	632
logis_fdd	632
logis_ldd	633
logis_ldda	633
logis_lddd	634
logis_lddda	634
logis_lmn	635
logis_lmnp	635
logis_logf	636
logis_logfdd	637
logis_logfddd	637
logis_loglik	638
logis_logscores	638
logis_mu1f	639
logis_mu1fa	639
logis_mu2f	640
logis_mu2fa	640
logis_p1f	641
logis_p1fa	641
logis_p1_cp	642
logis_p1_f1f	649
logis_p1_f1fa	650
logis_p1_f2f	651
logis_p1_f2fa	651
logis_p1_fd	652
logis_p1_fdd	652

logis_p1_ldd	653
logis_p1_ldda	654
logis_p1_lddd	654
logis_p1_lddda	655
logis_p1_lmn	655
logis_p1_lmnp	656
logis_p1_logf	657
logis_p1_logfdd	657
logis_p1_logfddd	658
logis_p1_loglik	658
logis_p1_logscores	659
logis_p1_means	659
logis_p1_mu1f	660
logis_p1_mu1fa	661
logis_p1_mu2f	661
logis_p1_mu2fa	662
logis_p1_p1f	662
logis_p1_p1fa	663
logis_p1_p2f	664
logis_p1_p2fa	664
logis_p1_pd	665
logis_p1_pdd	665
logis_p1_predictordata	666
logis_p1_waic	667
logis_p2f	668
logis_p2fa	668
logis_pd	669
logis_pdd	669
logis_waic	670
lst_k3_cp	670
lst_k3_f1f	677
lst_k3_f1fa	678
lst_k3_f2f	679
lst_k3_f2fa	679
lst_k3_fd	680
lst_k3_fdd	680
lst_k3_ldd	681
lst_k3_ldda	681
lst_k3_lddd	682
lst_k3_lddda	682
lst_k3_lmn	683
lst_k3_lmnp	683
lst_k3_logf	684
lst_k3_logfdd	685
lst_k3_logfddd	685
lst_k3_loglik	686
lst_k3_logscores	686
lst_k3_mu1f	687

lst_k3_mu2f	687
lst_k3_p1f	688
lst_k3_p2f	688
lst_k3_waic	689
lst_p1k3_cp	690
lst_p1k3_f1f	698
lst_p1k3_f1fa	699
lst_p1k3_f2f	699
lst_p1k3_f2fa	700
lst_p1k3_fd	701
lst_p1k3_fdd	701
lst_p1k3_ddd	702
lst_p1k3_ddd	703
lst_p1k3_ddd	703
lst_p1k3_ddd	704
lst_p1k3_lmn	705
lst_p1k3_lmnp	705
lst_p1k3_logf	706
lst_p1k3_logfdd	707
lst_p1k3_logfddd	707
lst_p1k3_loglik	708
lst_p1k3_logscores	709
lst_p1k3_mu1f	709
lst_p1k3_mu2f	710
lst_p1k3_p1f	711
lst_p1k3_p2f	711
lst_p1k3_predictordata	712
lst_p1k3_setics	713
lst_p1k3_waic	713
makemuhat0	714
makeq	715
maket0	715
maketa0	716
make_cwaic	716
make_maic	717
make_se	717
make_waic	718
man	718
man1f	728
man2f	728
mandsub	729
manf	729
manldd	736
manlddd	736
manlnn	736
manlnnn	737
manlogf	737
manloglik	737

manlogscores	738
manmeans	738
manpredictor	738
manvector	739
manwaic	739
movexiawayfromzero	739
ms_flat_1tail	740
ms_flat_2tail	741
ms_predictors_1tail	742
ms_predictors_2tail	743
nopdfcdfmsg	744
norm_cp	745
norm_dmgs_cp	750
norm_dmgs_loglik	756
norm_dmgs_logscores	757
norm_dmgs_means	757
norm_dmgs_mu1f	758
norm_dmgs_mu2f	759
norm_dmgs_p1f	759
norm_dmgs_p2f	760
norm_dmgs_waic	760
norm_f1f	761
norm_f1fa	762
norm_f2f	762
norm_f2fa	763
norm_fd	763
norm_fdd	764
norm_gg	764
norm_gmn	765
norm_ldd	765
norm_ldda	766
norm_lddd	766
norm_lddda	767
norm_lmn	767
norm_lmnp	768
norm_logf	769
norm_logfdd	769
norm_logfddd	770
norm_logscores	770
norm_ml_params	771
norm_mu1fa	771
norm_mu2fa	772
norm_p1fa	772
norm_p1_cp	773
norm_p1_f1f	780
norm_p1_f1fa	781
norm_p1_f2f	781
norm_p1_f2fa	782

norm_p1_fd	783
norm_p1_fdd	783
norm_p1_ddd	784
norm_p1_ldda	785
norm_p1_ddd	785
norm_p1_lddda	786
norm_p1_lmn	787
norm_p1_lmnp	787
norm_p1_logf	788
norm_p1_logfdd	789
norm_p1_logfddd	789
norm_p1_loglik	790
norm_p1_logscores	791
norm_p1_mlparams	791
norm_p1_mu1fa	792
norm_p1_mu2fa	792
norm_p1_p1fa	793
norm_p1_p2fa	794
norm_p1_pd	794
norm_p1_pdd	795
norm_p1_predictordata	796
norm_p1_waic	796
norm_p2fa	797
norm_pd	798
norm_pdd	798
norm_unbiasedv_params	799
norm_waic	799
pareto_k2_cp	800
pareto_k2_f1f	806
pareto_k2_f1fa	807
pareto_k2_f2f	807
pareto_k2_f2fa	808
pareto_k2_fd	808
pareto_k2_fdd	809
pareto_k2_l111	809
pareto_k2_ddd	810
pareto_k2_ldda	811
pareto_k2_ddd	811
pareto_k2_lddda	812
pareto_k2_logf	812
pareto_k2_logfdd	813
pareto_k2_logfddd	813
pareto_k2_logscores	814
pareto_k2_ml_params	814
pareto_k2_mu1fa	815
pareto_k2_mu2fa	815
pareto_k2_p1fa	816
pareto_k2_p2fa	816

pareto_k2_pd	817
pareto_k2_pdd	817
pareto_k2_waic	818
pareto_p1k2_cp	818
pareto_p1k2_f1f	826
pareto_p1k2_f1fa	827
pareto_p1k2_f2f	827
pareto_p1k2_f2fa	828
pareto_p1k2_fd	828
pareto_p1k2_fdd	829
pareto_p1k2_ddd	829
pareto_p1k2_ldda	830
pareto_p1k2_ddd	831
pareto_p1k2_ddd	831
pareto_p1k2_lmn	832
pareto_p1k2_lmp	832
pareto_p1k2_logf	833
pareto_p1k2_logfdd	834
pareto_p1k2_logfddd	834
pareto_p1k2_loglik	835
pareto_p1k2_logscores	835
pareto_p1k2_means	836
pareto_p1k2_mu1f	837
pareto_p1k2_mu1fa	838
pareto_p1k2_mu2f	838
pareto_p1k2_mu2fa	839
pareto_p1k2_p1f	839
pareto_p1k2_p1fa	840
pareto_p1k2_p2f	840
pareto_p1k2_p2fa	841
pareto_p1k2_pd	841
pareto_p1k2_pdd	842
pareto_p1k2_predictordata	842
pareto_p1k2_waic	843
pcauchy_p1	844
pexp_p1	845
pfrechet_p2k1	845
pgev_p1	846
pgev_p12	846
pgev_p123	847
pgev_p1k3	848
pgumbel_p1	848
plnorm_p1	849
plogis_p1	849
plst_p1k3	850
pnorm_p1	850
pnorm_p1_formula	851
ppareto_p1k2	851

punif_formula	852
pweibull_p2	852
qcauchy_p1	853
qexp_p1	853
qfrechet_p2k1	854
qgamma_k1_ppm	854
qgamma_ppm	856
qgev_k12_ppm	857
qgev_mpd_ppm	858
qgev_p1	860
qgev_p12	860
qgev_p123	861
qgev_p1k3	862
qgev_p1_ppm	862
qgev_ppm	864
qgpd_k1_ppm	865
qgumbel_p1	866
qlnorm_p1	867
qlogis_p1	868
qlst_p1k3	868
qnorm_p1	869
qnorm_p1_formula	869
qntt_ppm	870
qpareto_p1k2	871
qunif_formula	872
qweibull_p2	872
reltest	873
reltest2	876
reltest2_cases	879
reltest2_makekeep	879
reltest2_plot	880
reltest2_predict	881
reltest2_simulate	881
reltest_makekeep	882
reltest_makemaxep	883
reltest_predict	883
reltest_simulate	885
rgev_minmax	886
rgev_p123_minmax	886
rgev_p12_minmax	887
rgev_p1_minmax	888
rgpd_k1_minmax	889
rhp_dmgs_cpmethod	889
rust_pumethod	890
testppm_plot	890
unif_cp	891
weibull_cp	897
weibull_f1f	904

weibull_f1fa	904
weibull_f2f	905
weibull_f2fa	905
weibull_fd	906
weibull_fdd	906
weibull_ddd	907
weibull_ldda	907
weibull_ddd	908
weibull_ddd	908
weibull_lmn	909
weibull_lmnp	909
weibull_logf	910
weibull_logfdd	911
weibull_logfddd	911
weibull_loglik	912
weibull_logscores	912
weibull_means	913
weibull_mu1f	913
weibull_mu1fa	914
weibull_mu2f	914
weibull_mu2fa	915
weibull_p1f	915
weibull_p1fa	916
weibull_p2f	916
weibull_p2fa	917
weibull_p2_cp	917
weibull_p2_f1f	925
weibull_p2_f1fa	926
weibull_p2_f2f	926
weibull_p2_f2fa	927
weibull_p2_fd	927
weibull_p2_fdd	928
weibull_p2_ddd	928
weibull_p2_ldda	929
weibull_p2_ddd	930
weibull_p2_ddd	930
weibull_p2_lmn	931
weibull_p2_lmnp	932
weibull_p2_logf	932
weibull_p2_logfdd	933
weibull_p2_logfddd	934
weibull_p2_loglik	934
weibull_p2_logscores	935
weibull_p2_means	935
weibull_p2_mu1f	936
weibull_p2_mu1fa	937
weibull_p2_mu2f	937
weibull_p2_mu2fa	938

weibull_p2_p1f	938
weibull_p2_p1fa	939
weibull_p2_p2f	940
weibull_p2_p2fa	940
weibull_p2_pd	941
weibull_p2_pdd	941
weibull_p2_predictordata	942
weibull_p2_waic	943
weibull_pd	944
weibull_pdd	944
weibull_waic	945

Index	946
--------------	------------

adhoc_dmgs_cpmethod	<i>Generates a comment about the method</i>
---------------------	---

Description

Generates a comment about the method

Usage

adhoc_dmgs_cpmethod()

Value

String

analytic_cpmethod	<i>Generates a comment about the method</i>
-------------------	---

Description

Generates a comment about the method

Usage

analytic_cpmethod()

Value

String

bayesian_dq_4terms_v1	<i>Evaluate DMGS equation 3.3</i>
-----------------------	-----------------------------------

Description

Evaluate DMGS equation 3.3

Usage

bayesian_dq_4terms_v1(lddi, lddd, mu1, pidopi1, pidopi2, mu2, dim)

Arguments

lddi	inverse of second derivative of observed log-likelihood
lddd	third derivative of observed log-likelihood
mu1	DMGS mu1 vector
pidopi1	first part of the prior term
pidopi2	second part of the prior term
mu2	DMGS mu2 matrix
dim	number of parameters

Value

Vector

calc_revert2ml	<i>determine revert2ml or not</i>
----------------	-----------------------------------

Description

determine revert2ml or not

Usage

calc_revert2ml(v5h, v6h, t3)

Arguments

v5h	fifth parameter
v6h	sixth parameter
t3	a vector of predictors for the shape

Value

Logical

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qcauchy_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rcauchy_cp(
```

```

    n,
    x,
    d1 = 0.01,
    fd2 = 0.01,
    rust = FALSE,
    mlcp = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

dcauchy_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pcauchy_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tcauchy_cp(n, x, d1 = 0.01, fd2 = 0.01, debug = FALSE)

```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>d1</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the first parameter
<code>fd2</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the second parameter
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>waicscores</code>	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
<code>logscores</code>	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)

dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of x

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Cauchy distribution has probability density function

$$f(x; \mu, \sigma) = \frac{1}{\pi\sigma} \left(1 + \left(\frac{x - \mu}{\sigma} \right)^2 \right)^{-1}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The `cp` outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),

- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d42cauchy_example_data_v1
p=c(1:9)/10
q=qlogis_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qcauchy_cp)",
main="Cauchy: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

cauchy_f1f	<i>DMGS equation 3.3, f1 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

cauchy_f1f(y, v1, d1, v2, fd2)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_f1fa	<i>The first derivative of the density</i>
-------------	--

Description

The first derivative of the density

Usage

cauchy_f1fa(x, v1, v2)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

cauchy_f2f	<i>DMGS equation 3.3, f2 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

cauchy_f2f(y, v1, d1, v2, fd2)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_f2fa	<i>The second derivative of the density</i>
-------------	---

Description

The second derivative of the density

Usage

cauchy_f2fa(x, v1, v2)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

cauchy_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

cauchy_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

cauchy_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
cauchy_ldd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

cauchy_ldda	<i>The second derivative of the normalized log-likelihood</i>
-------------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
cauchy_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

cauchy_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
-------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
cauchy_lddd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

cauchy_lddda	<i>The third derivative of the normalized log-likelihood</i>
--------------	--

Description

The third derivative of the normalized log-likelihood

Usage

```
cauchy_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

cauchy_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
cauchy_lmn(x, v1, d1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

cauchy_lmp	<i>One component of the third derivative of the normalized log-likelihood</i>
------------	---

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
cauchy_lmp(x, v1, d1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

cauchy_logf	<i>Logf for RUST</i>
-------------	----------------------

Description

Logf for RUST

Usage

cauchy_logf(params, x)

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

cauchy_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

cauchy_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

cauchy_loglik	<i>log-likelihood function</i>
---------------	--------------------------------

Description

log-likelihood function

Usage

```
cauchy_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

cauchy_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
cauchy_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

cauchy_mu1f	<i>DMGS equation 3.3, mu1 term</i>
-------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

cauchy_mu1f(alpha, v1, d1, v2, fd2)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_mu2f	<i>DMGS equation 3.3, mu2 term</i>
-------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

cauchy_mu2f(alpha, v1, d1, v2, fd2)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_p1f

*DMGS equation 3.3, p1 term***Description**

DMGS equation 3.3, p1 term

Usage

cauchy_p1f(y, v1, d1, v2, fd2)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_p1_cp

*Cauchy Distribution with a Predictor, Predictions Based on a Calibrating Prior***Description**

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.

- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qcauchy_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rcauchy_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```

)

dcauchy_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

pcauchy_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tcauchy_p1_cp(n, x, t, d1 = 0.01, d2 = 0.01, fd3 = 0.01, debug = FALSE)

```

Arguments

x	a vector of training data values
t	a vector of predictors, such that <code>length(t)=length(x)</code>
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter

fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.

- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Cauchy distribution with a predictor has probability density function

$$f(x; a, b, \sigma) = \frac{1}{\pi\sigma} \left(1 + \left(\frac{x - \mu(a, b)}{\sigma} \right)^2 \right)^{-1}$$

where x is the random variable, $\mu = a + bt$ is the location parameter as a function of parameters a, b , and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d64cauchy_p1_example_data_v1_x
tt=fitdistcp::d64cauchy_p1_example_data_v1_t
p=c(1:9)/10
n0=10
```

```

q=qcauchy_p1_cp(x, tt, n0=n0, p=p, rust=TRUE, nrust=1000)
xmin=min(q$ml_quantiles, q$cp_quantiles);
xmax=max(q$ml_quantiles, q$cp_quantiles);
plot(q$ml_quantiles, p, xlab="quantile estimates", xlim=c(xmin, xmax),
sub="(from qcauchy_p1_cp)",
main="Cauchy w/ p1: quantile estimates");
points(q$cp_quantiles, p, col="red", lwd=2)
points(q$ru_quantiles, p, col="blue")

```

cauchy_p1_f1f

*DMGS equation 2.1, f1 term***Description**

DMGS equation 2.1, f1 term

Usage

cauchy_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_p1_f1fa	<i>The first derivative of the density</i>
----------------	--

Description

The first derivative of the density

Usage

cauchy_p1_f1fa(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

cauchy_p1_f2f	<i>DMGS equation 2.1, f2 term</i>
---------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

cauchy_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_p1_f2fa*The second derivative of the density*

Description

The second derivative of the density

Usage

cauchy_p1_f2fa(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

cauchy_p1_fd*First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

cauchy_p1_fd(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

cauchy_p1_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

cauchy_p1_fdd(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Matrix
Matrix

cauchy_p1_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
---------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

cauchy_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

cauchy_p1_ldda	<i>The second derivative of the normalized log-likelihood</i>
----------------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
cauchy_p1_ldda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

cauchy_p1_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
----------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

cauchy_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- v3 third parameter
- fd3 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

cauchy_p1_lddda	<i>The third derivative of the normalized log-likelihood</i>
-----------------	--

Description

The third derivative of the normalized log-likelihood

Usage

cauchy_p1_lddda(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

3d array

cauchy_p1_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
---------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
cauchy_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

cauchy_p1_lmp	<i>One component of the second derivative of the normalized log-likelihood</i>
---------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
cauchy_p1_lmp(x, t, v1, d1, v2, d2, v3, fd3, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

cauchy_p1_logf	<i>Logf for RUST</i>
----------------	----------------------

Description

Logf for RUST

Usage

```
cauchy_p1_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

cauchy_p1_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

cauchy_p1_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

cauchy_p1_loglik	<i>Cauchy-with-p1 observed log-likelihood function</i>
------------------	--

Description

Cauchy-with-p1 observed log-likelihood function

Usage

```
cauchy_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

cauchy_p1_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
---------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
cauchy_p1_logscores(logscores, x, t, d1, d2, fd3, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

cauchy_p1_means	<i>Cauchy distribution: RHP mean</i>
-----------------	--------------------------------------

Description

Cauchy distribution: RHP mean

Usage

```
cauchy_p1_means(t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

cauchy_p1_mu1f	<i>DMGS equation 3.3, mu1 term</i>
----------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

cauchy_p1_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3)

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_p1_mu2f	<i>DMGS equation 3.3, mu2 term</i>
----------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

cauchy_p1_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3)

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_p1_p1f	<i>DMGS equation 2.1, p1 term</i>
---------------	-----------------------------------

Description

DMGS equation 2.1, p1 term

Usage

cauchy_p1_p1f(y, t0, v1, d1, v2, d2, v3, fd3)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_p1_p2f	<i>DMGS equation 2.1, p2 term</i>
---------------	-----------------------------------

Description

DMGS equation 2.1, p2 term

Usage

cauchy_p1_p2f(y, t0, v1, d1, v2, d2, v3, fd3)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_p1_predictordata	<i>Predicted Parameter and Generalized Residuals</i>
-------------------------	--

Description

Predicted Parameter and Generalized Residuals

Usage

cauchy_p1_predictordata(predictordata, x, t, t0, params)

Arguments

- predictordata logical that indicates whether to calculate and return predictordata
- x a vector of training data values
- t a vector or matrix of predictors
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- params model parameters for calculating logf

Value

Two vectors

cauchy_p1_waic	<i>Waic</i>
----------------	-------------

Description

Waic

Usage

```
cauchy_p1_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  v3hat,  
  fd3,  
  lddi,
```

```
      lddd,  
      lambdad,  
      aderivs  
    )
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

cauchy_p2f	<i>DMGS equation 3.3, p2 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

```
cauchy_p2f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_waic	<i>Waic</i>
-------------	-------------

Description

Waic

Usage

```
cauchy_waic(waiccores, x, v1hat, d1, v2hat, fd2, lddi, lddd, lambdad, aderivs)
```

Arguments

waiccores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

crhpflat_dmgs_cpmethod
<i>Generates a comment about the method</i>

Description

Generates a comment about the method

Usage

crhpflat_dmgs_cpmethod()

Value

String

d100gamma_example_data_v1
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d101invgamma_example_data_v1
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d102invgauss_example_data_v1
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d105burr_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d10exp_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d110gev_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d11pareto_k2_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d120gpd_k1_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d150gev_p1_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d150gev_p1_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d151gev_p12_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d151gev_p12_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d152gev_p123_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d152gev_p123_example_data_v1_x

This is data to be included in my package

Description

This is data to be included in my package

d20halfnorm_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d25unif_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d30norm_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d31norm_dmgs_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d32gnorm_k3_example_data_v1
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d35lnorm_example_data_v1
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d36lnorm_dmgs_example_data_v1
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d40logis_example_data_v1
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d411st_k3_example_data_v1
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d42cauchy_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d50gumbel_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d51frechet_k1_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d52weibull_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d53gev_k3_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d55exp_p1_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d55exp_p1_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d56pareto_p1k2_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d56pareto_p1k2_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d60norm_p1_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d60norm_p1_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d61lnorm_p1_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d61lnorm_p1_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d62logis_p1_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d62logis_p1_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d63l1st_p1k3_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d63l1st_p1k3_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d64cauchy_p1_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d64cauchy_p1_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d70gumbel_p1_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d70gumbel_p1_example_data_v1_x

This is data to be included in my package

Description

This is data to be included in my package

d71frechet_p2k1_example_data_v1_t

This is data to be included in my package

Description

This is data to be included in my package

d71frechet_p2k1_example_data_v1_x

This is data to be included in my package

Description

This is data to be included in my package

d72weibull_p1_example_data_v1_t

This is data to be included in my package

Description

This is data to be included in my package

d72weibull_p1_example_data_v1_x

This is data to be included in my package

Description

This is data to be included in my package

d73weibull_p2_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d73weibull_p2_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d74gev_p1k3_example_data_v1_t
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d74gev_p1k3_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d80norm_p12_example_data_v1_t1
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d80norm_p12_example_data_v1_t2
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d80norm_p12_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d811st_p12k3_example_data_v1_t1
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d811st_p12k3_example_data_v1_t2
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d811st_p12k3_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d82weibull_p12_example_data_v1_t1
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d82weibull_p12_example_data_v1_t2
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

d82weibull_p12_example_data_v1_x
<i>This is data to be included in my package</i>

Description

This is data to be included in my package

dcauchysub	<i>Densities from MLE and RHP</i>
------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dcauchysub(x, y, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

- | | |
|---------|--|
| x | a vector of training data values |
| y | a vector of values at which to calculate the density and distribution functions |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| aderivs | logical for whether to use analytic derivatives (instead of numerical) |

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dcauchy_p1	<i>Cauchy-with-p1 density function</i>
------------	--

Description

Cauchy-with-p1 density function

Usage

```
dcauchy_p1(x, t0, ymn, slope, scale, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution
log	logical for the density evaluation

Value

Vector

dcauchy_p1sub	<i>Densities from MLE and RHP</i>
---------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dcauchy_p1sub(x, t, y, t0, d1, d2, fd3, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

deriv_copyfdd

Extract the results from derivatives and put them into f2

Description

Extract the results from derivatives and put them into f2

Usage

```
deriv_copyfdd(temp1, nx, dim)
```

Arguments

temp1	output from derivative calculations
nx	number of x values
dim	number of parameters

Value

3d array

deriv_copyld2	<i>Extract the results from derivatives and put them into ldd</i>
---------------	---

Description

Extract the results from derivatives and put them into ldd

Usage

deriv_copyld2(temp1, nx, dim)

Arguments

- | | |
|-------|-------------------------------------|
| temp1 | output from derivative calculations |
| nx | number of x values |
| dim | number of parameters |

Value

3d array

deriv_copyldd	<i>Extract the results from derivatives and put them into ldd</i>
---------------	---

Description

Extract the results from derivatives and put them into ldd

Usage

deriv_copyldd(temp1, nx, dim)

Arguments

- | | |
|-------|-------------------------------------|
| temp1 | output from derivative calculations |
| nx | number of x values |
| dim | number of parameters |

Value

Matrix

deriv_copylddd	<i>Extract the results from derivatives and put them into lddd</i>
----------------	--

Description

Extract the results from derivatives and put them into lddd

Usage

```
deriv_copylddd(temp1, nx, dim)
```

Arguments

temp1	output from derivative calculations
nx	number of x values
dim	number of parameters

Value

3d array

dexpsub	<i>Densities from MLE and RHP</i>
---------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dexpsub(x, y, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dexp_p1	<i>Exponential-with-p1 density function</i>
---------	---

Description

Exponential-with-p1 density function

Usage

```
dexp_p1(x, t0, ymn, slope, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
log	logical for the density evaluation

Value

Vector

dexp_p1sub	<i>Densities from MLE and RHP</i>
------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dexp_p1sub(x, t, y, t0, d1, d2, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dfrechetsub	<i>Densities from MLE and RHP</i>
-------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dfrechetsub(x, y, kloc, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
kloc	the known location parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dfrechetsub	<i>Frechet_k1-with-p2 density function</i>
-------------	--

Description

Frechet_k1-with-p2 density function

Usage

```
dfrechetsub(x, t0, ymn, slope, lambda, log = FALSE, kloc)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
lambda	the lambda parameter of the distribution
log	logical for the density evaluation
kloc	the known location parameter

Value

Vector

dfrechet_p2k1sub	<i>Densities from MLE and RHP</i>
------------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dfrechet_p2k1sub(x, t, y, t0, d1, d2, fd3, kloc, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgamma_{sub}
Densities from MLE and RHP

Description

Densities from MLE and RHP

Usage

```
dgammasub(x, y, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgev_{sub}
Densities for 5 predictions

Description

Densities for 5 predictions

Usage

```
dgevsub(
  x,
  y,
  ics,
  d1 = 0.01,
  fd2 = 0.01,
  d3 = 0.01,
  customprior,
  minxi,
  maxx,
  extramodels = FALSE,
  aderivs = TRUE
)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
d3	the delta used in the numerical derivatives with respect to the parameter
customprior	a custom value for the slope of the log prior at the maxlik estimate
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
extramodels	logical that indicates whether to add three additional prediction models
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgev_k3sub	<i>Densities from MLE and RHP</i>
------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dgev_k3sub(x, y, d1 = 0.01, fd2 = 0.01, kshape, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgev_p1	<i>GEVD-with-p1: Density function</i>
---------	---------------------------------------

Description

GEVD-with-p1: Density function

Usage

```
dgev_p1(x, t0, ymn, slope, sigma, xi, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
log	logical for the density evaluation

Value

Vector

dgev_p12	<i>GEVD-with-p1: Density function</i>
----------	---------------------------------------

Description

GEVD-with-p1: Density function

Usage

```
dgev_p12(x, t1, t2, ymn, slope, sigma1, sigma2, xi, log = FALSE)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi	the shape parameter of the distribution
log	logical for the density evaluation

Value

Vector

dgev_p123	<i>GEVD-with-p1: Density function</i>
-----------	---------------------------------------

Description

GEVD-with-p1: Density function

Usage

```
dgev_p123(x, t1, t2, t3, ymn, slope, sigma1, sigma2, xi1, xi2, log = FALSE)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution
log	logical for the density evaluation

Value

Vector

dgev_p123sub

*Densities for 5 predictions***Description**

Densities for 5 predictions

Usage

```
dgev_p123sub(
  x,
  t1,
  t2,
  t3,
  y,
  t01,
  t02,
  t03,
  ics,
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  d6 = 0.01,
  extramodels,
  debug,
  aderivs = TRUE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
ics	initial conditions for the maximum likelihood search
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
d3	the delta used in the numerical derivatives with respect to the parameter

d4	the delta used in the numerical derivatives with respect to the parameter
d5	the delta used in the numerical derivatives with respect to the parameter
d6	the delta used in the numerical derivatives with respect to the parameter
extramodels	logical that indicates whether to add three additional prediction models
debug	debug flag
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgev_p12sub	<i>Densities for 5 predictions</i>
-------------	------------------------------------

Description

Densities for 5 predictions

Usage

```
dgev_p12sub(
  x,
  t1,
  t2,
  y,
  t01,
  t02,
  ics,
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  minxi,
  maxx,
  debug,
  extramodels = FALSE,
  aderivs = TRUE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
ics	initial conditions for the maximum likelihood search
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
d3	the delta used in the numerical derivatives with respect to the parameter
d4	the delta used in the numerical derivatives with respect to the parameter
d5	the delta used in the numerical derivatives with respect to the parameter
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
debug	debug flag
extramodels	logical that indicates whether to add three additional prediction models
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgev_p1k3

GEV-with-known-shape-with-p1 density function

Description

GEV-with-known-shape-with-p1 density function

Usage

```
dgev_p1k3(x, t0, ymn, slope, sigma, log = FALSE, kshape)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
log	logical for the density evaluation
kshape	the known shape parameter

Value

Vector

dgev_p1k3sub	<i>Densities from MLE and RHP</i>
--------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dgev_p1k3sub(x, t, y, t0, d1, d2, fd3, kshape, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgev_p1sub	<i>Densities for 5 predictions</i>
------------	------------------------------------

Description

Densities for 5 predictions

Usage

```
dgev_p1sub(
  x,
  t,
  y,
  t0,
  ics,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  d4 = 0.01,
  minxi,
  maxx,
  extramodels = FALSE,
  aderivs = TRUE
)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
ics	initial conditions for the maximum likelihood search
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
d4	the delta used in the numerical derivatives with respect to the parameter
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
extramodels	logical that indicates whether to add three additional prediction models
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgnorm_k3sub	<i>Densities from MLE and RHP</i>
--------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dgnorm_k3sub(x, y, d1 = 0.01, fd2 = 0.01, kbeta, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgpdsb	<i>Densities for 5 predictions</i>
--------	------------------------------------

Description

Densities for 5 predictions

Usage

```
dgpdsb(
  x,
  y,
  ics,
  fd1 = 0.01,
  d2 = 0.01,
  kloc = 0,
  dlogpi = 0,
  minxi,
  maxx,
  extramodels = FALSE,
  aderivs = TRUE
)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
dlogpi	gradient of the log prior
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
extramodels	logical that indicates whether to add three additional prediction models
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

 dgumbelsub

Densities from MLE and RHP

Description

Densities from MLE and RHP

Usage

```
dgumbelsub(x, y, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgumbel_p1	<i>Gumbel-with-p1 density function</i>
------------	--

Description

Gumbel-with-p1 density function

Usage

```
dgumbel_p1(x, t0, ymn, slope, sigma, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
log	logical for the density evaluation

Value

Vector

dgumbel_p1sub	<i>Densities from MLE and RHP</i>
---------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dgumbel_p1sub(x, t, y, t0, d1, d2, fd3, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dhalfnorm_{sub}
Densities from MLE and RHP

Description

Densities from MLE and RHP

Usage

```
dhalfnormsub(x, y, fd1 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dinvgamma_{sub}
Densities from MLE and cp

Description

Densities from MLE and cp

Usage

```
dinvgammasub(x, y, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dinvgausssub	<i>Densities from MLE and RHP</i>
--------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dinvgausssub(x, y, prior, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
prior	logical indicating which prior to use
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dlnormsub	<i>Densities from MLE and RHP</i>
-----------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dlnormsub(x, y, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dlnorm_dmgssub	<i>Densities from MLE and RHP</i>
----------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dlnorm_dmgssub(x, y, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dlnorm_p1	<i>Normal-with-p1 density function</i>
-----------	--

Description

Normal-with-p1 density function

Usage

```
dlnorm_p1(x, t0, ymn, slope, sigma, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
log	logical for the density evaluation

Value

Vector

dlnorm_p1sub

*Densities from MLE and RHP***Description**

Densities from MLE and RHP

Usage

```
dlnorm_p1sub(x, t, y, t0, debug = FALSE, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
debug	debug flag
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dlogis2sub

*Densities from MLE and RHP***Description**

Densities from MLE and RHP

Usage

```
dlogis2sub(x, y, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dlogis_p1	<i>Logistic-with-p1 density function</i>
-----------	--

Description

Logistic-with-p1 density function

Usage

```
dlogis_p1(x, t0, ymn, slope, scale, log = FALSE)
```

Arguments

- x a vector of training data values
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- ymn the location parameter of the function of the predictor
- slope the slope of the function of the predictor
- scale the scale parameter of the distribution
- log logical for the density evaluation

Value

Vector

dlogis_p1sub	<i>Densities from MLE and RHP</i>
--------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dlogis_p1sub(x, t, y, t0, d1, d2, fd3, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dlst_k3sub	<i>Densities from MLE and RHP</i>
------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dlst_k3sub(x, y, d1 = 0.01, fd2 = 0.01, kdf, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dlst_p1k3	<i>LST-with-p1 density function</i>
-----------	-------------------------------------

Description

LST-with-p1 density function

Usage

dlst_p1k3(x, t0, ymn, slope, sigma, log = FALSE, kdf)

Arguments

- x a vector of training data values
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- ymn the location parameter of the function of the predictor
- slope the slope of the function of the predictor
- sigma the sigma parameter of the distribution
- log logical for the density evaluation
- kdf the known degrees of freedom parameter

Value

Vector

dlst_p1k3sub	<i>Densities from MLE and RHP</i>
--------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

dlst_p1k3sub(x, t, y, t0, d1, d2, fd3, kdf, aderivs = TRUE)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dmgs	<i>Evaluate DMGS equation 3.3</i>
------	-----------------------------------

Description

Evaluate DMGS equation 3.3

Usage

```
dmgs(lddi, lddd, mu1, pidopi, mu2, dim)
```

Arguments

lddi	inverse of second derivative of observed log-likelihood
lddd	third derivative of observed log-likelihood
mu1	DMGS mu1 vector
pidopi	derivative of log prior
mu2	DMGS mu2 matrix
dim	number of parameters

Value

Vector

dnormsub	<i>Densities from MLE and RHP</i>
----------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dnormsub(x, y, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dnorm_dmgssub	<i>Densities from MLE and RHP</i>
---------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dnorm_dmgssub(x, y, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dnorm_p1	<i>Normal-with-p1 density function</i>
----------	--

Description

Normal-with-p1 density function

Usage

```
dnorm_p1(x, t0, ymn, slope, sigma, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
log	logical for the density evaluation

Value

Vector

dnorm_p1sub	<i>Densities from MLE and RHP</i>
-------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dnorm_p1sub(x, t, y, t0, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

<code>dnorm_p1_formula</code>	<i>Linear regression formula, densities</i>
-------------------------------	---

Description

Linear regression formula, densities

Usage

```
dnorm_p1_formula(y, ta, ta0, nx, muhat0, v3hat)
```

Arguments

- `y` a vector of values at which to calculate the density and distribution functions
- `ta` predictor residuals
- `ta0` predictor residual at the point being predicted
- `nx` length of training data
- `muhat0` muhat at the point being predicted
- `v3hat` third parameter

Value

Vector

<code>dpareto_k2_sub</code>	<i>Densities from MLE and RHP</i>
-----------------------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dpareto_k2_sub(x, y, kscale, aderivs = TRUE)
```

Arguments

- `x` a vector of training data values
- `y` a vector of values at which to calculate the density and distribution functions
- `kscale` the known scale parameter
- `aderivs` logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dpareto_p1k2	<i>pareto_k1-with-p2 density function</i>
--------------	---

Description

pareto_k1-with-p2 density function

Usage

```
dpareto_p1k2(x, t0, ymn, slope, kscale, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
kscale	the known scale parameter
log	logical for the density evaluation

Value

Vector

dpareto_p1k2sub	<i>Densities from MLE and RHP</i>
-----------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dpareto_p1k2sub(x, t, y, t0, d1, d2, kscale, aderivs = TRUE, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)
debug	debug flag

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dunif_formula

Predictive PDFs

Description

Predictive PDFs

Usage

```
dunif_formula(x, y)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions

Value

Two vectors

dweibullsub

Densities from MLE and RHP

Description

Densities from MLE and RHP

Usage

```
dweibullsub(x, y, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dweibull_p2	<i>Weibull-with-p1 density function</i>
-------------	---

Description

Weibull-with-p1 density function

Usage

```
dweibull_p2(x, t0, shape, ymn, slope, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
shape	the shape parameter of the distribution
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
log	logical for the density evaluation

Value

Vector

dweibull_p2sub	<i>Densities from MLE and RHP</i>
----------------	-----------------------------------

Description

Densities from MLE and RHP

Usage

```
dweibull_p2sub(x, t, y, t0, fd1, d2, d3, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
d3	the delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

exp_cp

Exponential Distribution Predictions Based on a Calibrating Prior

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qexp_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)
```

```

rexp_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE, aderivs = TRUE)

dexp_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)

pexp_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)

texp_cp(n, x, debug = FALSE)

```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The exponential distribution has exceedance distribution function

$$S(x; \lambda) = \exp(-\lambda x)$$

where $x \geq 0$ is the random variable and $\lambda > 0$ is the rate parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\lambda) \propto \frac{1}{\lambda}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d10exp_example_data_v1
p=c(1:9)/10
q=qexp_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qexp_cp)",
main="Exponential: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

exp_f1f	<i>DMGS equation 2.1, f1 term</i>
---------	-----------------------------------

Description

DMGS equation 2.1, f1 term

Usage

```
exp_f1f(y, v1, fd1)
```

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

exp_f1fa	<i>The first derivative of the density</i>
----------	--

Description

The first derivative of the density

Usage

```
exp_f1fa(x, v1)
```

Arguments

- x a vector of training data values
- v1 first parameter

Value

Vector

exp_f2f	<i>DMGS equation 2.1, f2 term</i>
---------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

exp_f2f(y, v1, fd1)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

exp_f2fa	<i>The second derivative of the density</i>
----------	---

Description

The second derivative of the density

Usage

exp_f2fa(x, v1)

Arguments

- x a vector of training data values
- v1 first parameter

Value

Matrix

exp_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_fd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

exp_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_fdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

exp_l111	<i>Third derivative of the normalized log-likelihood</i>
----------	--

Description

Third derivative of the normalized log-likelihood

Usage

```
exp_l111(x, v1, fd1)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Scalar value

exp_1dd	<i>The second derivative of the normalized log-likelihood</i>
---------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
exp_1dd(x, v1, fd1)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

exp_ldda	<i>The second derivative of the normalized log-likelihood</i>
----------	---

Description

The second derivative of the normalized log-likelihood

Usage

exp_ldda(x, v1)

Arguments

- x a vector of training data values
- v1 first parameter

Value

Matrix

exp_ldddd	<i>Third derivative tensor of the log-likelihood</i>
-----------	--

Description

Third derivative tensor of the log-likelihood

Usage

exp_ldddd(x, v1, fd1)

Arguments

- x a vector of training data values
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

exp_lddda	<i>The third derivative of the normalized log-likelihood</i>
-----------	--

Description

The third derivative of the normalized log-likelihood

Usage

```
exp_lddda(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

3d array

exp_logf	<i>Logf for RUST</i>
----------	----------------------

Description

Logf for RUST

Usage

```
exp_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

exp_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

exp_logfdd(x, v1)

Arguments

- x a vector of training data values
- v1 first parameter

Value

Matrix

exp_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

exp_logfddd(x, v1)

Arguments

- x a vector of training data values
- v1 first parameter

Value

3d array

exp_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
---------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
exp_logscores(logscores, x)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values

Value

Two scalars

exp_p1fa	<i>The first derivative of the cdf</i>
----------	--

Description

The first derivative of the cdf

Usage

```
exp_p1fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

exp_p1_cp

*Exponential Distribution with a Predictor, Predictions Based on a Calibrating Prior***Description**

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qexp_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
```

```
    centering = TRUE,  
    debug = FALSE,  
    aderivs = TRUE  
  )
```

```
rexp_p1_cp(  
  n,  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  d1 = 0.01,  
  d2 = 0.01,  
  rust = FALSE,  
  mlcp = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
dexp_p1_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  y = x,  
  d1 = 0.01,  
  d2 = 0.01,  
  rust = FALSE,  
  nrust = 1000,  
  centering = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
pexp_p1_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  y = x,  
  d1 = 0.01,  
  d2 = 0.01,  
  rust = FALSE,  
  nrust = 1000,  
  centering = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
texp_p1_cp(n, x, t, d1 = 0.01, d2 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that <code>length(t)=length(x)</code>
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	the fractional delta used in the numerical derivatives with respect to the location parameter
d2	the fractional delta used in the numerical derivatives with respect to the slope parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.

- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of x

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The exponential distribution with a predictor has exceedance distribution function

$$S(x; a, b) = \exp(-x\lambda(a, b))$$

where $x \geq 0$ is the random variable and $\lambda(a, b) = e^{-a-bt}$ is the rate parameter, modelled as a function of the parameters a, b and a predictor t .

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b) \propto 1$$

. as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean (`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d55exp_p1_example_data_v1_x
tt=fitdistcp::d55exp_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qexp_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qexp_p1_cp)",
main="Exponential w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

exp_p1_f1f	DMGS equation 2.1, f1 term
------------	----------------------------

Description

DMGS equation 2.1, f1 term

Usage

```
exp_p1_f1f(y, t0, v1, d1, v2, d2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

exp_p1_f1fa	<i>The first derivative of the density</i>
-------------	--

Description

The first derivative of the density

Usage

exp_p1_f1fa(x, t, v1, v2)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter

Value

Vector

exp_p1_f2f	<i>DMGS equation 2.1, f2 term</i>
------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

exp_p1_f2f(y, t0, v1, d1, v2, d2)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter

Value

3d array

exp_p1_f2fa	<i>The second derivative of the density</i>
-------------	---

Description

The second derivative of the density

Usage

```
exp_p1_f2fa(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_fd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Vector

exp_p1_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_fdd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
exp_p1_ldd(x, t, v1, d1, v2, d2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

exp_p1_ldda	<i>The second derivative of the normalized log-likelihood</i>
-------------	---

Description

The second derivative of the normalized log-likelihood

Usage

exp_p1_ldda(x, t, v1, v2)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter

Value

Matrix

exp_p1_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
-------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

exp_p1_lddd(x, t, v1, d1, v2, d2)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

exp_p1_lddda	<i>The third derivative of the normalized log-likelihood</i>
--------------	--

Description

The third derivative of the normalized log-likelihood

Usage

exp_p1_lddda(x, t, v1, v2)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter

Value

3d array

exp_p1_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

exp_p1_lmn(x, t, v1, d1, v2, d2, mm, nn)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- mm an index for which derivative to calculate
- nn an index for which derivative to calculate

Value

Scalar value

exp_p1_lmp	<i>One component of the third derivative of the normalized log-likelihood</i>
------------	---

Description

One component of the third derivative of the normalized log-likelihood

Usage

exp_p1_lmp(x, t, v1, d1, v2, d2, mm, nn, rr)

Arguments

- | | |
|----|---|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| mm | an index for which derivative to calculate |
| nn | an index for which derivative to calculate |
| rr | an index for which derivative to calculate |

Value

Scalar value

exp_p1_logf	<i>Logf for RUST</i>
-------------	----------------------

Description

Logf for RUST

Usage

exp_p1_logf(params, x, t)

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

exp_p1_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

exp_p1_logfdd(x, t, v1, v2)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

exp_p1_logfddd(x, t, v1, v2)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter

Value

3d array

exp_p1_loglik	<i>observed log-likelihood function</i>
---------------	---

Description

observed log-likelihood function

Usage

exp_p1_loglik(vv, x, t)

Arguments

- vv parameters
- x a vector of training data values
- t a vector or matrix of predictors

Value

Scalar value.

exp_p1_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
exp_p1_logscores(logscores, x, t, d1, d2, aderivs)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

exp_p1_means	<i>exp distribution: RHP means</i>
--------------	------------------------------------

Description

exp distribution: RHP means

Usage

```
exp_p1_means(means, t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

exp_p1_mu1f	<i>DMGS equation 3.3, mu1 term</i>
-------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

exp_p1_mu1f(alpha, t0, v1, d1, v2, d2)

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

exp_p1_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
--------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

exp_p1_mu1fa(alpha, t, v1, v2)

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |

Value

Vector

exp_p1_mu2f	<i>DMGS equation 3.3, mu2 term</i>
-------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

exp_p1_mu2f(alpha, t0, v1, d1, v2, d2)

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |

Value

3d array

exp_p1_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
--------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

```
exp_p1_mu2fa(alpha, t, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_p1f	<i>DMGS equation 2.1, p1 term</i>
------------	-----------------------------------

Description

DMGS equation 2.1, p1 term

Usage

```
exp_p1_p1f(y, t0, v1, d1, v2, d2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

exp_p1_p1fa	<i>The first derivative of the cdf</i>
Description	
The first derivative of the cdf	
Usage	
exp_p1_p1fa(x, t, v1, v2)	
Arguments	
x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
Value	
Vector	
exp_p1_p2f	<i>DMGS equation 2.1, p2 term</i>
Description	
DMGS equation 2.1, p2 term	
Usage	
exp_p1_p2f(y, t0, v1, d1, v2, d2)	
Arguments	
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
Value	
3d array	

exp_p1_p2fa

*The second derivative of the cdf***Description**

The second derivative of the cdf

Usage

```
exp_p1_p2fa(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_pd

*First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol***Description**

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_pd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Vector

exp_p1_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_pdd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_predictordata	<i>Predicted Parameter and Generalized Residuals</i>
----------------------	--

Description

Predicted Parameter and Generalized Residuals

Usage

```
exp_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

exp_p1_waic	<i>Waic</i>
-------------	-------------

Description

Waic

Usage

```
exp_p1_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

exp_p2fa	<i>The second derivative of the cdf</i>
----------	---

Description

The second derivative of the cdf

Usage

```
exp_p2fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

exp_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_pd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

exp_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_pdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

exp_waic	<i>Waicscores</i>
----------	-------------------

Description

Waicscores

Usage

```
exp_waic(waicscores, x, v1hat, fd1, aderivs)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

fixgevrange	<i>Deal with situations in which the user wants d or p outside the GEV range</i>
-------------	--

Description

Deal with situations in which the user wants d or p outside the GEV range

Usage

```
fixgevrange(y, v1, v2, v3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

fixgpdrange	<i>Deal with situations in which the user wants d or p outside the GPD range</i>
-------------	--

Description

Deal with situations in which the user wants d or p outside the GPD range

Usage

```
fixgpdrange(y, v1, v2, v3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qfrechet_k1_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kloc = 0,
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)
```

```

rfrechet_k1_cp(
  n,
  x,
  kloc = 0,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

```

```

dfrechet_k1_cp(
  x,
  y = x,
  kloc = 0,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

```

```

pfrechet_k1_cp(
  x,
  y = x,
  kloc = 0,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

```

```

tfrechet_k1_cp(n, x, kloc = 0, fd1 = 0.01, fd2 = 0.01, debug = FALSE)

```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kloc	the known location parameter
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter

means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times $-1/2$.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Frechet distribution has distribution function

$$F(x; \sigma, \lambda) = \exp \left(- \left(\frac{x - \mu}{\sigma} \right)^{-\lambda} \right)$$

where $x > \mu$ is the random variable, $\sigma > 0, \lambda > 0$ are the parameters and we consider μ to be known (hence the k1 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma, \lambda) \propto \frac{1}{\sigma \lambda}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d51frechet_k1_example_data_v1
p=c(1:9)/10
q=qfrechet_k1_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
```

```
xmax=max(q$m1_quantiles,q$cp_quantiles);
plot(q$m1_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qfrechet_k1_cp)",
main="Frechet: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

frechet_k1_f1f	<i>DMGS equation 3.3, f1 term</i>
----------------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

```
frechet_k1_f1f(y, v1, fd1, v2, fd2, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Matrix

frechet_k1_f1fa	<i>The first derivative of the density</i>
-----------------	--

Description

The first derivative of the density

Usage

```
frechet_k1_f1fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

frechet_k1_f2f	<i>DMGS equation 3.3, f2 term</i>
----------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

frechet_k1_f2f(y, v1, fd1, v2, fd2, kloc)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

frechet_k1_f2fa	<i>The second derivative of the density</i>
-----------------	---

Description

The second derivative of the density

Usage

```
frechet_k1_f2fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

frechet_k1_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

frechet_k1_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

frechet_k1_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
----------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
frechet_k1_ldd(x, v1, fd1, v2, fd2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Square scalar matrix

frechet_k1_ldda	<i>The second derivative of the normalized log-likelihood</i>
-----------------	---

Description

The second derivative of the normalized log-likelihood

Usage

frechet_k1_ldda(x, v1, v2, kloc)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- kloc the known location parameter

Value

Matrix

frechet_k1_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
-----------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

frechet_k1_lddd(x, v1, fd1, v2, fd2, kloc)

Arguments

- x a vector of training data values
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter
- kloc the known location parameter

Value

Cubic scalar array

frechet_k1_lddda	<i>The third derivative of the normalized log-likelihood</i>
------------------	--

Description

The third derivative of the normalized log-likelihood

Usage

frechet_k1_lddda(x, v1, v2, kloc)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- kloc the known location parameter

Value

3d array

frechet_k1_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
----------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

frechet_k1_lmn(x, v1, fd1, v2, fd2, kloc, mm, nn)

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

frechet_k1_lmp	<i>One component of the third derivative of the normalized log-likelihood</i>
----------------	---

Description

One component of the third derivative of the normalized log-likelihood

Usage

frechet_k1_lmp(x, v1, fd1, v2, fd2, kloc, mm, nn, rr)

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

frechet_k1_logf	<i>Logf for RUST</i>
-----------------	----------------------

Description

Logf for RUST

Usage

```
frechet_k1_logf(params, x, kloc)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kloc	the known location parameter

Value

Scalar value.

frechet_k1_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

frechet_k1_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

frechet_k1_mu1f	<i>DMGS equation 3.3, mu1 term</i>
-----------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

```
frechet_k1_mu1f(alpha, v1, fd1, v2, fd2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Matrix

frechet_k1_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
------------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

```
frechet_k1_mu1fa(alpha, v1, v2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

frechet_k1_mu2f	<i>DMGS equation 3.3, mu2 term</i>
-----------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

```
frechet_k1_mu2f(alpha, v1, fd1, v2, fd2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

frechet_k1_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
------------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

frechet_k1_mu2fa(alpha, v1, v2, kloc)

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| v2 | second parameter |
| kloc | the known location parameter |

Value

Matrix

frechet_k1_p1f	<i>DMGS equation 3.3, p1 term</i>
----------------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

frechet_k1_p1f(y, v1, fd1, v2, fd2, kloc)

Arguments

- | | |
|------|--|
| y | a vector of values at which to calculate the density and distribution functions |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| kloc | the known location parameter |

Value

Matrix

frechet_k1_p1fa	<i>The first derivative of the cdf</i>
-----------------	--

Description

The first derivative of the cdf

Usage

frechet_k1_p1fa(x, v1, v2, kloc)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- kloc the known location parameter

Value

Vector

frechet_k1_p2f	<i>DMGS equation 3.3, p2 term</i>
----------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

frechet_k1_p2f(y, v1, fd1, v2, fd2, kloc)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter
- kloc the known location parameter

Value

3d array

frechet_k1_p2fa	<i>The second derivative of the cdf</i>
-----------------	---

Description

The second derivative of the cdf

Usage

frechet_k1_p2fa(x, v1, v2, kloc)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- kloc the known location parameter

Value

Matrix

frechet_k1_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

frechet_k1_pd(x, v1, v2, v3)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Vector

frechet_k1_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_pdd(x, v1, v2, v3)
```

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Matrix

frechet_k1_waic	<i>Waic</i>
-----------------	-------------

Description

Waic

Usage

```
frechet_k1_waic(  
  waicscores,  
  x,  
  v1hat,  
  fd1,  
  v2hat,  
  fd2,  
  kloc,
```

```
    lddi,  
    lddd,  
    lambdad,  
    aderivs  
  )
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

frechet_loglik	<i>log-likelihood function</i>
----------------	--------------------------------

Description

log-likelihood function

Usage

```
frechet_loglik(vv, x, kloc)
```

Arguments

vv	parameters
x	a vector of training data values
kloc	the known location parameter

Value

Scalar value.

frechet_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
-------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
frechet_logscores(logscores, x, fd1 = 0.01, fd2 = 0.01, kloc, aderivs)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

frechet_means	<i>MLE and RHP predictive means</i>
---------------	-------------------------------------

Description

MLE and RHP predictive means

Usage

```
frechet_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2, kloc)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lamdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kloc	the known location parameter

Value

Two scalars

frechet_p2k1_cp	<i>Frechet Distribution with Predictor, Predictions Based on a Calibrating Prior</i>
-----------------	--

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qfrechet_p2k1_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  p = seq(0.1, 0.9, 0.1),  
  d1 = 0.01,  
  d2 = 0.01,  
  fd3 = 0.01,  
  means = FALSE,  
  waicscores = FALSE,  
  logscores = FALSE,  
  kloc = 0,  
  dmgs = TRUE,  
  rust = FALSE,  
  nrust = 1e+05,  
  predictordata = TRUE,  
  centering = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
rfrechet_p2k1_cp(  
  n,  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  d1 = 0.01,  
  d2 = 0.01,  
  fd3 = 0.01,  
  kloc = 0,  
  rust = FALSE,  
  mlcp = TRUE,  
  centering = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
dfrechet_p2k1_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  y = x,  
  d1 = 0.01,  
  d2 = 0.01,
```

```

    fd3 = 0.01,
    kloc = 0,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

```

```

pfrechet_p2k1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kloc = 0,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

```

```

tfrechet_p2k1_cp(
  n,
  x,
  t,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kloc = 0,
  debug = FALSE
)

```

Arguments

x	a vector of training data values
t	a vector of predictors, such that <code>length(t)=length(x)</code>
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter

fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
kloc	the known location parameter
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Frechet distribution with predictor has distribution function

$$F(x; a, b, \lambda) = \exp \left(- \left(\frac{x - \mu}{\sigma(a, b)} \right)^{-\lambda} \right)$$

where $x > \mu$ is the random variable, $\sigma = e^{a+bt}$ is the scale parameter, modelled as a function of parameters a, b and predictor t , and $\lambda > 0$ is the shape parameter. We consider μ to be known (hence the k1 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b) \propto 1$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean (`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d71frechet_p2k1_example_data_v1_x
tt=fitdistcp::d71frechet_p2k1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qfrechet_p2k1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$m1_quantiles,q$cp_quantiles);
xmax=max(q$m1_quantiles,q$cp_quantiles);
plot(q$m1_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qfrechet_p2k1_cp)",
main="Frechet w/ p2: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

frechet_p2k1_f1f	<i>DMGS equation 2.1, f1 term</i>
------------------	-----------------------------------

Description

DMGS equation 2.1, f1 term

Usage

```
frechet_p2k1_f1f(y, t0, v1, d1, v2, d2, v3, fd3, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Matrix

frechet_p2k1_f1fa	<i>The first derivative of the density</i>
-------------------	--

Description

The first derivative of the density

Usage

frechet_p2k1_f1fa(x, t, v1, v2, v3, kloc)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter
- kloc the known location parameter

Value

Vector

frechet_p2k1_f2f	<i>DMGS equation 2.1, f2 term</i>
------------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

frechet_p2k1_f2f(y, t0, v1, d1, v2, d2, v3, fd3, kloc)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

frechet_p2k1_f2fa	<i>The second derivative of the density</i>
-------------------	---

Description

The second derivative of the density

Usage

frechet_p2k1_f2fa(x, t, v1, v2, v3, kloc)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Matrix

frechet_p2k1_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_fd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

frechet_p2k1_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_fdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

frechet_p2k1_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
------------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

frechet_p2k1_ldd(x, t, v1, d1, v2, d2, v3, fd3, kloc)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Square scalar matrix

frechet_p2k1_ldda	<i>The second derivative of the normalized log-likelihood</i>
-------------------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
frechet_p2k1_ldda(x, t, v1, v2, v3, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Matrix

frechet_p2k1_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
-------------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
frechet_p2k1_lddd(x, t, v1, d1, v2, d2, v3, fd3, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Cubic scalar array

frechet_p2k1_lddda	<i>The third derivative of the normalized log-likelihood</i>
--------------------	--

Description

The third derivative of the normalized log-likelihood

Usage

frechet_p2k1_lddda(x, t, v1, v2, v3, kloc)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

3d array

frechet_p2k1_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
------------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
frechet_p2k1_lmn(x, t, v1, d1, v2, d2, v3, fd3, kloc, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

frechet_p2k1_lmp	<i>One component of the second derivative of the normalized log-likelihood</i>
------------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
frechet_p2k1_lmp(x, t, v1, d1, v2, d2, v3, fd3, kloc, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

frechet_p2k1_logf	<i>Logf for RUST</i>
-------------------	----------------------

Description

Logf for RUST

Usage

frechet_p2k1_logf(params, x, t, kloc)

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors
kloc	the known location parameter

Value

Scalar value.

frechet_p2k1_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_logfdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

frechet_p2k1_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_logfddd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

frechet_p2k1_loglik	<i>observed log-likelihood function</i>
---------------------	---

Description

observed log-likelihood function

Usage

```
frechet_p2k1_loglik(vv, x, t, kloc)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors
kloc	the known location parameter

Value

Scalar value.

frechet_p2k1_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
------------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
frechet_p2k1_logscores(logscores, x, t, d1, d2, fd3, kloc, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

frechet_p2k1_means	<i>frechet_k1 distribution: RHP mean</i>
--------------------	--

Description

frechet_k1 distribution: RHP mean

Usage

```
frechet_p2k1_means(  
  means,  
  t0,  
  ml_params,  
  lddi,  
  lddd,  
  lambdad_rhp,  
  nx,  
  dim,  
  kloc  
)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kloc	the known location parameter

Value

Two scalars

frechet_p2k1_mu1f	<i>DMGS equation 3.3, mu1 term</i>
-------------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

```
frechet_p2k1_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Matrix

frechet_p2k1_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
--------------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

```
frechet_p2k1_mu1fa(alpha, t, v1, v2, v3, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Vector

frechet_p2k1_mu2f	<i>DMGS equation 3.3, mu2 term</i>
-------------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

frechet_p2k1_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3, kloc)

Arguments

- | | |
|-------|--|
| alpha | a vector of values of alpha (one minus probability) |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| v3 | third parameter |
| fd3 | the fractional delta used in the numerical derivatives with respect to the parameter |
| kloc | the known location parameter |

Value

3d array

frechet_p2k1_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
--------------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

frechet_p2k1_mu2fa(alpha, t, v1, v2, v3, kloc)

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Matrix

frechet_p2k1_p1f	<i>DMGS equation 2.1, p1 term</i>
------------------	-----------------------------------

Description

DMGS equation 2.1, p1 term

Usage

frechet_p2k1_p1f(y, t0, v1, d1, v2, d2, v3, fd3, kloc)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Matrix

frechet_p2k1_p1fa	<i>The first derivative of the cdf</i>
-------------------	--

Description

The first derivative of the cdf

Usage

frechet_p2k1_p1fa(x, t, v1, v2, v3, kloc)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter
- kloc the known location parameter

Value

Vector

frechet_p2k1_p2f	<i>DMGS equation 2.1, p2 term</i>
------------------	-----------------------------------

Description

DMGS equation 2.1, p2 term

Usage

frechet_p2k1_p2f(y, t0, v1, d1, v2, d2, v3, fd3, kloc)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter

v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

frechet_p2k1_p2fa	<i>The second derivative of the cdf</i>
-------------------	---

Description

The second derivative of the cdf

Usage

frechet_p2k1_p2fa(x, t, v1, v2, v3, kloc)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Matrix

frechet_p2k1_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_pd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

frechet_p2k1_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_pdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

frechet_p2k1_predictordata
<i>Predicted Parameter and Generalized Residuals</i>

Description

Predicted Parameter and Generalized Residuals

Usage

```
frechet_p2k1_predictordata(predictordata, x, t, t0, params, kloc)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf
kloc	the known location parameter

Value

Two vectors

frechet_p2k1_waic	Waic
-------------------	------

Description

Waic

Usage

```
frechet_p2k1_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  v3hat,  
  fd3,  
  kloc,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gamma_cp

Gamma Distribution Predictions Based on a Calibrating Prior

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgamma_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  prior = "type 1",
  debug = FALSE,
```

```

    aderivs = TRUE
  )

  rgamma_cp(
    n,
    x,
    fd1 = 0.01,
    fd2 = 0.01,
    rust = FALSE,
    mlcp = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  dgamma_cp(
    x,
    y = x,
    fd1 = 0.01,
    fd2 = 0.01,
    rust = FALSE,
    nrust = 1000,
    debug = FALSE,
    aderivs = TRUE
  )

  pgamma_cp(
    x,
    y = x,
    fd1 = 0.01,
    fd2 = 0.01,
    rust = FALSE,
    nrust = 1000,
    debug = FALSE,
    aderivs = TRUE
  )

  tgamma_cp(n, x, fd1 = 0.01, fd2 = 0.01, debug = FALSE)

```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
prior	logical indicating which prior to use
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times $-1/2$.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Gamma distribution has probability density function

$$f(x; \alpha, \sigma) = \frac{1}{\sigma^\alpha \Gamma(\alpha)} x^{\alpha-1} e^{-x/\sigma}$$

where $x \geq 0$ is the random variable and $\alpha > 0, \sigma > 0$ are the parameters.

The calibrating prior we use is

$$\pi(\alpha, \sigma) \propto \frac{1}{\alpha \sigma}$$

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If rust=TRUE:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If rust=TRUE:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If rust=TRUE:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d100gamma_example_data_v1
p=c(1:9)/10
q=qgamma_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgamma_cp)",
main="Gamma: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

gamma_f1f	<i>DMGS equation 3.3, f1 term</i>
-----------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

gamma_f1f(y, v1, fd1, v2, fd2)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Matrix

gamma_f1fa	<i>The first derivative of the density</i>
------------	--

Description

The first derivative of the density

Usage

gamma_f1fa(x, v1, v2)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |

Value

Vector

`gamma_f2f`*DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage`gamma_f2f(y, v1, fd1, v2, fd2)`**Arguments**

<code>y</code>	a vector of values at which to calculate the density and distribution functions
<code>v1</code>	first parameter
<code>fd1</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>fd2</code>	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gamma_f2fa`*The second derivative of the density*

Description

The second derivative of the density

Usage`gamma_f2fa(x, v1, v2)`**Arguments**

<code>x</code>	a vector of training data values
<code>v1</code>	first parameter
<code>v2</code>	second parameter

Value

Matrix

gamma_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

gamma_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gamma_gg	<i>Second derivative matrix of the expected log-likelihood</i>
----------	--

Description

Second derivative matrix of the expected log-likelihood

Usage

```
gamma_gg(v1, fd1, v2, fd2)
```

Arguments

v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gamma_gmn	<i>One component of the second derivative of the expected log-likelihood</i>
-----------	--

Description

One component of the second derivative of the expected log-likelihood

Usage

```
gamma_gmn(alpha, v1, fd1, v2, fd2, mm, nn)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

<code>gamma_1dd</code>	<i>Second derivative matrix of the normalized log-likelihood</i>
------------------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

`gamma_1dd(x, v1, fd1, v2, fd2)`

Arguments

<code>x</code>	a vector of training data values
<code>v1</code>	first parameter
<code>fd1</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>fd2</code>	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

<code>gamma_1dda</code>	<i>The second derivative of the normalized log-likelihood</i>
-------------------------	---

Description

The second derivative of the normalized log-likelihood

Usage

`gamma_1dda(x, v1, v2)`

Arguments

<code>x</code>	a vector of training data values
<code>v1</code>	first parameter
<code>v2</code>	second parameter

Value

Matrix

gamma_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
------------	---

Description

Thirdderivative tensor of the normalized log-likelihood

Usage

gamma_lddd(x, v1, fd1, v2, fd2)

Arguments

- | | |
|-----|--|
| x | a vector of training data values |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Cubic scalar array

gamma_lddda	<i>The third derivative of the normalized log-likelihood</i>
-------------	--

Description

The third derivative of the normalized log-likelihood

Usage

gamma_lddda(x, v1, v2)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |

Value

3d array

gamma_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
-----------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

gamma_lmn(x, v1, fd1, v2, fd2, mm, nn)

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

gamma_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
-----------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

gamma_lmn(x, v1, fd1, v2, fd2, mm, nn, rr)

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gamma_logf	<i>Logf for RUST</i>
------------	----------------------

Description

Logf for RUST

Usage

gamma_logf(params, x)

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

gamma_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gamma_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

gamma_loglik	<i>log-likelihood function</i>
--------------	--------------------------------

Description

log-likelihood function

Usage

```
gamma_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

gamma_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
-----------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
gamma_logscores(logscores, x, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

- | | |
|-----------|---|
| logscores | logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime) |
| x | a vector of training data values |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| aderivs | logical for whether to use analytic derivatives (instead of numerical) |

Value

Two scalars

gamma_means	<i>MLE and RHP predictive means</i>
-------------	-------------------------------------

Description

MLE and RHP predictive means

Usage

```
gamma_means(means, ml_params, lddi, lddd, lambdad_cp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_cp	derivative of the log prior
nx	length of training data
dim	number of parameters

Value

Two scalars

gamma_mu1f	<i>DMGS equation 3.3, mu1 term</i>
------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

```
gamma_mu1f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gamma_mu2f	<i>DMGS equation 3.3, mu2 term</i>
------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

gamma_mu2f(alpha, v1, fd1, v2, fd2)

Arguments

- | | |
|-------|--|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

gamma_p1f	<i>DMGS equation 3.3, p1 term</i>
-----------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

gamma_p1f(y, v1, fd1, v2, fd2)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gamma_p2f	<i>DMGS equation 3.3, p2 term</i>
-----------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

gamma_p2f(y, v1, fd1, v2, fd2)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gamma_waic`*Waic*

Description

Waic

Usage

```
gamma_waic(waiccores, x, v1hat, fd1, v2hat, fd2, lddi, lddd, lambdad, aderivs)
```

Arguments

<code>waiccores</code>	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
<code>x</code>	a vector of training data values
<code>v1hat</code>	first parameter
<code>fd1</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>v2hat</code>	second parameter
<code>fd2</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>lddi</code>	inverse observed information matrix
<code>lddd</code>	third derivative of log-likelihood
<code>lambdad</code>	derivative of the log prior
<code>aderivs</code>	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`gev_checkmle`*Check MLE*

Description

Check MLE

Usage

```
gev_checkmle(ml_params, minxi, maxx)
```

Arguments

ml_params	parameters
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

gev_cp	<i>Generalized Extreme Value Distribution, Predictions Based on a Calibrating Prior</i>
--------	---

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0),
  d1 = 0.01,
```

```
    fd2 = 0.01,  
    d3 = 0.01,  
    fdalpha = 0.01,  
    minxi = -1,  
    maxxi = 999,  
    means = FALSE,  
    waicscores = FALSE,  
    extramodels = FALSE,  
    pdf = FALSE,  
    customprior = 0,  
    dmgs = TRUE,  
    rust = FALSE,  
    nrust = 1e+05,  
    pwm = FALSE,  
    debug = FALSE,  
    aderivs = TRUE  
)
```

```
rgev_cp(  
  n,  
  x,  
  ics = c(0, 0, 0),  
  d1 = 0.01,  
  fd2 = 0.01,  
  d3 = 0.01,  
  minxi = -0.45,  
  maxxi = 0.45,  
  extramodels = FALSE,  
  rust = FALSE,  
  mlcp = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
dgev_cp(  
  x,  
  y = x,  
  ics = c(0, 0, 0),  
  d1 = 0.01,  
  fd2 = 0.01,  
  d3 = 0.01,  
  minxi = -0.45,  
  maxxi = 0.45,  
  extramodels = FALSE,  
  rust = FALSE,  
  nrust = 1000,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```

)

pgev_cp(
  x,
  y = x,
  ics = c(0, 0, 0),
  d1 = 0.01,
  fd2 = 0.01,
  d3 = 0.01,
  minxi = -0.45,
  maxx = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tgev_cp(
  n,
  x,
  ics = c(0, 0, 0),
  d1 = 0.01,
  fd2 = 0.01,
  d3 = 0.01,
  extramodels = FALSE,
  debug = FALSE
)

```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>ics</code>	initial conditions for the maximum likelihood search
<code>d1</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the first parameter
<code>fd2</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the second parameter
<code>d3</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the third parameter
<code>fdalpha</code>	if <code>pdf=TRUE</code> , the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
<code>minxi</code>	the minimum allowed value of the shape parameter (decrease with caution)
<code>maxxi</code>	the maximum allowed value of the shape parameter (increase with caution)
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
customprior	a custom value for the slope of the log prior at the maxlik estimate
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
pwm	logical for whether to include PWM results (longer runtime)
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.

- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The GEV distribution has distribution function

$$F(x; \mu, \sigma, \xi) = \exp(-t(x; \mu, \sigma, \xi))$$

where

$$t(x; \mu, \sigma, \xi) = \begin{cases} [1 + \xi (\frac{x-\mu}{\sigma})]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp(-\frac{x-\mu}{\sigma}) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable and $\mu, \sigma > 0, \xi$ are the parameters.

The calibrating prior we use is given by

$$\pi(\mu, \sigma, \xi) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range $(\min x_i, \max x_i)$, since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q***** optionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_quantiles: predictive quantiles calculated from Bayesian integration with a flat prior.
- rh_ml_quantiles: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximum likelihood estimate for the shape parameter.
- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
shape=-0.4
x=fitdistcp::d110gev_example_data_v1
p=c(1:9)/10
q=qgev_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgev_cp)",
main="GEVD: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

gev_f1f

DMGS equation 3.3, f1 term

Description

DMGS equation 3.3, f1 term

Usage

```
gev_f1f(y, v1, d1, v2, fd2, v3, d3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gev_f1fa	<i>The first derivative of the density</i>
----------	--

Description

The first derivative of the density

Usage

```
gev_f1fa(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_f2f	<i>DMGS equation 3.3, f2 term</i>
---------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

```
gev_f2f(y, v1, d1, v2, fd2, v3, d3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

gev_f2fa

The second derivative of the density

Description

The second derivative of the density

Usage

```
gev_f2fa(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_fd

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_fdd(x, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Matrix

gev_ggd_mev	<i>Derivative of expected information matrix, based on MEV routine gev.infomat</i>
-------------	--

Description

Derivative of expected information matrix, based on MEV routine gev.infomat

Usage

gev_ggd_mev(v1, d1, v2, fd2, v3, d3)

Arguments

v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gev_ggid_mev	<i>Derivative of inverse expected information matrix, based on MEV routine gev.infomat</i>
--------------	--

Description

Derivative of inverse expected information matrix, based on MEV routine gev.infomat

Usage

gev_ggid_mev(v1, d1, v2, fd2, v3, d3)

Arguments

v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

 gev_k12_ppm_minusloglik

Temporary dummy for one of the ppm models

Description

Temporary dummy for one of the ppm models

Usage

```
gev_k12_ppm_minusloglik(x)
```

Arguments

x a vector of training data values

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.

- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

gev_k3_cp

Generalized Extreme Value Distribution with Known Shape, Predictions Based on a Calibrating Prior

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_k3_cp(  
  x,  
  p = seq(0.1, 0.9, 0.1),  
  d1 = 0.01,  
  fd2 = 0.01,  
  fdalpha = 0.01,  
  kshape = 0,  
  means = FALSE,  
  waicscores = FALSE,  
  pdf = FALSE,  
  dmgs = TRUE,  
  rust = FALSE,  
  nrust = 1e+05,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
rgev_k3_cp(  
  n,  
  x,  
  d1 = 0.01,  
  fd2 = 0.01,  
  kshape = 0,  
  rust = FALSE,  
  mlcp = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
dgev_k3_cp(  
  x,  
  y = x,  
  d1 = 0.01,  
  fd2 = 0.01,  
  kshape = 0,  
  rust = FALSE,  
  nrust = 1000,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
pgev_k3_cp(  
  x,  
  y = x,  
  d1 = 0.01,  
  fd2 = 0.01,  
  kshape = 0,
```

```

    rust = FALSE,
    nrust = 1000,
    debug = FALSE,
    aderivs = TRUE
  )

  tgev_k3_cp(n, x, d1 = 0.01, fd2 = 0.01, kshape = 0, debug = FALSE)

```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
kshape	the known shape parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.

- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The GEV distribution with known shape has distribution function

$$F(x; \mu, \sigma) = \exp(-t(x; \mu, \sigma))$$

where

$$t(x; \mu, \sigma) = \begin{cases} [1 + \xi (\frac{x-\mu}{\sigma})]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp(-\frac{x-\mu}{\sigma}) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu, \sigma > 0$ are the parameters and ξ is known (hence the k3 in the name).

The calibrating prior we use is given by

$$\pi(\mu, \sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

`q****` optionally returns the following, for EVT models only:

- `cp_pdf`: the density function at quantiles corresponding to input probabilities `p`. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),

- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
kshape=-0.4
x=fitdistcp::d53gev_k3_example_data_v1
p=c(1:9)/10
q=qgev_k3_cp(x,p,kshape=kshape,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgev_k3_cp)",
main="GEV: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
muhat=q$ml_params[1]
sghat=q$ml_params[2]
xi=kshape
qmax=ifelse(xi<0,muhat-sghat/xi,Inf)
cat(" ml_params=",q$ml_params,",")
cat(" qmax=",qmax,"\n")
```

gev_k3_f1f	<i>DMGS equation 3.3, f1 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

gev_k3_f1f(y, v1, d1, v2, fd2, kshape)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Matrix

gev_k3_f1fa	<i>The first derivative of the density</i>
-------------	--

Description

The first derivative of the density

Usage

gev_k3_f1fa(x, v1, v2, kshape)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

Vector

gev_k3_f2f	<i>DMGS equation 3.3, f2 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

gev_k3_f2f(y, v1, d1, v2, fd2, kshape)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

3d array

gev_k3_f2fa	<i>The second derivative of the density</i>
-------------	---

Description

The second derivative of the density

Usage

gev_k3_f2fa(x, v1, v2, kshape)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

Matrix

gev_k3_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_k3_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_k3_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

gev_k3_ldd(x, v1, d1, v2, fd2, kshape)

Arguments

- x a vector of training data values
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter
- kshape the known shape parameter

Value

Square scalar matrix

gev_k3_ldda	<i>The second derivative of the normalized log-likelihood</i>
-------------	---

Description

The second derivative of the normalized log-likelihood

Usage

gev_k3_ldda(x, v1, v2, kshape)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- kshape the known shape parameter

Value

Matrix

gev_k3_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
-------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

gev_k3_lddd(x, v1, d1, v2, fd2, kshape)

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Cubic scalar array

gev_k3_lddda	<i>The third derivative of the normalized log-likelihood</i>
--------------	--

Description

The third derivative of the normalized log-likelihood

Usage

gev_k3_lddda(x, v1, v2, kshape)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

3d array

gev_k3_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

gev_k3_lmn(x, v1, d1, v2, fd2, kshape, mm, nn)

Arguments

- x a vector of training data values
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter
- kshape the known shape parameter
- mm an index for which derivative to calculate
- nn an index for which derivative to calculate

Value

Scalar value

gev_k3_lmp	<i>One component of the third derivative of the normalized log-likelihood</i>
------------	---

Description

One component of the third derivative of the normalized log-likelihood

Usage

gev_k3_lmp(x, v1, d1, v2, fd2, kshape, mm, nn, rr)

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gev_k3_logf	<i>Logf for RUST</i>
-------------	----------------------

Description

Logf for RUST

Usage

gev_k3_logf(params, x, kshape)

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kshape	the known shape parameter

Value

Scalar value.

gev_k3_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_k3_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gev_k3_loglik	<i>log-likelihood function</i>
---------------	--------------------------------

Description

log-likelihood function

Usage

```
gev_k3_loglik(vv, x, kshape)
```

Arguments

vv	parameters
x	a vector of training data values
kshape	the known shape parameter

Value

Scalar value.

gev_k3_means	<i>MLE and RHP means</i>
--------------	--------------------------

Description

MLE and RHP means

Usage

```
gev_k3_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2, kshape)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kshape	the known shape parameter

Value

Two scalars

gev_k3_mu1f	<i>DMGS equation 3.3, mu1 term</i>
-------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

gev_k3_mu1f(alpha, v1, d1, v2, fd2, kshape)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Matrix

gev_k3_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
--------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

gev_k3_mu1fa(alpha, v1, v2, kshape)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

Vector

gev_k3_mu2f	<i>DMGS equation 3.3, mu2 term</i>
-------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

gev_k3_mu2f(alpha, v1, d1, v2, fd2, kshape)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

3d array

gev_k3_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
--------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

gev_k3_mu2fa(alpha, v1, v2, kshape)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

Matrix

gev_k3_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_pd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_k3_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_pdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_k3_waic	<i>Waic</i>
-------------	-------------

Description

Waic

Usage

```
gev_k3_waic(  
  waicscores,  
  x,  
  v1hat,  
  d1,  
  v2hat,  
  fd2,  
  kshape,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gev_ld12a	<i>The combined derivative of the normalized log-likelihood</i>
-----------	---

Description

The combined derivative of the normalized log-likelihood

Usage

gev_ld12a(x, v1, v2, v3)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

3d array

gev_lda	<i>The first derivative of the normalized log-likelihood</i>
---------	--

Description

The first derivative of the normalized log-likelihood

Usage

gev_lda(x, v1, v2, v3)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Vector

gev_1dd	<i>Second derivative matrix of the normalized log-likelihood</i>
---------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gev_1dd(x, v1, d1, v2, fd2, v3, d3)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gev_1dda	<i>The second derivative of the normalized log-likelihood</i>
----------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_1dda(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
----------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
gev_lddd(x, v1, d1, v2, fd2, v3, d3)
```

Arguments

- | | |
|-----|--|
| x | a vector of training data values |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v3 | third parameter |
| d3 | the delta used in the numerical derivatives with respect to the parameter |

Value

Cubic scalar array

gev_lddda	<i>The third derivative of the normalized log-likelihood</i>
-----------	--

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_lddda(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gev_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
---------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_lmn(x, v1, d1, v2, fd2, v3, d3, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

gev_lmp	<i>One component of the second derivative of the normalized log-likelihood</i>
---------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_lmp(x, v1, d1, v2, fd2, v3, d3, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gev_logf	<i>Logf for RUST</i>
----------	----------------------

Description

Logf for RUST

Usage

```
gev_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

gev_logfd	<i>First derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_logfd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_logfddd(x, v1, v2, v3)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gev_loglik	<i>log-likelihood function</i>
------------	--------------------------------

Description

log-likelihood function

Usage

gev_loglik(vv, x)

Arguments

- vv parameters
- x a vector of training data values

Value

Scalar value.

gev_means	<i>Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1</i>
-----------	---

Description

Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_means(  
  means,  
  ml_params,  
  lddi,  
  lddi_k3,  
  lddd,  
  lddd_k3,  
  lambdad_flat,  
  lambdad_rh_mle,  
  lambdad_rh_flat,  
  lambdad_jp,  
  lambdad_custom,  
  nx,  
  dim = 3  
)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddi_k3	inverse observed information matrix, fixed shape parameter
lddd	third derivative of log-likelihood
lddd_k3	third derivative of log-likelihood, fixed shape parameter
lambdad_flat	derivative of the log flat prior
lambdad_rh_mle	derivative of the log CRHP-MLE prior
lambdad_rh_flat	derivative of the log CRHP-FLAT prior
lambdad_jp	derivative of the log JP prior
lambdad_custom	custom value of the derivative of the log prior
nx	length of training data
dim	number of parameters

Value

Two scalars

gev_mu1f	<i>DMGS equation 3.3, mu1 term</i>
----------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

gev_mu1f(alpha, v1, d1, v2, fd2, v3, d3)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gev_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
-----------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

gev_mu1fa(alpha, v1, v2, v3)

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Vector

gev_mu2f	<i>DMGS equation 3.3, mu2 term</i>
----------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

gev_mu2f(alpha, v1, d1, v2, fd2, v3, d3)

Arguments

- | | |
|-------|--|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v3 | third parameter |
| d3 | the delta used in the numerical derivatives with respect to the parameter |

Value

3d array

gev_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
-----------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

gev_mu2fa(alpha, v1, v2, v3)

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Matrix

gev_p123_checkmle	<i>Check MLE</i>
-------------------	------------------

Description

Check MLE

Usage

gev_p123_checkmle(ml_params, minxi, maxx, t1, t2, t3)

Arguments

- | | |
|-----------|--------------------------------------|
| ml_params | parameters |
| minxi | minimum value of shape parameter xi |
| maxxi | maximum value of shape parameter xi |
| t1 | a vector of predictors for the mean |
| t2 | a vector of predictors for the sd |
| t3 | a vector of predictors for the shape |

Value

No return value (just a message to the screen).

gev_p123_cp

Generalized Extreme Value Distribution with Three Predictors, Predictions based on a Calibrating Prior

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  p = seq(0.1, 0.9, 0.1),
```

```

    ics = c(0, 0, 0, 0, 0, 0),
    d1 = 0.01,
    d2 = 0.01,
    d3 = 0.01,
    d4 = 0.01,
    d5 = 0.01,
    d6 = 0.01,
    fdalpha = 0.01,
    minxi = -0.45,
    maxx = 0.45,
    means = FALSE,
    waicscores = FALSE,
    extramodels = FALSE,
    pdf = FALSE,
    dmgs = TRUE,
    rust = FALSE,
    nrust = 1e+05,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

```

```

rgev_p123_cp(
  n,
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  ics = c(0, 0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  d6 = 0.01,
  minxi = -0.45,
  maxx = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE,

```

```
    aderivs = TRUE
  )

dgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  d6 = 0.01,
  minxi = -0.45,
  maxx = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 10,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

pgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
```

```

    d4 = 0.01,
    d5 = 0.01,
    d6 = 0.01,
    minxi = -0.45,
    maxxi = 0.45,
    extramodels = FALSE,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

tgev_p123_cp(
  n,
  x,
  t1,
  t2,
  t3,
  ics = c(0, 0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  d6 = 0.01,
  extramodels = FALSE,
  debug = FALSE
)

```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean, such that <code>length(t1)=length(x)</code>
t2	a vector of predictors for the sd, such that <code>length(t2)=length(x)</code>
t3	a vector of predictors for the shape, such that <code>length(t3)=length(x)</code>
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)
n02	an index for the predictor (specify either t02 or n02 but not both)
n03	an index for the predictor (specify either t03 or n03 but not both)
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
d1	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the first parameter

d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
d3	if aderivs=FALSE, the delta used for numerical derivatives with respect to the third parameter
d4	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fourth parameter
d5	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fifth parameter
d6	if aderivs=FALSE, the delta used for numerical derivatives with respect to the sixth parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.

- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The GEV distribution with three predictors has distribution function

$$F(x; a_1, b_1, a_2, b_2, a_3, b_3) = \exp(-t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi(a_3, b_3)))$$

where

$$t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi(a_3, b_3)) = \begin{cases} \left[1 + \xi(a_3, b_3) \left(\frac{x - \mu(a_1, b_1)}{\sigma(a_2, b_2)} \right) \right]^{-1/\xi(a_3, b_3)} & \text{if } \xi(a_3, b_3) \neq 0 \\ \exp\left(-\frac{x - \mu(a_1, b_1)}{\sigma(a_2, b_2)}\right) & \text{if } \xi(a_3, b_3) = 0 \end{cases}$$

where x is the random variable, $\mu = a_1 + b_1 t_1$ is the location parameter, modelled as a function of parameters a_1, b_1 and predictor t_1 , $\sigma = e^{a_2 + b_2 t_2}$ is the scale parameter, modelled as a function of parameters a_2, b_2 and predictor t_2 , and $\xi = a_3 + b_3 t_3$ is the shape parameter, modelled as a function of parameters a_3, b_3 and predictor t_3 .

The calibrating prior we use is given by

$$\pi(a_1, b_1, a_2, b_2, a_3, b_3) \propto 1$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi, maxx), since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d152gev_p123_example_data_v1_x
tt=fitdistcp::d152gev_p123_example_data_v1_t
t1=tt[,1]
t2=tt[,2]
t3=tt[,3]
p=c(1:9)/10
n01=10
n02=10
n03=10
q=qgev_p123_cp(x=x,t1=t1,t2=t2,t3=t3,n01=n01,n02=n02,n03=n03,t01=NA,t02=NA,t03=NA,
p=p,rust=FALSE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
```

```

plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgev_p123_cp)",
main="GEVD w/ p123: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
cat(" ml_params=",q$ml_params,"\n")

```

gev_p123_f1f	<i>DMGS equation 2.1, f1 term, fixed shape parameter DMGS equation 2.1, f1 term</i>
--------------	---

Description

DMGS equation 2.1, f1 term, fixed shape parameter DMGS equation 2.1, f1 term

Usage

```
gev_p123_f1f(y, t01, t02, t03, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, v6, d6)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gev_p123_f1fa	<i>The first derivative of the density</i>
---------------	--

Description

The first derivative of the density

Usage

gev_p123_f1fa(x, t01, t02, t03, v1, v2, v3, v4, v5, v6)

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p123_f2f	<i>GEVD-with-p1: DMGS equation 1.2 f2 term</i>
--------------	--

Description

GEVD-with-p1: DMGS equation 1.2 f2 term

Usage

gev_p123_f2f(y, t01, t02, t03, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, v6, d6)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

gev_p123_f2fa	<i>The second derivative of the density</i>
---------------	---

Description

The second derivative of the density

Usage

gev_p123_f2fa(x, t01, t02, t03, v1, v2, v3, v4, v5, v6)

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
v2	second parameter

v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_p123_fd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p123_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_fdd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
--------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gev_p123_ldd(x, t1, t2, t3, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, v6, d6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gev_p123_ldda	<i>The second derivative of the normalized log-likelihood</i>
---------------	---

Description

The second derivative of the normalized log-likelihood

Usage

gev_p123_ldda(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter

v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_ddd	<i>Third derivative tensor of the normalized log-likelihood, with fixed shape parameter</i>
--------------	---

Description

Third derivative tensor of the normalized log-likelihood, with fixed shape parameter

Usage

```
gev_p123_ddd(x, t1, t2, t3, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, v6, d6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

gev_p123_lddda	<i>The third derivative of the normalized log-likelihood</i>
----------------	--

Description

The third derivative of the normalized log-likelihood

Usage

gev_p123_lddda(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)

Arguments

- x a vector of training data values
- t1 a vector of predictors for the mean
- t2 a vector of predictors for the sd
- t3 a vector of predictors for the shape
- v1 first parameter
- v2 second parameter
- v3 third parameter
- v4 fourth parameter
- v5 fifth parameter
- v6 sixth parameter

Value

3d array

gev_p123_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p123_lmn(  
  x,  
  t1,  
  t2,  
  t3,  
  v1,  
  d1,  
  v2,  
  d2,  
  v3,  
  d3,  
  v4,  
  d4,  
  v5,  
  d5,  
  v6,  
  d6,  
  mm,  
  nn  
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

gev_p123_lmp	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p123_lmp(  
  x,  
  t1,  
  t2,  
  t3,  
  v1,  
  d1,  
  v2,  
  d2,  
  v3,  
  d3,  
  v4,  
  d4,  
  v5,  
  d5,  
  v6,  
  d6,  
  mm,  
  nn,  
  rr  
)
```

Arguments

- | | |
|----|---|
| x | a vector of training data values |
| t1 | a vector of predictors for the mean |
| t2 | a vector of predictors for the sd |
| t3 | a vector of predictors for the shape |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |

v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gev_p123_logf	<i>Logf for RUST</i>
---------------	----------------------

Description

Logf for RUST

Usage

gev_p123_logf(params, x, t1, t2, t3)

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape

Value

Scalar value.

gev_p123_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_p123_logfdd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_p123_logfddd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

3d array

gev_p123_loglik	<i>observed log-likelihood function</i>
-----------------	---

Description

observed log-likelihood function

Usage

```
gev_p123_loglik(vv, x, t1, t2, t3)
```

Arguments

vv	parameters
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape

Value

Scalar value.

gev_p123_means	<i>Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1</i>
----------------	---

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_p123_means(means, t01, t02, t03, ml_params, nx)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
ml_params	parameters
nx	length of training data

Value

Two scalars

gev_p123_mu1f	<i>GEVD-with-p1: DMGS equation 3.3 mul term</i>
---------------	---

Description

GEVD-with-p1: DMGS equation 3.3 mul term

Usage

```
gev_p123_mu1f(  
  alpha,  
  t01,  
  t02,  
  t03,  
  v1,  
  d1,  
  v2,
```

```
    d2,  
    v3,  
    d3,  
    v4,  
    d4,  
    v5,  
    d5,  
    v6,  
    d6  
  )
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gev_p123_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
----------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p123_mu1fa(alpha, t01, t02, t03, v1, v2, v3, v4, v5, v6)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p123_mu2f	<i>GEVD-with-p1: DMGS equation 3.3 mu2 term</i>
---------------	---

Description

GEVD-with-p1: DMGS equation 3.3 mu2 term

Usage

```
gev_p123_mu2f(  
  alpha,  
  t01,  
  t02,  
  t03,  
  v1,  
  d1,  
  v2,  
  d2,  
  v3,  
  d3,  
  v4,  
  d4,  
  v5,  
  d5,  
  v6,  
  d6  
)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

gev_p123_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
----------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

gev_p123_mu2fa(alpha, t01, t02, t03, v1, v2, v3, v4, v5, v6)

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
v2	second parameter

v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_p123_pd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p123_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_p123_pdd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)

Arguments

- | | |
|----|--------------------------------------|
| x | a vector of training data values |
| t1 | a vector of predictors for the mean |
| t2 | a vector of predictors for the sd |
| t3 | a vector of predictors for the shape |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |
| v4 | fourth parameter |
| v5 | fifth parameter |
| v6 | sixth parameter |

Value

Matrix

gev_p123_predictordata	<i>Predicted Parameter and Generalized Residuals</i>
------------------------	--

Description

Predicted Parameter and Generalized Residuals

Usage

gev_p123_predictordata(x, t1, t2, t3, t01, t02, t03, params)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
params	model parameters for calculating logf

Value

Two vectors

gev_p123_setics	<i>Set initial conditions</i>
-----------------	-------------------------------

Description

Set initial conditions

Usage

```
gev_p123_setics(x, t1, t2, t3, ics)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ics	initial conditions for the maximum likelihood search

Value

Vector

gev_p123_waic	<i>Waic</i>
---------------	-------------

Description

Waic

Usage

```
gev_p123_waic(  
  waicscores,  
  x,  
  t1,  
  t2,  
  t3,  
  v1h,  
  d1,  
  v2h,  
  d2,  
  v3h,  
  d3,  
  v4h,  
  d4,  
  v5h,  
  d5,  
  v6h,  
  d6,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1h	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2h	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

v3h	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4h	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5h	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6h	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gev_p12k3_f1f	<i>DMGS equation 2.1, f1 term, fixed shape parameter</i>
---------------	--

Description

DMGS equation 2.1, f1 term, fixed shape parameter

Usage

```
gev_p12k3_f1f(y, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

Matrix

gev_p12k3_f1fa	<i>The first derivative of the density</i>
----------------	--

Description

The first derivative of the density

Usage

gev_p12k3_f1fa(x, t, v1, v2, v3, v4, kshape)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter
- v4 fourth parameter
- kshape the known shape parameter

Value

Vector

gev_p12k3_f2f	<i>GEVD-with-p1: DMGS equation 1.2 f2 term</i>
---------------	--

Description

GEVD-with-p1: DMGS equation 1.2 f2 term

Usage

gev_p12k3_f2f(y, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

3d array

gev_p12k3_f2fa	<i>The second derivative of the density</i>
----------------	---

Description

The second derivative of the density

Usage

gev_p12k3_f2fa(x, t, v1, v2, v3, v4, kshape)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Matrix

gev_p12k3_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_fd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p12k3_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_fdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12k3_ldd	<i>Second derivative matrix of the normalized log-likelihood, with fixed shape parameter</i>
---------------	--

Description

Second derivative matrix of the normalized log-likelihood, with fixed shape parameter

Usage

gev_p12k3_ldd(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

Square scalar matrix

gev_p12k3_ldda	<i>The second derivative of the normalized log-likelihood</i>
----------------	---

Description

The second derivative of the normalized log-likelihood

Usage

gev_p12k3_ldda(x, t, v1, v2, v3, v4, kshape)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter
- v4 fourth parameter
- kshape the known shape parameter

Value

Matrix

gev_p12k3_lddd	<i>Third derivative tensor of the normalized log-likelihood, with fixed shape parameter</i>
----------------	---

Description

Third derivative tensor of the normalized log-likelihood, with fixed shape parameter

Usage

gev_p12k3_lddd(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

Cubic scalar array

gev_p12k3_lddda	<i>The third derivative of the normalized log-likelihood</i>
-----------------	--

Description

The third derivative of the normalized log-likelihood

Usage

gev_p12k3_lddda(x, t, v1, v2, v3, v4, kshape)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

3d array

gev_p12k3_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_logfdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12k3_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_logfddd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

3d array

gev_p12k3_mu1f	<i>GEVD-with-p1: DMGS equation 3.3 mu1 term, fixed shape parameter</i>
----------------	--

Description

GEVD-with-p1: DMGS equation 3.3 mu1 term, fixed shape parameter

Usage

gev_p12k3_mu1f(alpha, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5)

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

Matrix

gev_p12k3_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
-----------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

gev_p12k3_mu1fa(alpha, t, v1, v2, v3, v4, kshape)

Arguments

- | | |
|--------|---|
| alpha | a vector of values of alpha (one minus probability) |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |
| v4 | fourth parameter |
| kshape | the known shape parameter |

Value

Vector

gev_p12k3_mu2f	<i>GEVD-with-p1: DMGS equation 3.3 mu2 term, fixed shape parameter</i>
----------------	--

Description

GEVD-with-p1: DMGS equation 3.3 mu2 term, fixed shape parameter

Usage

gev_p12k3_mu2f(alpha, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5)

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

3d array

gev_p12k3_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
-----------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

gev_p12k3_mu2fa(alpha, t, v1, v2, v3, v4, kshape)

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Matrix

gev_p12k3_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_pd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p12k3_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_pdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_checkmle	<i>Check MLE</i>
------------------	------------------

Description

Check MLE

Usage

```
gev_p12_checkmle(ml_params, minxi, maxx)
```

Arguments

ml_params	parameters
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

gev_p12_cp

Generalized Extreme Value Distribution with Two Predictors, Predictions based on a Calibrating Prior

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
```

```

    fdalpha = 0.01,
    minxi = -0.45,
    maxxi = 0.45,
    means = FALSE,
    waicscores = FALSE,
    extramodels = FALSE,
    pdf = FALSE,
    dmgs = TRUE,
    rust = FALSE,
    nrust = 1e+05,
    predictordata = TRUE,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

```

```

rgev_p12_cp(
  n,
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  ics = c(0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

```

```

dgev_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,

```

```

    n02 = NA,
    y = x,
    ics = c(0, 0, 0, 0, 0),
    d1 = 0.01,
    d2 = 0.01,
    d3 = 0.01,
    d4 = 0.01,
    d5 = 0.01,
    minxi = -0.45,
    maxxi = 0.45,
    extramodels = FALSE,
    rust = FALSE,
    nrust = 10,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

```

```

pgev_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

```

```

tgev_p12_cp(
  n,
  x,
  t1,
  t2,

```

```

ics = c(0, 0, 0, 0, 0),
d1 = 0.01,
d2 = 0.01,
d3 = 0.01,
d4 = 0.01,
d5 = 0.01,
extramodels = FALSE,
debug = FALSE
)

```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean, such that <code>length(t1)=length(x)</code>
t2	a vector of predictors for the sd, such that <code>length(t2)=length(x)</code>
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)
n02	an index for the predictor (specify either t02 or n02 but not both)
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
d1	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the first parameter
d2	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the second parameter
d3	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the third parameter
d4	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the fourth parameter
d5	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the fifth parameter
fdalpha	if <code>pdf=TRUE</code> , the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)

pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer run-time)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The GEV distribution with two predictors has distribution function

$$F(x; a_1, b_1, a_2, b_2, \xi) = \exp(-t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi))$$

where

$$t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi) = \begin{cases} \left[1 + \xi \left(\frac{x - \mu(a_1, b_1)}{\sigma(a_2, b_2)}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x - \mu(a_1, b_1)}{\sigma(a_2, b_2)}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = a_1 + b_1 t_1$ is the location parameter, modelled as a function of parameters a_1, b_1 and predictor t_1 , $\sigma = e^{a_2 + b_2 t_2}$ is the scale parameter, modelled as a function of parameters a_2, b_2 and predictor t_2 , and ξ is the shape parameter.

The calibrating prior we use is given by

$$\pi(a_1, b_1, a_2, b_2, \xi) \propto 1$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range $(\min xi, \max xi)$, since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
# example 1
x=fitdistcp::d151gev_p12_example_data_v1_x
tt=fitdistcp::d151gev_p12_example_data_v1_t
t1=tt[,1]
t2=tt[,2]
p=c(1:9)/10
n01=10
n02=10
q=qgev_p12_cp(x=x,t1=t1,t2=t2,n01=n01,n02=n02,t01=NA,t02=NA,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgev_p12_cp)",
main="GEVD w/ p12: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

gev_p12_f1f

DMGS equation 2.1, f1 term

Description

DMGS equation 2.1, f1 term

Usage

```
gev_p12_f1f(y, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gev_p12_f1fa	<i>The first derivative of the density</i>
--------------	--

Description

The first derivative of the density

Usage

gev_p12_f1fa(x, t01, t02, v1, v2, v3, v4, v5)

Arguments

- | | |
|-----|--|
| x | a vector of training data values |
| t01 | a single value of the predictor (specify either t01 or n01 but not both) |
| t02 | a single value of the predictor (specify either t02 or n02 but not both) |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |
| v4 | fourth parameter |
| v5 | fifth parameter |

Value

Vector

gev_p12_f2f	<i>GEVD-with-p1: DMGS equation 1.2 f2 term</i>
-------------	--

Description

GEVD-with-p1: DMGS equation 1.2 f2 term

Usage

gev_p12_f2f(y, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

gev_p12_f2fa	<i>The second derivative of the density</i>
--------------	---

Description

The second derivative of the density

Usage

gev_p12_f2fa(x, t01, t02, v1, v2, v3, v4, v5)

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_fd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p12_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_fdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_ggd	<i>Derivative of information matrix, based on ldd</i>
-------------	---

Description

Derivative of information matrix, based on ldd

Usage

gev_p12_ggd(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`gev_p12_ldd`*Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gev_p12_ldd(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)
```

Arguments

<code>x</code>	a vector of training data values
<code>t1</code>	a vector of predictors for the mean
<code>t2</code>	a vector of predictors for the sd
<code>v1</code>	first parameter
<code>d1</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>d2</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v3</code>	third parameter
<code>d3</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v4</code>	fourth parameter
<code>d4</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v5</code>	fifth parameter
<code>d5</code>	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`gev_p12_ldda`*The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p12_ldda(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_lddd	<i>Third derivative tensor of the normalized log-likelihood, with fixed shape parameter</i>
--------------	---

Description

Third derivative tensor of the normalized log-likelihood, with fixed shape parameter

Usage

gev_p12_lddd(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

gev_p12_lddda	<i>The third derivative of the normalized log-likelihood</i>
---------------	--

Description

The third derivative of the normalized log-likelihood

Usage

gev_p12_lddda(x, t1, t2, v1, v2, v3, v4, v5)

Arguments

- x a vector of training data values
- t1 a vector of predictors for the mean
- t2 a vector of predictors for the sd
- v1 first parameter
- v2 second parameter
- v3 third parameter
- v4 fourth parameter
- v5 fifth parameter

Value

3d array

gev_p12_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
-------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

gev_p12_lmn(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, mm, nn)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

gev_p12_lmp	<i>One component of the second derivative of the normalized log-likelihood</i>
-------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

gev_p12_lmp(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, mm, nn, rr)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter

d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gev_p12_logf	<i>Logf for RUST</i>
--------------	----------------------

Description

Logf for RUST

Usage

gev_p12_logf(params, x, t1, t2)

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd

Value

Scalar value.

gev_p12_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_logfdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_logfddd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

3d array

gev_p12_loglik	<i>observed log-likelihood function</i>
----------------	---

Description

observed log-likelihood function

Usage

gev_p12_loglik(vv, x, t1, t2)

Arguments

vv	parameters
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd

Value

Scalar value.

gev_p12_means	<i>Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1</i>
---------------	---

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

gev_p12_means(means, t01, t02, ml_params, nx)

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
ml_params	parameters
nx	length of training data

Value

Two scalars

gev_p12_mu1f	<i>GEVD-with-p1: DMGS equation 3.3 mul term</i>
--------------	---

Description

GEVD-with-p1: DMGS equation 3.3 mul term

Usage

gev_p12_mu1f(alpha, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gev_p12_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
---------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p12_mu1fa(alpha, t01, t02, v1, v2, v3, v4, v5)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p12_mu2f	<i>GEVD-with-p1: DMGS equation 3.3 mu2 term</i>
--------------	---

Description

GEVD-with-p1: DMGS equation 3.3 mu2 term

Usage

gev_p12_mu2f(alpha, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| t01 | a single value of the predictor (specify either t01 or n01 but not both) |
| t02 | a single value of the predictor (specify either t02 or n02 but not both) |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| v3 | third parameter |
| d3 | the delta used in the numerical derivatives with respect to the parameter |
| v4 | fourth parameter |
| d4 | the delta used in the numerical derivatives with respect to the parameter |
| v5 | fifth parameter |
| d5 | the delta used in the numerical derivatives with respect to the parameter |

Value

3d array

gev_p12_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
---------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

gev_p12_mu2fa(alpha, t01, t02, v1, v2, v3, v4, v5)

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_pd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p12_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_p12_pdd(x, t1, t2, v1, v2, v3, v4, v5)

Arguments

- x a vector of training data values
- t1 a vector of predictors for the mean
- t2 a vector of predictors for the sd
- v1 first parameter
- v2 second parameter
- v3 third parameter
- v4 fourth parameter
- v5 fifth parameter

Value

Matrix

gev_p12_predictordata	<i>Predicted Parameter and Generalized Residuals</i>
-----------------------	--

Description

Predicted Parameter and Generalized Residuals

Usage

gev_p12_predictordata(predictordata, x, t1, t2, t01, t02, params)

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
params	model parameters for calculating logf

Value

Two vectors

gev_p12_setics	<i>Set initial conditions</i>
----------------	-------------------------------

Description

Set initial conditions

Usage

gev_p12_setics(x, t1, t2, ics)

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
ics	initial conditions for the maximum likelihood search

Value

Vector

gev_p12_waic	Waic
--------------	------

Description

Waic

Usage

```
gev_p12_waic(  
  waicscores,  
  x,  
  t1,  
  t2,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  v3hat,  
  d3,  
  v4hat,  
  d4,  
  v5hat,  
  d5,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4hat	fourth parameter

d4	the delta used in the numerical derivatives with respect to the parameter
v5hat	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gev_p1k3_cp	<i>GEV Distribution with Known Shape with a Predictor, Predictions Based on a Calibrating Prior</i>
-------------	---

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p1k3_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  p = seq(0.1, 0.9, 0.1),  
  d1 = 0.01,  
  d2 = 0.01,  
  fd3 = 0.01,  
  fdalpha = 0.01,  
  kshape = 0,  
  means = FALSE,  
  waicscores = FALSE,  
  pdf = FALSE,  
  dmgs = TRUE,  
  rust = FALSE,  
  nrust = 1e+05,  
  predictordata = TRUE,  
  centering = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
rgev_p1k3_cp(  
  n,  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  d1 = 0.01,  
  d2 = 0.01,  
  fd3 = 0.01,  
  kshape = 0,  
  rust = FALSE,  
  mlcp = TRUE,  
  centering = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
dgev_p1k3_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  y = x,  
  d1 = 0.01,
```

```

    d2 = 0.01,
    fd3 = 0.01,
    kshape = 0,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

```

```

pgev_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kshape = 0,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

```

```

tgev_p1k3_cp(
  n,
  x,
  t,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kshape = 0,
  debug = FALSE
)

```

Arguments

x	a vector of training data values
t	a vector of predictors, such that <code>length(t)=length(x)</code>
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter

d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
kshape	the known shape parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of x

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The GEV distribution with known shape with a predictor has distribution function

$$F(x; a, b, \sigma) = \exp(-t(x; \mu(a, b), \sigma))$$

where

$$t(x; a, b, \sigma) = \begin{cases} \left[1 + \xi \left(\frac{x - \mu(a, b)}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x - \mu(a, b)}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, $\sigma > 0$ is the shape parameter and ξ is known (hence the k3 in the name).

The calibrating prior we use is given by

$$\pi(\mu, \sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean (`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d150gev_p1_example_data_v1_x #use data for 150
tt=fitdistcp::d150gev_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qgev_p1k3_cp(x=x, t=tt, n0=n0, t0=NA, p=p, rust=TRUE, nrust=1000)
xmin=min(q$ml_quantiles, q$cp_quantiles);
xmax=max(q$ml_quantiles, q$cp_quantiles);
plot(q$ml_quantiles, p, xlab="quantile estimates", xlim=c(xmin, xmax),
sub="(from qgev_p1k3_cp)",
main="GEVD w/ p1: quantile estimates");
points(q$cp_quantiles, p, col="red", lwd=2)
points(q$ru_quantiles, p, col="blue", lwd=2)
cat(" ml_params=", q$ml_params, "\n")
```

gev_p1k3_f1f

DMGS equation 2.1, f1 term, fixed shape parameter

Description

DMGS equation 2.1, f1 term, fixed shape parameter

DMGS equation 2.1, f1 term

Usage

```
gev_p1k3_f1f(y, t0, v1, d1, v2, d2, v3, fd3, kshape)
```

```
gev_p1k3_f1f(y, t0, v1, d1, v2, d2, v3, fd3, kshape)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter

d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Matrix

gev_p1k3_f1fa	<i>The first derivative of the density</i>
---------------	--

Description

The first derivative of the density

Usage

gev_p1k3_f1fa(x, t, v1, v2, v3, kshape)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Vector

gev_p1k3_f2f	<i>GEVD-with-p1: DMGS equation 1.2 f2 term</i>
--------------	--

Description

GEVD-with-p1: DMGS equation 1.2 f2 term
DMGS equation 2.1, f2 term

Usage

gev_p1k3_f2f(y, t0, v1, d1, v2, d2, v3, fd3, kshape)

gev_p1k3_f2f(y, t0, v1, d1, v2, d2, v3, fd3, kshape)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- v3 third parameter
- fd3 the fractional delta used in the numerical derivatives with respect to the parameter
- kshape the known shape parameter

Value

3d array

gev_p1k3_f2fa	<i>The second derivative of the density</i>
---------------	---

Description

The second derivative of the density

Usage

gev_p1k3_f2fa(x, t, v1, v2, v3, kshape)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Matrix

gev_p1k3_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_p1k3_fd(x, t, v1, v2, v3, v4)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

gev_p1k3_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_fdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1k3_ldd	<i>Second derivative matrix of the normalized log-likelihood, with fixed shape parameter</i>
--------------	--

Description

Second derivative matrix of the normalized log-likelihood, with fixed shape parameter

Second derivative matrix of the normalized log-likelihood

Usage

```
gev_p1k3_ldd(x, t, v1, d1, v2, d2, v3, fd3, kshape)
```

```
gev_p1k3_ldd(x, t, v1, d1, v2, d2, v3, fd3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Square scalar matrix

gev_p1k3_ldda	<i>The second derivative of the normalized log-likelihood</i>
---------------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p1k3_ldda(x, t, v1, v2, v3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Matrix

gev_p1k3_lddd	<i>Third derivative tensor of the normalized log-likelihood, with fixed shape parameter</i>
---------------	---

Description

Third derivative tensor of the normalized log-likelihood, with fixed shape parameter
Third derivative tensor of the normalized log-likelihood

Usage

```
gev_p1k3_lddd(x, t, v1, d1, v2, d2, v3, fd3, kshape)  
  
gev_p1k3_lddd(x, t, v1, d1, v2, d2, v3, fd3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Cubic scalar array

gev_p1k3_lddda	<i>The third derivative of the normalized log-likelihood</i>
----------------	--

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p1k3_lddda(x, t, v1, v2, v3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

3d array

gev_p1k3_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p1k3_lmn(x, t, v1, d1, v2, d2, v3, fd3, kshape, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

gev_p1k3_lmp	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p1k3_lmp(x, t, v1, d1, v2, d2, v3, fd3, kshape, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gev_p1k3_logf	<i>Logf for RUST</i>
---------------	----------------------

Description

Logf for RUST

Usage

```
gev_p1k3_logf(params, x, t, kshape)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors
kshape	the known shape parameter

Value

Scalar value.

gev_p1k3_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_p1k3_logfdd(x, t, v1, v2, v3, v4)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1k3_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_logfddd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

gev_p1k3_loglik	<i>GEV-with-known-shape-with-p1 observed log-likelihood function</i>
-----------------	--

Description

GEV-with-known-shape-with-p1 observed log-likelihood function

Usage

```
gev_p1k3_loglik(vv, x, t, kshape)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors
kshape	the known shape parameter

Value

Scalar value.

gev_p1k3_means	<i>Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1</i>
----------------	---

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

gev_p1k3_means(means, t0, ml_params, kshape, nx)

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
kshape	the known shape parameter
nx	length of training data

Value

Two scalars

gev_p1k3_mu1f	<i>GEVD-with-p1: DMGS equation 3.3 mu1 term, fixed shape parameter</i>
---------------	--

Description

GEVD-with-p1: DMGS equation 3.3 mu1 term, fixed shape parameter
DMGS equation 3.3, mu1 term

Usage

gev_p1k3_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3, kshape)
gev_p1k3_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3, kshape)

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Matrix

gev_p1k3_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
----------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

gev_p1k3_mu1fa(alpha, t, v1, v2, v3, kshape)

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Vector

gev_p1k3_mu2f	<i>GEVD-with-p1: DMGS equation 3.3 mu2 term, fixed shape parameter</i>
---------------	--

Description

GEVD-with-p1: DMGS equation 3.3 mu2 term, fixed shape parameter
DMGS equation 3.3, mu2 term

Usage

gev_p1k3_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3, kshape)

gev_p1k3_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3, kshape)

Arguments

- alpha a vector of values of alpha (one minus probability)
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- v3 third parameter
- fd3 the fractional delta used in the numerical derivatives with respect to the parameter
- kshape the known shape parameter

Value

3d array

gev_p1k3_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
----------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

gev_p1k3_mu2fa(alpha, t, v1, v2, v3, kshape)

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Matrix

gev_p1k3_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_p1k3_pd(x, t, v1, v2, v3, v4)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

gev_p1k3_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_p1k3_pdd(x, t, v1, v2, v3, v4)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1k3_predictordata	<i>Predicted Parameter and Generalized Residuals</i>
------------------------	--

Description

Predicted Parameter and Generalized Residuals

Usage

gev_p1k3_predictordata(predictordata, x, t, t0, params, kshape)

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf
kshape	the known shape parameter

Value

Two vectors

gev_p1k3_waic	Waic
---------------	------

Description

Waic

Usage

```
gev_p1k3_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  v3hat,  
  fd3,  
  kshape,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
lddi	inverse observed information matrix

lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gev_p1_checkmle	<i>Check MLE</i>
-----------------	------------------

Description

Check MLE

Usage

```
gev_p1_checkmle(ml_params, minxi, maxx)
```

Arguments

ml_params	parameters
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

gev_p1_cp	<i>Generalized Extreme Value Distribution with a Predictor, Predictions Based on a Calibrating Prior</i>
-----------	--

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.

- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  d4 = 0.01,
  fdalpha = 0.01,
  minxi = -0.45,
  maxx = 0.45,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rgev_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  ics = c(0, 0, 0, 0),
```

```
    d1 = 0.01,  
    d2 = 0.01,  
    fd3 = 0.01,  
    d4 = 0.01,  
    minxi = -0.45,  
    maxxi = 0.45,  
    extramodels = FALSE,  
    rust = FALSE,  
    mlcp = TRUE,  
    debug = FALSE,  
    aderivs = TRUE  
  )
```

```
dgev_p1_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  y = x,  
  ics = c(0, 0, 0, 0),  
  d1 = 0.01,  
  d2 = 0.01,  
  fd3 = 0.01,  
  d4 = 0.01,  
  minxi = -0.45,  
  maxxi = 0.45,  
  extramodels = FALSE,  
  rust = FALSE,  
  nrust = 1000,  
  centering = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
pgev_p1_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  y = x,  
  ics = c(0, 0, 0, 0),  
  d1 = 0.01,  
  d2 = 0.01,  
  fd3 = 0.01,  
  d4 = 0.01,  
  minxi = -0.45,  
  maxxi = 0.45,  
  extramodels = FALSE,
```

```

    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  tgev_p1_cp(
    n,
    x,
    t,
    ics = c(0, 0, 0, 0),
    d1 = 0.01,
    d2 = 0.01,
    fd3 = 0.01,
    d4 = 0.01,
    extramodels = FALSE,
    debug = FALSE
  )

```

Arguments

<code>x</code>	a vector of training data values
<code>t</code>	a vector of predictors, such that <code>length(t)=length(x)</code>
<code>t0</code>	a single value of the predictor (specify either <code>t0</code> or <code>n0</code> but not both)
<code>n0</code>	an index for the predictor (specify either <code>t0</code> or <code>n0</code> but not both)
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>ics</code>	initial conditions for the maximum likelihood search
<code>d1</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the first parameter
<code>d2</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the second parameter
<code>fd3</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the third parameter
<code>d4</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the fourth parameter
<code>fdalpha</code>	if <code>pdf=TRUE</code> , the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
<code>minxi</code>	the minimum allowed value of the shape parameter (decrease with caution)
<code>maxxi</code>	the maximum allowed value of the shape parameter (increase with caution)
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>waicscores</code>	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)

extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with a predictor has distribution function

$$F(x; a, b, \sigma, \xi) = \exp(-t(x; \mu(a, b), \sigma, \xi))$$

where

$$t(x; \mu(a, b), \sigma, \xi) = \begin{cases} \left[1 + \xi \left(\frac{x - \mu(a, b)}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x - \mu(a, b)}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, modelled as a function of parameters a, b and predictor t , and $\sigma > 0, \xi$ are the scale and shape parameters.

The calibrating prior we use is given by

$$\pi(a, b, \sigma, \xi) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range $(\min x_i, \max x_i)$, since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q***** optionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_quantiles: predictive quantiles calculated from Bayesian integration with a flat prior.
- rh_ml_quantiles: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximum likelihood estimate for the shape parameter.
- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
# example 1
x=fitdistcp::d150gev_p1_example_data_v1_x
tt=fitdistcp::d150gev_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qgev_p1_cp(x=x,t=tt,n0=n0,t0=NA,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgev_p1_cp)",
main="GEVD w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

gev_p1_f1f	<i>DMGS equation 2.1, f1 term</i>
------------	-----------------------------------

Description

DMGS equation 2.1, f1 term

Usage

```
gev_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3, v4, d4)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gev_p1_f1fa	<i>The first derivative of the density</i>
-------------	--

Description

The first derivative of the density

Usage

gev_p1_f1fa(x, t, v1, v2, v3, v4)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter
- v4 fourth parameter

Value

Vector

gev_p1_f2f	<i>GEVD-with-p1: DMGS equation 1.2 f2 term</i>
------------	--

Description

GEVD-with-p1: DMGS equation 1.2 f2 term

Usage

gev_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3, v4, d4)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter

v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

gev_p1_f2fa	<i>The second derivative of the density</i>
-------------	---

Description

The second derivative of the density

Usage

gev_p1_f2fa(x, t, v1, v2, v3, v4)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1_fd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

gev_p1_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1_fdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1_ggd	<i>Derivative of information matrix, based on ldd</i>
------------	---

Description

Derivative of information matrix, based on ldd

Usage

gev_p1_ggd(x, t, v1, d1, v2, d2, v3, fd3, v4, d4)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`gev_p1_ldd`*Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gev_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3, v4, d4)
```

Arguments

<code>x</code>	a vector of training data values
<code>t</code>	a vector or matrix of predictors
<code>v1</code>	first parameter
<code>d1</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>d2</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v3</code>	third parameter
<code>fd3</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>v4</code>	fourth parameter
<code>d4</code>	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`gev_p1_ldda`*The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p1_ldda(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1_ddd	<i>Third derivative tensor of the normalized log-likelihood, with fixed shape parameter</i>
------------	---

Description

Third derivative tensor of the normalized log-likelihood, with fixed shape parameter

Usage

gev_p1_ddd(x, t, v1, d1, v2, d2, v3, fd3, v4, d4)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

gev_p1_lddda	<i>The third derivative of the normalized log-likelihood</i>
--------------	--

Description

The third derivative of the normalized log-likelihood

Usage

gev_p1_lddda(x, t, v1, v2, v3, v4)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter
- v4 fourth parameter

Value

3d array

gev_p1_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

gev_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, v4, d4, mm, nn)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

gev_p1_lmp	<i>One component of the second derivative of the normalized log-likelihood</i>
------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

gev_p1_lmp(x, t, v1, d1, v2, d2, v3, fd3, v4, d4, mm, nn, rr)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gev_p1_logf	<i>Logf for RUST</i>
-------------	----------------------

Description

Logf for RUST

Usage

```
gev_p1_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

gev_p1_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1_logfdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gev_p1_logfddd(x, t, v1, v2, v3, v4)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

gev_p1_loglik	<i>observed log-likelihood function</i>
---------------	---

Description

observed log-likelihood function

Usage

gev_p1_loglik(vv, x, t)

Arguments

- vv parameters
- x a vector of training data values
- t a vector or matrix of predictors

Value

Scalar value.

gev_p1_means	<i>Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1</i>
--------------	---

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_p1_means(  
  means,  
  t0,  
  ml_params,  
  lddi,  
  lddi_k4,  
  lddd,  
  lddd_k4,  
  lambdad_flat,  
  lambdad_rh_mle,  
  lambdad_rh_flat,  
  lambdad_jp,  
  nx,  
  dim = 4  
)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddi_k4	inverse observed information matrix, fixed shape parameter
lddd	third derivative of log-likelihood
lddd_k4	third derivative of log-likelihood, fixed shape parameter
lambdad_flat	derivative of the log flat prior
lambdad_rh_mle	derivative of the log CRHP-MLE prior
lambdad_rh_flat	derivative of the log CRHP-FLAT prior
lambdad_jp	derivative of the log JP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

gev_p1_mu1f	<i>GEVD-with-p1: DMGS equation 3.3 mu1 term</i>
-------------	---

Description

GEVD-with-p1: DMGS equation 3.3 mu1 term

Usage

gev_p1_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3, v4, d4)

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gev_p1_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
--------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

gev_p1_mu1fa(alpha, t, v1, v2, v3, v4)

Arguments

- alpha a vector of values of alpha (one minus probability)
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter
- v4 fourth parameter

Value

Vector

gev_p1_mu2f	<i>GEVD-with-p1: DMGS equation 3.3 mu2 term</i>
-------------	---

Description

GEVD-with-p1: DMGS equation 3.3 mu2 term

Usage

gev_p1_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3, v4, d4)

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

gev_p1_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
--------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

gev_p1_mu2fa(alpha, t, v1, v2, v3, v4)

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1_pd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

gev_p1_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1_pdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1_predictordata	<i>Predicted Parameter and Generalized Residuals</i>
----------------------	--

Description

Predicted Parameter and Generalized Residuals

Usage

```
gev_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

gev_p1_setics	<i>Set initial conditions</i>
---------------	-------------------------------

Description

Set initial conditions

Usage

```
gev_p1_setics(x, t, ics)
```

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- ics initial conditions for the maximum likelihood search

Value

Vector

gev_p1_waic	<i>Waic</i>
-------------	-------------

Description

Waic

Usage

```
gev_p1_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  v3hat,  
  fd3,  
  v4hat,  
  d4,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4hat	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gev_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_pd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_pdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_pwm_params	<i>PWM parameter estimation</i>
----------------	---------------------------------

Description

PWM parameter estimation

Usage

```
gev_pwm_params(x)
```

Arguments

x	a vector of training data values
---	----------------------------------

Value

Vector

gev_setics	<i>Set initial conditions</i>
------------	-------------------------------

Description

Set initial conditions

Usage

```
gev_setics(x, ics)
```

Arguments

- x a vector of training data values
- ics initial conditions for the maximum likelihood search

Value

Vector

gev_waic	<i>Waic</i>
----------	-------------

Description

Waic

Usage

```
gev_waic(  
  waicscores,  
  x,  
  v1hat,  
  d1,  
  v2hat,  
  fd2,  
  v3hat,  
  d3,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gnorm_k3_cp	<i>Generalized Normal Distribution Predictions Based on a Calibrating Prior</i>
-------------	---

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgnorm_k3_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kbeta = 4,
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rgnorm_k3_cp(
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  kbeta = 4,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
dgnorm_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kbeta = 4,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)
```

```

pgnorm_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kbeta = 4,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

```

```

tgnorm_k3_cp(n, x, d1 = 0.01, fd2 = 0.01, kbeta = 4, debug = FALSE)

```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kbeta	the known beta parameter
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times $-1/2$.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The generalized normal distribution has probability density function

$$f(x; \mu, \alpha) = \frac{\beta}{2\alpha\Gamma(1/\beta)} e^{-(|x-\mu|/\alpha)^\beta}$$

where x is the random variable, $\mu, \alpha > 0$ are the parameters and we consider β to be known (hence the k3 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\alpha) \propto \frac{1}{\alpha}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),

- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d32gnorm_k3_example_data_v1
p=c(1:9)/10
q=qgnorm_k3_cp(x,p,kbeta=4,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgnorm_k3_cp)",
main="gnorm: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

gnorm_k3_f1f

DMGS equation 3.3, f1 term

Description

DMGS equation 3.3, f1 term

Usage

```
gnorm_k3_f1f(y, v1, d1, v2, fd2, kbeta)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

Matrix

gnorm_k3_f1fa	<i>The first derivative of the density</i>
---------------	--

Description

The first derivative of the density

Usage

gnorm_k3_f1fa(x, v1, v2, kbeta)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kbeta	the known beta parameter

Value

Vector

gnorm_k3_f2f	<i>DMGS equation 3.3, f2 term</i>
--------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

gnorm_k3_f2f(y, v1, d1, v2, fd2, kbeta)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter
- kbeta the known beta parameter

Value

3d array

gnorm_k3_f2fa	<i>The second derivative of the density</i>
---------------	---

Description

The second derivative of the density

Usage

gnorm_k3_f2fa(x, v1, v2, kbeta)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- kbeta the known beta parameter

Value

Matrix

Matrix

gnorm_k3_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gnorm_k3_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gnorm_k3_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gnorm_k3_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gnorm_k3_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
--------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

gnorm_k3_ldd(x, v1, d1, v2, fd2, kbeta)

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

Square scalar matrix

gnorm_k3_ldda	<i>The second derivative of the normalized log-likelihood</i>
---------------	---

Description

The second derivative of the normalized log-likelihood

Usage

gnorm_k3_ldda(x, v1, v2, kbeta)

Arguments

- | | |
|-------|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |
| kbeta | the known beta parameter |

Value

Matrix

gnorm_k3_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
---------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

gnorm_k3_lddd(x, v1, d1, v2, fd2, kbeta)

Arguments

- | | |
|-------|--|
| x | a vector of training data values |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| kbeta | the known beta parameter |

Value

Cubic scalar array

<code>gnorm_k3_lddda</code>	<i>The third derivative of the normalized log-likelihood</i>
-----------------------------	--

Description

The third derivative of the normalized log-likelihood

Usage

`gnorm_k3_lddda(x, v1, v2, kbeta)`

Arguments

- `x` a vector of training data values
- `v1` first parameter
- `v2` second parameter
- `kbeta` the known beta parameter

Value

3d array

<code>gnorm_k3_lmn</code>	<i>One component of the second derivative of the normalized log-likelihood</i>
---------------------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

`gnorm_k3_lmn(x, v1, d1, v2, fd2, kbeta, mm, nn)`

Arguments

- `x` a vector of training data values
- `v1` first parameter
- `d1` the delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `fd2` the fractional delta used in the numerical derivatives with respect to the parameter
- `kbeta` the known beta parameter
- `mm` an index for which derivative to calculate
- `nn` an index for which derivative to calculate

Value

Scalar value

gnorm_k3_logf	<i>Logf for RUST</i>
---------------	----------------------

Description

Logf for RUST

Usage

gnorm_k3_logf(params, x, kbeta)

Arguments

- | | |
|--------|---------------------------------------|
| params | model parameters for calculating logf |
| x | a vector of training data values |
| kbeta | the known beta parameter |

Value

Scalar value.

gnorm_k3_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gnorm_k3_logfdd(x, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Matrix

gnorm_k3_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gnorm_k3_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gnorm_k3_loglik	<i>log-likelihood function</i>
-----------------	--------------------------------

Description

log-likelihood function

Usage

```
gnorm_k3_loglik(vv, x, kbeta)
```

Arguments

vv	parameters
x	a vector of training data values
kbeta	the known beta parameter

Value

Scalar value.

gnorm_k3_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
--------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

gnorm_k3_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, kbeta, aderivs)

Arguments

- | | |
|-----------|---|
| logscores | logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime) |
| x | a vector of training data values |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| kbeta | the known beta parameter |
| aderivs | logical for whether to use analytic derivatives (instead of numerical) |

Value

Two scalars

gnorm_k3_mu1f	<i>DMGS equation 3.3, mu1 term</i>
---------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

gnorm_k3_mu1f(alpha, v1, d1, v2, fd2, kbeta)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

Matrix

gnorm_k3_mu2f	<i>DMGS equation 3.3, mu2 term</i>
---------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

gnorm_k3_mu2f(alpha, v1, d1, v2, fd2, kbeta)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

3d array

gnorm_k3_p1f	<i>DMGS equation 3.3, p1 term</i>
--------------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

gnorm_k3_p1f(y, v1, d1, v2, fd2, kbeta)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

Matrix

gnorm_k3_p2f	<i>DMGS equation 3.3, p2 term</i>
--------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

gnorm_k3_p2f(y, v1, d1, v2, fd2, kbeta)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

3d array

<code>gnorm_lmp</code>	<i>One component of the second derivative of the normalized log-likelihood</i>
------------------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

`gnorm_lmp(x, v1, d1, v2, fd2, kbeta, mm, nn, rr)`

Arguments

<code>x</code>	a vector of training data values
<code>v1</code>	first parameter
<code>d1</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>fd2</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>kbeta</code>	the known beta parameter
<code>mm</code>	an index for which derivative to calculate
<code>nn</code>	an index for which derivative to calculate
<code>rr</code>	an index for which derivative to calculate

Value

Scalar value

gnorm_waic	<i>Waic for RUST</i>
------------	----------------------

Description

Waic for RUST

Usage

```
gnorm_waic(  
  waicscores,  
  x,  
  v1hat,  
  d1,  
  v2hat,  
  fd2,  
  kbeta,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gpd_k13_f1f	<i>DMGS equation 3.3, f1 term</i>
-------------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

gpd_k13_f1f(y, v1, fd1, v2, kloc)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k13_f1fa	<i>The first derivative of the density</i>
--------------	--

Description

The first derivative of the density

Usage

gpd_k13_f1fa(x, v1, v2, kloc)

Arguments

- | | |
|------|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |
| kloc | the known location parameter |

Value

Vector

gpd_k13_f2f	<i>DMGS equation 3.3, f2 term</i>
-------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

gpd_k13_f2f(y, v1, fd1, v2, kloc)

Arguments

- | | |
|------|--|
| y | a vector of values at which to calculate the density and distribution functions |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| kloc | the known location parameter |

Value

3d array

gpd_k13_f2fa	<i>The second derivative of the density</i>
--------------	---

Description

The second derivative of the density

Usage

gpd_k13_f2fa(x, v1, v2, kloc)

Arguments

- | | |
|------|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |
| kloc | the known location parameter |

Value

Matrix

gpd_k13_fd

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gpd_k13_fdd

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gpd_k13_l11	<i>One component of the second derivative of the normalized log-likelihood</i>
-------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gpd_k13_l11(x, v1, fd1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Scalar value

gpd_k13_l111	<i>One component of the third derivative of the normalized log-likelihood</i>
--------------	---

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
gpd_k13_l111(x, v1, fd1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Scalar value

<code>gpd_k13_ldd</code>	<i>Second derivative matrix of the normalized log-likelihood, with fixed shape</i>
--------------------------	--

Description

Second derivative matrix of the normalized log-likelihood, with fixed shape

Usage

`gpd_k13_ldd(x, v1, fd1, v2, kloc)`

Arguments

<code>x</code>	a vector of training data values
<code>v1</code>	first parameter
<code>fd1</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>kloc</code>	the known location parameter

Value

Square scalar matrix

<code>gpd_k13_ldda</code>	<i>The second derivative of the normalized log-likelihood</i>
---------------------------	---

Description

The second derivative of the normalized log-likelihood

Usage

`gpd_k13_ldda(x, v1, v2, kloc)`

Arguments

<code>x</code>	a vector of training data values
<code>v1</code>	first parameter
<code>v2</code>	second parameter
<code>kloc</code>	the known location parameter

Value

Matrix

gpd_k13_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
--------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

gpd_k13_lddd(x, v1, fd1, v2, kloc)

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Cubic scalar array

gpd_k13_lddda	<i>The third derivative of the normalized log-likelihood</i>
---------------	--

Description

The third derivative of the normalized log-likelihood

Usage

gpd_k13_lddda(x, v1, v2, kloc)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

3d array

gpd_k13_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gpd_k13_logfdd(x, v1, v2, v3)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Matrix

gpd_k13_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gpd_k13_logfddd(x, v1, v2, v3)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

3d array

gpd_k13_mu1f	<i>DMGS equation 3.3, mu1 term</i>
--------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

gpd_k13_mu1f(alpha, v1, fd1, v2, kloc)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k13_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
---------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

gpd_k13_mu1fa(alpha, v1, v2, kloc)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

<code>gpd_k13_mu2f</code>	<i>DMGS equation 3.3, mu2 term</i>
---------------------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

`gpd_k13_mu2f(alpha, v1, fd1, v2, kloc)`

Arguments

<code>alpha</code>	a vector of values of alpha (one minus probability)
<code>v1</code>	first parameter
<code>fd1</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>kloc</code>	the known location parameter

Value

3d array

<code>gpd_k13_mu2fa</code>	<i>Minus the second derivative of the cdf, at alpha</i>
----------------------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

`gpd_k13_mu2fa(alpha, v1, v2, kloc)`

Arguments

<code>alpha</code>	a vector of values of alpha (one minus probability)
<code>v1</code>	first parameter
<code>v2</code>	second parameter
<code>kloc</code>	the known location parameter

Value

Matrix

gpd_k13_p1f	<i>DMGS equation 3.3, p1 term</i>
-------------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

gpd_k13_p1f(y, v1, fd1, v2, kloc)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k13_p2f	<i>DMGS equation 3.3, p2 term</i>
-------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

gpd_k13_p2f(y, v1, fd1, v2, kloc)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

3d array

gpd_k13_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_pd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gpd_k13_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_pdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gpd_k1_checkmle	<i>Check MLE</i>
-----------------	------------------

Description

Check MLE

Usage

```
gpd_k1_checkmle(ml_params, kloc, minxi, maxx)
```

Arguments

ml_params	parameters
kloc	the known location parameter
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

gpd_k1_cp	<i>Generalized Pareto Distribution with Known Location Parameter, Predictions Based on a Calibrating Prior</i>
-----------	--

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`

- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgpd_k1_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kloc = 0,
  ics = c(0, 0),
  fd1 = 0.01,
  d2 = 0.01,
  fdalpha = 0.01,
  customprior = 0,
  minxi = -0.45,
  maxx = 2,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rgpd_k1_cp(
  n,
  x,
  kloc = 0,
  ics = c(0, 0),
  fd1 = 0.01,
  d2 = 0.01,
  minxi = -0.45,
  maxx = 2,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
)

dgpdk1_cp(
  x,
  y = x,
  kloc = 0,
  ics = c(0, 0),
  fd1 = 0.01,
  d2 = 0.01,
  customprior = 0,
  minxi = -0.45,
  maxx = 2,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pgpd_k1_cp(
  x,
  y = x,
  kloc = 0,
  ics = c(0, 0),
  fd1 = 0.01,
  d2 = 0.01,
  customprior = 0,
  minxi = -0.45,
  maxx = 2,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tgpd_k1_cp(
  n,
  x,
  kloc = 0,
  ics = c(0, 0),
  fd1 = 0.01,
  d2 = 0.01,
  extramodels = FALSE,
  debug = FALSE
)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kloc	the known location parameter
ics	initial conditions for the maximum likelihood search
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
customprior	a custom value for the slope of the log prior at the maxlik estimate
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Details

The GP distribution has exceedance distribution function

$$S(x; \mu, \sigma, \xi) = \begin{cases} [1 + \xi (\frac{x-\mu}{\sigma})]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp(-\frac{x-\mu}{\sigma}) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable and $\mu, \sigma > 0, \xi$ are the parameters.

The calibrating prior we use is given by

$$\pi(\mu, \sigma, \xi) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range $(\min x_i, \max x_i)$, since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of x

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.

- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

`q****` optionally returns the following, for EVT models only:

- `cp_pdf`: the density function at quantiles corresponding to input probabilities `p`. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

`q*****` optionally returns the following, for some EVT models only:

If `extramodels=TRUE`:

- `flat_quantiles`: predictive quantiles calculated from Bayesian integration with a flat prior.
- `rh_ml_quantiles`: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximum likelihood estimate for the shape parameter.
- `jp_quantiles`: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

`r****` additionally returns the following, for some EVT models only:

If `extramodels=TRUE`:

- `flat_deviates`: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- `rh_ml_deviates`: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- `jp_deviates`: predictive random deviates calculated using a Bayesian analysis with the JP.

`d****` additionally returns the following, for some EVT models only:

If `extramodels=TRUE`:

- `flat_pdf`: predictive density function from a Bayesian analysis with the flat prior.
- `rh_ml_pdf`: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- `jp_pdf`: predictive density function from a Bayesian analysis with the JP.

`p****` additionally returns the following, for some EVT models only:

If `extramodels=TRUE`:

- `flat_cdf`: predictive distribution function from a Bayesian analysis with the flat prior.
- `rh_ml_cdf`: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- `jp_cdf`: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d120gpd_k1_example_data_v1
p=c(1:9)/10
q=qgpd_k1_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgpd_k1_cp)",
main="GPD: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

gpd_k1_f1f	<i>DMGS equation 3.3, f1 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

```
gpd_k1_f1f(y, v1, fd1, v2, d2, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Matrix

gpd_k1_f1fa	<i>The first derivative of the density</i>
-------------	--

Description

The first derivative of the density

Usage

gpd_k1_f1fa(x, v1, v2, kloc)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

gpd_k1_f2f	<i>DMGS equation 3.3, f2 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

gpd_k1_f2f(y, v1, fd1, v2, d2, kloc)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

gpd_k1_f2fa	<i>The second derivative of the density</i>
-------------	---

Description

The second derivative of the density

Usage

```
gpd_k1_f2fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k1_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gpd_k1_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gpd_k1_ggd_mev	<i>Derivative of expected information matrix, based on MEV routine gpd.infomat</i>
----------------	--

Description

Derivative of expected information matrix, based on MEV routine gpd.infomat

Usage

```
gpd_k1_ggd_mev(v1, fd1, v2, d2, kloc)
```

Arguments

v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Cubic scalar array

<code>gpd_k1_ldd</code>	<i>Second derivative matrix of the normalized log-likelihood</i>
-------------------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

`gpd_k1_ldd(x, v1, fd1, v2, d2, kloc)`

Arguments

- | | |
|-------------------|--|
| <code>x</code> | a vector of training data values |
| <code>v1</code> | first parameter |
| <code>fd1</code> | the fractional delta used in the numerical derivatives with respect to the parameter |
| <code>v2</code> | second parameter |
| <code>d2</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>kloc</code> | the known location parameter |

Value

Square scalar matrix

<code>gpd_k1_ldda</code>	<i>The second derivative of the normalized log-likelihood</i>
--------------------------	---

Description

The second derivative of the normalized log-likelihood

Usage

`gpd_k1_ldda(x, v1, v2, kloc)`

Arguments

- | | |
|-------------------|----------------------------------|
| <code>x</code> | a vector of training data values |
| <code>v1</code> | first parameter |
| <code>v2</code> | second parameter |
| <code>kloc</code> | the known location parameter |

Value

Matrix

gpd_k1_ddd	<i>Third derivative tensor of the normalized log-likelihood</i>
------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

gpd_k1_ddd(x, v1, fd1, v2, d2, kloc)

Arguments

- | | |
|------|--|
| x | a vector of training data values |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| kloc | the known location parameter |

Value

Cubic scalar array

gpd_k1_ddd	<i>The third derivative of the normalized log-likelihood</i>
------------	--

Description

The third derivative of the normalized log-likelihood

Usage

gpd_k1_ddd(x, v1, v2, kloc)

Arguments

- | | |
|------|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |
| kloc | the known location parameter |

Value

3d array

<code>gpd_k1_lmn</code>	<i>One component of the second derivative of the normalized log-likelihood</i>
-------------------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

`gpd_k1_lmn(x, v1, fd1, v2, d2, kloc, mm, nn)`

Arguments

- `x` a vector of training data values
- `v1` first parameter
- `fd1` the fractional delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `d2` the delta used in the numerical derivatives with respect to the parameter
- `kloc` the known location parameter
- `mm` an index for which derivative to calculate
- `nn` an index for which derivative to calculate

Value

Scalar value

<code>gpd_k1_lmn</code>	<i>One component of the second derivative of the normalized log-likelihood</i>
-------------------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

`gpd_k1_lmn(x, v1, fd1, v2, d2, kloc, mm, nn, rr)`

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gpd_k1_logf

Logf for RUST

Description

Logf for RUST

Usage

```
gpd_k1_logf(params, x, kloc)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kloc	the known location parameter

Value

Scalar value.

gpd_k1_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gpd_k1_logfdd(x, v1, v2, v3)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Matrix

gpd_k1_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gpd_k1_logfddd(x, v1, v2, v3)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

3d array

gpd_k1_loglik	<i>log-likelihood function</i>
---------------	--------------------------------

Description

log-likelihood function

Usage

```
gpd_k1_loglik(vv, x, kloc)
```

Arguments

- | | |
|------|----------------------------------|
| vv | parameters |
| x | a vector of training data values |
| kloc | the known location parameter |

Value

Scalar value.

gpd_k1_means	<i>Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1</i>
--------------	---

Description

Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gpd_k1_means(  
  means,  
  ml_params,  
  lddi,  
  lddi_k2,  
  lddd,  
  lddd_k2,  
  lambdad_flat,  
  lambdad_rh_mle,  
  lambdad_rh_flat,  
  lambdad_jp,  
  nx,  
  dim = 2,  
  kloc = 0  
)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddi_k2	inverse observed information matrix, fixed shape parameter
lddd	third derivative of log-likelihood
lddd_k2	third derivative of log-likelihood, fixed shape parameter
lambdad_flat	derivative of the log flat prior
lambdad_rh_mle	derivative of the log CRHP-MLE prior
lambdad_rh_flat	derivative of the log CRHP-FLAT prior
lambdad_jp	derivative of the log JP prior
nx	length of training data
dim	number of parameters
kloc	the known location parameter

Value

Two scalars

<i>gpd_k1_mu1f</i>	<i>DMGS equation 3.3, mu1 term</i>
--------------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

`gpd_k1_mu1f(alpha, v1, fd1, v2, d2, kloc)`

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Matrix

gpd_k1_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
--------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

gpd_k1_mu1fa(alpha, v1, v2, kloc)

Arguments

- alpha a vector of values of alpha (one minus probability)
- v1 first parameter
- v2 second parameter
- kloc the known location parameter

Value

Vector

gpd_k1_mu2f	<i>DMGS equation 3.3, mu2 term</i>
-------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

gpd_k1_mu2f(alpha, v1, fd1, v2, d2, kloc)

Arguments

- alpha a vector of values of alpha (one minus probability)
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- kloc the known location parameter

Value

3d array

<code>gpd_k1_mu2fa</code>	<i>Minus the second derivative of the cdf, at alpha</i>
---------------------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

`gpd_k1_mu2fa(alpha, v1, v2, kloc)`

Arguments

- `alpha` a vector of values of alpha (one minus probability)
- `v1` first parameter
- `v2` second parameter
- `kloc` the known location parameter

Value

Matrix

<code>gpd_k1_p1f</code>	<i>DMGS equation 3.3, p1 term</i>
-------------------------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

`gpd_k1_p1f(y, v1, fd1, v2, d2, kloc)`

Arguments

- `y` a vector of values at which to calculate the density and distribution functions
- `v1` first parameter
- `fd1` the fractional delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `d2` the delta used in the numerical derivatives with respect to the parameter
- `kloc` the known location parameter

Value

Matrix

gpd_k1_p2f	<i>DMGS equation 3.3, p2 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

gpd_k1_p2f(y, v1, fd1, v2, d2, kloc)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

gpd_k1_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gpd_k1_pd(x, v1, v2, v3)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gpd_k1_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_pdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gpd_k1_setics	<i>Set initial conditions</i>
---------------	-------------------------------

Description

Set initial conditions

Usage

```
gpd_k1_setics(x, ics)
```

Arguments

x	a vector of training data values
ics	initial conditions for the maximum likelihood search

Value

Vector

gpd_k1_waic	<i>Waic</i>
-------------	-------------

Description

Waic

Usage

```
gpd_k1_waic(  
  waicscores,  
  x,  
  v1hat,  
  fd1,  
  v2hat,  
  d2,  
  kloc,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgumbel_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rgumbel_cp(
```

```

    n,
    x,
    d1 = 0.01,
    fd2 = 0.01,
    rust = FALSE,
    mlcp = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

dgumbel_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pgumbel_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tgumbel_cp(n, x, d1 = 0.01, fd2 = 0.01, debug = FALSE)

```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>d1</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the first parameter
<code>fd2</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the second parameter
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>waicscores</code>	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
<code>logscores</code>	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)

dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of x

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Gumbel distribution has distribution function

$$F(x; \mu, \sigma) = \exp \left(- \exp \left(- \frac{x - \mu}{\sigma} \right) \right)$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible

- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d50gumbel_example_data_v1
p=c(1:9)/10
q=qgumbel_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgumbel_cp)",
main="Gumbel: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

<code>gumbel_f1f</code>	<i>DMGS equation 3.3, f1 term</i>
-------------------------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

`gumbel_f1f(y, v1, d1, v2, fd2)`

Arguments

- | | |
|------------------|--|
| <code>y</code> | a vector of values at which to calculate the density and distribution functions |
| <code>v1</code> | first parameter |
| <code>d1</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>v2</code> | second parameter |
| <code>fd2</code> | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Matrix

<code>gumbel_f1fa</code>	<i>The first derivative of the density</i>
--------------------------	--

Description

The first derivative of the density

Usage

`gumbel_f1fa(x, v1, v2)`

Arguments

- | | |
|-----------------|----------------------------------|
| <code>x</code> | a vector of training data values |
| <code>v1</code> | first parameter |
| <code>v2</code> | second parameter |

Value

Vector

<code>gumbel_f2f</code>	<i>DMGS equation 3.3, f2 term</i>
-------------------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

`gumbel_f2f(y, v1, d1, v2, fd2)`

Arguments

- `y` a vector of values at which to calculate the density and distribution functions
- `v1` first parameter
- `d1` the delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `fd2` the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

<code>gumbel_f2fa</code>	<i>The second derivative of the density</i>
--------------------------	---

Description

The second derivative of the density

Usage

`gumbel_f2fa(x, v1, v2)`

Arguments

- `x` a vector of training data values
- `v1` first parameter
- `v2` second parameter

Value

Matrix

gumbel_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

gumbel_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gumbel_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gumbel_ldd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gumbel_ldda	<i>The second derivative of the normalized log-likelihood</i>
-------------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
gumbel_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

<code>gumbel_1ddd</code>	<i>Third derivative tensor of the normalized log-likelihood</i>
--------------------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

`gumbel_1ddd(x, v1, d1, v2, fd2)`

Arguments

- `x` a vector of training data values
- `v1` first parameter
- `d1` the delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `fd2` the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

<code>gumbel_1ddda</code>	<i>The third derivative of the normalized log-likelihood</i>
---------------------------	--

Description

The third derivative of the normalized log-likelihood

Usage

`gumbel_1ddda(x, v1, v2)`

Arguments

- `x` a vector of training data values
- `v1` first parameter
- `v2` second parameter

Value

3d array

<code>gumbel_lmn</code>	<i>One component of the second derivative of the normalized log-likelihood</i>
-------------------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

`gumbel_lmn(x, v1, d1, v2, fd2, mm, nn)`

Arguments

<code>x</code>	a vector of training data values
<code>v1</code>	first parameter
<code>d1</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>fd2</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>mm</code>	an index for which derivative to calculate
<code>nn</code>	an index for which derivative to calculate

Value

Scalar value

<code>gumbel_lmn</code>	<i>One component of the third derivative of the normalized log-likelihood</i>
-------------------------	---

Description

One component of the third derivative of the normalized log-likelihood

Usage

`gumbel_lmn(x, v1, d1, v2, fd2, mm, nn, rr)`

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

<i>gumbel_logf</i>	<i>Logf for RUST</i>
--------------------	----------------------

Description

Logf for RUST

Usage

`gumbel_logf(params, x)`

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

gumbel_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gumbel_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

<code>gumbel_loglik</code>	<i>log-likelihood function</i>
----------------------------	--------------------------------

Description

log-likelihood function

Usage

`gumbel_loglik(vv, x)`

Arguments

<code>vv</code>	parameters
<code>x</code>	a vector of training data values

Value

Scalar value.

<code>gumbel_logscores</code>	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
-------------------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

`gumbel_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)`

Arguments

<code>logscores</code>	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
<code>x</code>	a vector of training data values
<code>d1</code>	the delta used in the numerical derivatives with respect to the parameter
<code>fd2</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>aderivs</code>	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

<code>gumbel_means</code>	<i>MLE and RHP predictive means</i>
---------------------------	-------------------------------------

Description

MLE and RHP predictive means

Usage

```
gumbel_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

<code>means</code>	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
<code>ml_params</code>	parameters
<code>lddi</code>	inverse observed information matrix
<code>lddd</code>	third derivative of log-likelihood
<code>lambdad_rhp</code>	derivative of the log RHP prior
<code>nx</code>	length of training data
<code>dim</code>	number of parameters

Value

Two scalars

<code>gumbel_mu1f</code>	<i>DMGS equation 3.3, mu1 term</i>
--------------------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

```
gumbel_mu1f(alpha, v1, d1, v2, fd2)
```

Arguments

<code>alpha</code>	a vector of values of alpha (one minus probability)
<code>v1</code>	first parameter
<code>d1</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>fd2</code>	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

<code>gumbel_mu1fa</code>	<i>Minus the first derivative of the cdf, at alpha</i>
---------------------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

`gumbel_mu1fa(alpha, v1, v2)`

Arguments

- | | |
|--------------------|---|
| <code>alpha</code> | a vector of values of alpha (one minus probability) |
| <code>v1</code> | first parameter |
| <code>v2</code> | second parameter |

Value

Vector

<code>gumbel_mu2f</code>	<i>DMGS equation 3.3, mu2 term</i>
--------------------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

`gumbel_mu2f(alpha, v1, d1, v2, fd2)`

Arguments

- | | |
|--------------------|--|
| <code>alpha</code> | a vector of values of alpha (one minus probability) |
| <code>v1</code> | first parameter |
| <code>d1</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>v2</code> | second parameter |
| <code>fd2</code> | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

<code>gumbel_mu2fa</code>	<i>Minus the second derivative of the cdf, at alpha</i>
---------------------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

`gumbel_mu2fa(alpha, v1, v2)`

Arguments

- | | |
|--------------------|---|
| <code>alpha</code> | a vector of values of alpha (one minus probability) |
| <code>v1</code> | first parameter |
| <code>v2</code> | second parameter |

Value

Matrix

<code>gumbel_p1f</code>	<i>DMGS equation 3.3, p1 term</i>
-------------------------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

`gumbel_p1f(y, v1, d1, v2, fd2)`

Arguments

- | | |
|------------------|--|
| <code>y</code> | a vector of values at which to calculate the density and distribution functions |
| <code>v1</code> | first parameter |
| <code>d1</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>v2</code> | second parameter |
| <code>fd2</code> | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Matrix

<code>gumbel_p1fa</code>	<i>The first derivative of the cdf</i>
--------------------------	--

Description

The first derivative of the cdf

Usage

`gumbel_p1fa(x, v1, v2)`

Arguments

- | | |
|-----------------|----------------------------------|
| <code>x</code> | a vector of training data values |
| <code>v1</code> | first parameter |
| <code>v2</code> | second parameter |

Value

Vector

<code>gumbel_p1_cp</code>	<i>Gumbel Distribution with a Predictor, Predictions Based on a Calibrating Prior</i>
---------------------------	---

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgumbel_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rgumbel_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
dgumbel_p1_cp(
  x,
  t,
```

```

    t0 = NA,
    n0 = NA,
    y = x,
    d1 = 0.01,
    d2 = 0.01,
    fd3 = 0.01,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

```

```

pgumbel_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

```

```

tgumbel_p1_cp(n, x, t, d1 = 0.01, d2 = 0.01, fd3 = 0.01, debug = FALSE)

```

Arguments

x	a vector of training data values
t	a vector of predictors, such that <code>length(t)=length(x)</code>
t0	a single value of the predictor (specify either <code>t0</code> or <code>n0</code> but not both)
n0	an index for the predictor (specify either <code>t0</code> or <code>n0</code> but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	the fractional delta used in the numerical derivatives with respect to the location parameter
d2	the fractional delta used in the numerical derivatives with respect to the slope parameter
fd3	the fractional delta used in the numerical derivatives with respect to the scale parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Gumbel distribution with a predictor has distribution function

$$F(x; a, b, \sigma) = \exp \left(- \exp \left(- \frac{x - \mu(a, b)}{\sigma} \right) \right)$$

where x is the random variable, $\mu = a + bt$ is the shape parameter as a function of parameters a, b and predictor t , and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d70gumbel_p1_example_data_v1_x
tt=fitdistcp::d70gumbel_p1_example_data_v1_t
p=c(1:9)/10
n0=10
```

```

q=qgumbel_p1_cp(x, tt, n0=n0, p=p, rust=TRUE, nrust=1000)
xmin=min(q$ml_quantiles, q$cp_quantiles);
xmax=max(q$ml_quantiles, q$cp_quantiles);
plot(q$ml_quantiles, p, xlab="quantile estimates", xlim=c(xmin, xmax),
sub="(from qgumbel_p1_cp)",
main="Gumbel w/ p1: quantile estimates");
points(q$cp_quantiles, p, col="red", lwd=2)
points(q$ru_quantiles, p, col="blue")

```

gumbel_p1_f1f

*DMGS equation 2.1, f1 term***Description**

DMGS equation 2.1, f1 term

Usage

gumbel_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

<code>gumbel_p1_f1fa</code>	<i>The first derivative of the density</i>
-----------------------------	--

Description

The first derivative of the density

Usage

`gumbel_p1_f1fa(x, t, v1, v2, v3)`

Arguments

- | | |
|-----------------|----------------------------------|
| <code>x</code> | a vector of training data values |
| <code>t</code> | a vector or matrix of predictors |
| <code>v1</code> | first parameter |
| <code>v2</code> | second parameter |
| <code>v3</code> | third parameter |

Value

Vector

<code>gumbel_p1_f2f</code>	<i>DMGS equation 2.1, f2 term</i>
----------------------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

`gumbel_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3)`

Arguments

- | | |
|------------------|--|
| <code>y</code> | a vector of values at which to calculate the density and distribution functions |
| <code>t0</code> | a single value of the predictor (specify either <code>t0</code> or <code>n0</code> but not both) |
| <code>v1</code> | first parameter |
| <code>d1</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>v2</code> | second parameter |
| <code>d2</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>v3</code> | third parameter |
| <code>fd3</code> | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

gumbel_p1_f2fa	<i>The second derivative of the density</i>
----------------	---

Description

The second derivative of the density

Usage

gumbel_p1_f2fa(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gumbel_p1_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

gumbel_p1_fd(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

<code>gumbel_p1_fdd</code>	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

`gumbel_p1_fdd(x, t, v1, v2, v3)`

Arguments

- `x` a vector of training data values
- `t` a vector or matrix of predictors
- `v1` first parameter
- `v2` second parameter
- `v3` third parameter

Value

Matrix

<code>gumbel_p1_ldd</code>	<i>Second derivative matrix of the normalized log-likelihood</i>
----------------------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

`gumbel_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3)`

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gumbel_p1_ldda	<i>The second derivative of the normalized log-likelihood</i>
----------------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
gumbel_p1_ldda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

<code>gumbel_p1_lddd</code>	<i>Third derivative tensor of the normalized log-likelihood</i>
-----------------------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

`gumbel_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3)`

Arguments

- `x` a vector of training data values
- `t` a vector or matrix of predictors
- `v1` first parameter
- `d1` the delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `d2` the delta used in the numerical derivatives with respect to the parameter
- `v3` third parameter
- `fd3` the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

<code>gumbel_p1_lddda</code>	<i>The third derivative of the normalized log-likelihood</i>
------------------------------	--

Description

The third derivative of the normalized log-likelihood

Usage

`gumbel_p1_lddda(x, t, v1, v2, v3)`

Arguments

- `x` a vector of training data values
- `t` a vector or matrix of predictors
- `v1` first parameter
- `v2` second parameter
- `v3` third parameter

Value

3d array

<code>gumbel_p1_lmn</code>	<i>One component of the second derivative of the normalized log-likelihood</i>
----------------------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gumbel_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn)
```

Arguments

<code>x</code>	a vector of training data values
<code>t</code>	a vector or matrix of predictors
<code>v1</code>	first parameter
<code>d1</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>d2</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v3</code>	third parameter
<code>fd3</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>mm</code>	an index for which derivative to calculate
<code>nn</code>	an index for which derivative to calculate

Value

Scalar value

<code>gumbel_p1_lmp</code>	<i>One component of the second derivative of the normalized log-likelihood</i>
----------------------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

`gumbel_p1_lmp(x, t, v1, d1, v2, d2, v3, fd3, mm, nn, rr)`

Arguments

<code>x</code>	a vector of training data values
<code>t</code>	a vector or matrix of predictors
<code>v1</code>	first parameter
<code>d1</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>d2</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v3</code>	third parameter
<code>fd3</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>mm</code>	an index for which derivative to calculate
<code>nn</code>	an index for which derivative to calculate
<code>rr</code>	an index for which derivative to calculate

Value

Scalar value

<code>gumbel_p1_logf</code>	<i>Logf for RUST</i>
-----------------------------	----------------------

Description

Logf for RUST

Usage

`gumbel_p1_logf(params, x, t)`

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

gumbel_p1_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

<code>gumbel_p1_logfddd</code>	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

`gumbel_p1_logfddd(x, t, v1, v2, v3)`

Arguments

- `x` a vector of training data values
- `t` a vector or matrix of predictors
- `v1` first parameter
- `v2` second parameter
- `v3` third parameter

Value

3d array

<code>gumbel_p1_loglik</code>	<i>observed log-likelihood function</i>
-------------------------------	---

Description

observed log-likelihood function

Usage

`gumbel_p1_loglik(vv, x, t)`

Arguments

- `vv` parameters
- `x` a vector of training data values
- `t` a vector or matrix of predictors

Value

Scalar value.

gumbel_p1_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
---------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
gumbel_p1_logscores(logscores, x, t, d1, d2, fd3, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

gumbel_p1_means	<i>Gumbel distribution: RHP mean</i>
-----------------	--------------------------------------

Description

Gumbel distribution: RHP mean

Usage

```
gumbel_p1_means(means, t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

<code>gumbel_p1_mu1f</code>	<i>DMGS equation 3.3, mu1 term</i>
-----------------------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

`gumbel_p1_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3)`

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gumbel_p1_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
-----------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gumbel_p1_mu1fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gumbel_p1_mu2f	<i>DMGS equation 3.3, mu2 term</i>
----------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

```
gumbel_p1_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

<code>gumbel_p1_mu2fa</code>	<i>Minus the second derivative of the cdf, at alpha</i>
------------------------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

`gumbel_p1_mu2fa(alpha, t, v1, v2, v3)`

Arguments

- | | |
|--------------------|---|
| <code>alpha</code> | a vector of values of alpha (one minus probability) |
| <code>t</code> | a vector or matrix of predictors |
| <code>v1</code> | first parameter |
| <code>v2</code> | second parameter |
| <code>v3</code> | third parameter |

Value

Matrix

<code>gumbel_p1_p1f</code>	<i>DMGS equation 2.1, p1 term</i>
----------------------------	-----------------------------------

Description

DMGS equation 2.1, p1 term

Usage

`gumbel_p1_p1f(y, t0, v1, d1, v2, d2, v3, fd3)`

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

<i>gumbel_p1_p1fa</i>	<i>The first derivative of the cdf</i>
-----------------------	--

Description

The first derivative of the cdf

Usage

`gumbel_p1_p1fa(x, t, v1, v2, v3)`

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

<code>gumbel_p1_p2f</code>	<i>DMGS equation 2.1, p2 term</i>
----------------------------	-----------------------------------

Description

DMGS equation 2.1, p2 term

Usage

`gumbel_p1_p2f(y, t0, v1, d1, v2, d2, v3, fd3)`

Arguments

- | | |
|------------------|--|
| <code>y</code> | a vector of values at which to calculate the density and distribution functions |
| <code>t0</code> | a single value of the predictor (specify either <code>t0</code> or <code>n0</code> but not both) |
| <code>v1</code> | first parameter |
| <code>d1</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>v2</code> | second parameter |
| <code>d2</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>v3</code> | third parameter |
| <code>fd3</code> | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

<code>gumbel_p1_p2fa</code>	<i>The second derivative of the cdf</i>
-----------------------------	---

Description

The second derivative of the cdf

Usage

`gumbel_p1_p2fa(x, t, v1, v2, v3)`

Arguments

- | | |
|-----------------|----------------------------------|
| <code>x</code> | a vector of training data values |
| <code>t</code> | a vector or matrix of predictors |
| <code>v1</code> | first parameter |
| <code>v2</code> | second parameter |
| <code>v3</code> | third parameter |

Value

Matrix

gumbel_p1_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_pd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gumbel_p1_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_pdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gumbel_p1_predictordata
<i>Predicted Parameter and Generalized Residuals</i>

Description

Predicted Parameter and Generalized Residuals

Usage

gumbel_p1_predictordata(predictordata, x, t, t0, params)

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

gumbel_p1_waic	<i>Waic</i>
----------------	-------------

Description

Waic

Usage

```
gumbel_p1_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  v3hat,  
  fd3,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gumbel_p2f	<i>DMGS equation 3.3, p2 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

gumbel_p2f(y, v1, d1, v2, fd2)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

gumbel_p2fa	<i>The second derivative of the cdf</i>
-------------	---

Description

The second derivative of the cdf

Usage

gumbel_p2fa(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Matrix

gumbel_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

gumbel_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gumbel_waic	<i>Waic</i>
-------------	-------------

Description

Waic

Usage

gumbel_waic(waiccores, x, v1hat, d1, v2hat, fd2, lddi, lddd, lambdad, aderivs)

Arguments

- | | |
|-----------|--|
| waiccores | logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime) |
| x | a vector of training data values |
| v1hat | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2hat | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| lddi | inverse observed information matrix |
| lddd | third derivative of log-likelihood |
| lambdad | derivative of the log prior |
| aderivs | logical for whether to use analytic derivatives (instead of numerical) |

Value

Two numeric values.

halfnorm_cp	<i>Half-Normal Distribution Predictions Based on a Calibrating Prior</i>
-------------	--

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qhalfnorm_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rhalfnorm_cp(
  n,
  x,
  fd1 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
dhalfnorm_cp(
  x,
  y = x,
  fd1 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
```

```

    aderivs = TRUE
  )

  phalfnorm_cp(
    x,
    y = x,
    fd1 = 0.01,
    rust = FALSE,
    nrust = 1000,
    debug = FALSE,
    aderivs = TRUE
  )

  thalfnorm_cp(n, x, fd1 = 0.01, debug = FALSE)

```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>fd1</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the first parameter
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>waicscores</code>	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
<code>logscores</code>	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
<code>dmgs</code>	logical that indicates whether DMGS calculations should be run or not (longer run time)
<code>rust</code>	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
<code>nrust</code>	the number of posterior samples used in the RUST calculations
<code>debug</code>	logical for turning on debug messages
<code>aderivs</code>	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
<code>n</code>	the number of random samples required
<code>mlcp</code>	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
<code>y</code>	a vector of values at which to calculate the density and distribution functions

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.

- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The half-normal distribution has probability density function

$$f(x; \theta) = \frac{2\theta}{\pi} e^{-\theta^2 x^2 / \pi}$$

where $x \geq 0$ is the random variable and $\theta > 0$ is the parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\theta) \propto \frac{1}{\theta}$$

as given in Jewson et al. (2025). Some other authors may parametrize the half-normal differently.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),

- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d20halfnorm_example_data_v1
p=c(1:9)/10
q=qhalfnorm_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles)
xmax=max(q$ml_quantiles,q$cp_quantiles)
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qhalfnorm_cp)",
main="Halfnorm: quantile estimates")
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

halfnorm_f1f	<i>DMGS equation 2.1, f1 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, f1 term

Usage

```
halfnorm_f1f(y, v1, fd1)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

halfnorm_f1fa	<i>The first derivative of the density</i>
---------------	--

Description

The first derivative of the density

Usage

halfnorm_f1fa(x, v1)

Arguments

- x a vector of training data values
- v1 first parameter

Value

Vector

halfnorm_f2f	<i>DMGS equation 2.1, f2 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

halfnorm_f2f(y, v1, fd1)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

halfnorm_f2fa	<i>The second derivative of the density</i>
---------------	---

Description

The second derivative of the density

Usage

```
halfnorm_f2fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

halfnorm_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
halfnorm_fd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

halfnorm_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
halfnorm_fdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

halfnorm_gg	<i>Expected information matrix</i>
-------------	------------------------------------

Description

Expected information matrix

Usage

```
halfnorm_gg(v1, fd1)
```

Arguments

v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

halfnorm_gg11	<i>Second derivative of the expected log-likelihood</i>
---------------	---

Description

Second derivative of the expected log-likelihood

Usage

halfnorm_gg11(alpha, v1, fd1)

Arguments

- | | |
|-------|--|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Scalar value

halfnorm_l111	<i>Third derivative of the normalized log-likelihood</i>
---------------	--

Description

Third derivative of the normalized log-likelihood

Usage

halfnorm_l111(x, v1, fd1)

Arguments

- | | |
|-----|--|
| x | a vector of training data values |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Scalar value

`halfnorm_ldd`*The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
halfnorm_ldd(x, v1, fd1)
```

Arguments

<code>x</code>	a vector of training data values
<code>v1</code>	first parameter
<code>fd1</code>	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`halfnorm_ldda`*The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
halfnorm_ldda(x, v1)
```

Arguments

<code>x</code>	a vector of training data values
<code>v1</code>	first parameter

Value

Matrix

halfnorm_lddd	<i>Third derivative tensor of the log-likelihood</i>
---------------	--

Description

Third derivative tensor of the log-likelihood

Usage

halfnorm_lddd(x, v1, fd1)

Arguments

- x a vector of training data values
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

halfnorm_lddda	<i>The third derivative of the normalized log-likelihood</i>
----------------	--

Description

The third derivative of the normalized log-likelihood

Usage

halfnorm_lddda(x, v1)

Arguments

- x a vector of training data values
- v1 first parameter

Value

3d array

halfnorm_logf	<i>Logf for RUST</i>
---------------	----------------------

Description

Logf for RUST

Usage

halfnorm_logf(params, x)

Arguments

- | | |
|--------|---------------------------------------|
| params | model parameters for calculating logf |
| x | a vector of training data values |

Value

Scalar value.

halfnorm_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

halfnorm_logfdd(x, v1)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |

Value

Matrix

halfnorm_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
halfnorm_logfddd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

3d array

halfnorm_loglik	<i>Log-likelihood function</i>
-----------------	--------------------------------

Description

Log-likelihood function

Usage

```
halfnorm_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

halfnorm_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
--------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
halfnorm_logscores(logscores, x, fd1 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

halfnorm_means	<i>MLE and RHP predictive means RHP mean based on the expectation of DMGS equation 2.1</i>
----------------	--

Description

MLE and RHP predictive means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
halfnorm_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 1)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

halfnorm_mu1f	<i>DMGS equation 3.3, mu1 term</i>
---------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

halfnorm_mu1f(alpha, v1, fd1)

Arguments

- | | |
|-------|--|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Matrix

halfnorm_mu2f	<i>DMGS equation 3.3, mu2 term</i>
---------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

halfnorm_mu2f(alpha, v1, fd1)

Arguments

- | | |
|-------|--|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

halfnorm_p1f	<i>DMGS equation 2.1, p1 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, p1 term

Usage

halfnorm_p1f(y, v1, fd1)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Matrix

halfnorm_p2f	<i>DMGS equation 2.1, p2 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, p2 term

Usage

halfnorm_p2f(y, v1, fd1)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

halfnorm_waic	<i>Waic</i>
---------------	-------------

Description

Waic

Usage

```
halfnorm_waic(waiccores, x, v1hat, fd1, lddi, lddd, lambdad, aderivs)
```

Arguments

waiccores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

invgamma_cp	<i>Inverse Gamma Distribution, Predictions Based on a Calibrating Prior</i>
-------------	---

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.

- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qinvgamma_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  prior = "type 1",
  debug = FALSE,
  aderivs = TRUE
)
```

```
rinvgamma_cp(
  n,
  x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
dinvgamma_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
```

```

    debug = FALSE,
    aderivs = TRUE
  )

  pinvgamma_cp(
    x,
    y = x,
    fd1 = 0.01,
    fd2 = 0.01,
    rust = FALSE,
    nrust = 1000,
    debug = FALSE,
    aderivs = TRUE
  )

  tinvgamma_cp(n, x, fd1 = 0.01, fd2 = 0.01, debug = FALSE)

```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
prior	logical indicating which prior to use
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times $-1/2$.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Inverse Gamma distribution has probability density function

$$f(x; \alpha, \sigma) = \frac{1}{x\Gamma(\alpha)} \left(\frac{\sigma}{x}\right)^\alpha e^{-\sigma/x}$$

where $x \geq 0$ is the random variable and $\alpha > 0, \sigma > 0$ are the parameters.

The calibrating prior we use is

$$\pi(\alpha, \sigma) \propto \frac{1}{\alpha\sigma}$$

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),

- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d101invgamma_example_data_v1
p=c(1:9)/10
q=qinvgamma_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qinvgamma_cp)",
main="Invgamma: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

invgamma_f1f

DMGS equation 3.3, f1 term

Description

DMGS equation 3.3, f1 term

Usage

```
invgamma_f1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

invgamma_f1fa	<i>The first derivative of the density</i>
---------------	--

Description

The first derivative of the density

Usage

```
invgamma_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

invgamma_f2f	<i>DMGS equation 3.3, f2 term</i>
--------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

invgamma_f2f(y, v1, fd1, v2, fd2)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

invgamma_f2fa	<i>The second derivative of the density</i>
---------------	---

Description

The second derivative of the density

Usage

invgamma_f2fa(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Matrix

invgamma_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

invgamma_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgamma_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
--------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
invgamma_ldd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

invgamma_ldda	<i>The second derivative of the normalized log-likelihood</i>
---------------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
invgamma_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgamma_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
---------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
invgamma_lddd(x, v1, fd1, v2, fd2)
```

Arguments

- | | |
|-----|--|
| x | a vector of training data values |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Cubic scalar array

invgamma_lddda	<i>The third derivative of the normalized log-likelihood</i>
----------------	--

Description

The third derivative of the normalized log-likelihood

Usage

```
invgamma_lddda(x, v1, v2)
```

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |

Value

3d array

invgamma_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

invgamma_lmn(x, v1, fd1, v2, fd2, mm, nn)

Arguments

- x a vector of training data values
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter
- mm an index for which derivative to calculate
- nn an index for which derivative to calculate

Value

Scalar value

invgamma_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

invgamma_lmn(x, v1, fd1, v2, fd2, mm, nn, rr)

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

invgamma_logf	<i>Logf for RUST</i>
---------------	----------------------

Description

Logf for RUST

Usage

invgamma_logf(params, x)

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

invgamma_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgamma_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

invgamma_loglik	<i>log-likelihood function</i>
-----------------	--------------------------------

Description

log-likelihood function

Usage

```
invgamma_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

invgamma_logscores	<i>Log scores for MLE and cp predictions calculated using leave-one-out</i>
--------------------	---

Description

Log scores for MLE and cp predictions calculated using leave-one-out

Usage

```
invgamma_logscores(logscores, x, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

invgamma_mu1f	DMGS equation 3.3, mu1 term
---------------	-----------------------------

Description

DMGS equation 3.3, mu1 term

Usage

```
invgamma_mu1f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

invgamma_mu2f	DMGS equation 3.3, mu2 term
---------------	-----------------------------

Description

DMGS equation 3.3, mu2 term

Usage

```
invgamma_mu2f(alpha, v1, fd1, v2, fd2)
```

Arguments

- | | |
|-------|--|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

invgamma_p1f	<i>DMGS equation 3.3, p1 term</i>
--------------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

invgamma_p1f(y, v1, fd1, v2, fd2)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Matrix

invgamma_p2f	<i>DMGS equation 3.3, p2 term</i>
--------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

invgamma_p2f(y, v1, fd1, v2, fd2)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

invgamma_waic	<i>Waic</i>
---------------	-------------

Description

Waic

Usage

```
invgamma_waic(  
  waicscores,  
  x,  
  v1hat,  
  fd1,  
  v2hat,  
  fd2,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter

fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

invgauss_cp	<i>Inverse Gauss Distribution, Predictions Based on a Calibrating Prior</i>
-------------	---

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qinvgauss_cp(  
  x,  
  p = seq(0.1, 0.9, 0.1),  
  fd1 = 0.01,  
  fd2 = 0.01,  
  means = FALSE,  
  waicscores = FALSE,  
  logscores = FALSE,  
  dmgs = TRUE,  
  rust = FALSE,  
  nrust = 1e+05,  
  prior = "type 1",  
  debug = FALSE,  
  aderivs = TRUE  
)  
  
rinvgauss_cp(  
  n,  
  x,  
  fd1 = 0.01,  
  fd2 = 0.01,  
  rust = FALSE,  
  prior = "type 1",  
  mlcp = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)  
  
dinvgauss_cp(  
  x,  
  y = x,  
  fd1 = 0.01,  
  fd2 = 0.01,  
  rust = FALSE,  
  nrust = 1000,  
  prior = "type 1",  
  debug = FALSE,  
  aderivs = TRUE  
)  
  
pinvgauss_cp(  
  x,  
  y = x,  
  fd1 = 0.01,  
  fd2 = 0.01,  
  rust = FALSE,  
  nrust = 1000,
```

```

    prior = "type 1",
    debug = FALSE,
    aderivs = TRUE
  )

  tinvgauss_cp(n, x, fd1 = 0.01, fd2 = 0.01, prior = "type 1", debug = FALSE)

```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>fd1</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the first parameter
<code>fd2</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the second parameter
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>waicscores</code>	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
<code>logscores</code>	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
<code>dmgs</code>	logical that indicates whether DMGS calculations should be run or not (longer run time)
<code>rust</code>	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
<code>nrust</code>	the number of posterior samples used in the RUST calculations
<code>prior</code>	logical indicating which prior to use
<code>debug</code>	logical for turning on debug messages
<code>aderivs</code>	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
<code>n</code>	the number of random samples required
<code>mlcp</code>	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
<code>y</code>	a vector of values at which to calculate the density and distribution functions

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.

- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times -1/2.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Inverse Gaussian distribution has probability density function

$$f(x; \mu, \phi) = \left(\frac{1}{2\pi\phi x^3} \right)^{1/2} \exp \left(-\frac{(x - \mu)^2}{2\mu^2\phi x} \right)$$

where $x \geq 0$ is the random variable and $\mu > 0, \phi > 0$ are the parameters.

The calibrating prior we use by default is

$$\pi(\alpha, \sigma) \propto \frac{1}{\phi}$$

The prior

$$\pi(\alpha, \sigma) \propto \frac{1}{\mu\phi}$$

is also available as an option with `prior="type 2"`.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean (`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
debug=FALSE
# example 1 can go wrong for small sample sizes, so I've increased to 50
#
# example 1
if(debug)cat("example 1\n")
x=fitdistcp::d102invgauss_example_data_v1
if(debug)cat("x=",x,"\n")
p=c(1:9)/10
q=qinvgauss_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qinvgauss_cp)",
main="Invgauss: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

invgauss_f1f	<i>DMGS equation 3.3, f1 term</i>
--------------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

```
invgauss_f1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

invgauss_f1fa	<i>The first derivative of the density</i>
---------------	--

Description

The first derivative of the density

Usage

invgauss_f1fa(x, v1, v2)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

invgauss_f2f	<i>DMGS equation 3.3, f2 term</i>
--------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

invgauss_f2f(y, v1, fd1, v2, fd2)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

invgauss_f2fa	<i>The second derivative of the density</i>
---------------	---

Description

The second derivative of the density

Usage

invgauss_f2fa(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Matrix

invgauss_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

invgauss_fd(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Vector

invgauss_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgauss_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgauss_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
--------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
invgauss_ldd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

invgauss_ldda	<i>The second derivative of the normalized log-likelihood</i>
---------------	---

Description

The second derivative of the normalized log-likelihood

Usage

invgauss_ldda(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Matrix

invgauss_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
---------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

invgauss_lddd(x, v1, fd1, v2, fd2)

Arguments

- x a vector of training data values
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

invgauss_lddda	<i>The third derivative of the normalized log-likelihood</i>
----------------	--

Description

The third derivative of the normalized log-likelihood

Usage

```
invgauss_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

invgauss_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
invgauss_lmn(x, v1, fd1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

invgauss_lmnp	<i>One component of the second derivative of the normalized log-likelihood</i>
---------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
invgauss_lmnp(x, v1, fd1, v2, fd2, mm, nn, rr)
```

Arguments

- | | |
|-----|--|
| x | a vector of training data values |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| mm | an index for which derivative to calculate |
| nn | an index for which derivative to calculate |
| rr | an index for which derivative to calculate |

Value

Scalar value

invgauss_logf	<i>Logf for RUST</i>
---------------	----------------------

Description

Logf for RUST

Usage

```
invgauss_logf(params, x, prior)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
prior	logical indicating which prior to use

Value

Scalar value.

invgauss_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgauss_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgauss_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgauss_logfddd(x, v1, v2)
```

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

3d array

invgauss_loglik	<i>log-likelihood function</i>
-----------------	--------------------------------

Description

log-likelihood function

Usage

invgauss_loglik(vv, x)

Arguments

- vv parameters
- x a vector of training data values

Value

Scalar value.

invgauss_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
--------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

invgauss_logscores(logscores, x, prior, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
prior	logical indicating which prior to use
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

invgauss_means	<i>MLE and RHP predictive means</i>
----------------	-------------------------------------

Description

MLE and RHP predictive means

Usage

```
invgauss_means(means, ml_params, lddi, lddd, lambdad_cp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_cp	derivative of the log prior
nx	length of training data
dim	number of parameters

Value

Two scalars

invgauss_mu1f	DMGS equation 3.3, mu1 term
---------------	-----------------------------

Description

DMGS equation 3.3, mu1 term

Usage

```
invgauss_mu1f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

invgauss_mu2f	DMGS equation 3.3, mu2 term
---------------	-----------------------------

Description

DMGS equation 3.3, mu2 term

Usage

```
invgauss_mu2f(alpha, v1, fd1, v2, fd2)
```

Arguments

- | | |
|-------|--|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

invgauss_p1f	<i>DMGS equation 3.3, p1 term</i>
--------------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

invgauss_p1f(y, v1, fd1, v2, fd2)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Matrix

invgauss_p2f	<i>DMGS equation 3.3, p2 term</i>
--------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

invgauss_p2f(y, v1, fd1, v2, fd2)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

invgauss_waic	<i>Waic</i>
---------------	-------------

Description

Waic

Usage

```
invgauss_waic(  
  waicscores,  
  x,  
  v1hat,  
  fd1,  
  v2hat,  
  fd2,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter

fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

jpf2p	<i>Jeffreys' Prior with two parameters</i>
-------	--

Description

Jeffreys' Prior with two parameters

Usage

jpf2p(ggd, detg, ggi)

Arguments

ggd	gradient of the expected information matrix
detg	determinant of the expected information matrix
ggi	inverse of the expected information matrix

Value

Vector of 2 values

jpf3p	<i>Jeffreys' Prior with three parameters</i>
-------	--

Description

Jeffreys' Prior with three parameters

Usage

jpf3p(ggd, detg, ggi)

Arguments

ggd	gradient of the expected information matrix
detg	determinant of the expected information matrix
ggi	inverse of the expected information matrix

Value

Vector of 3 values

jpf4p	<i>Jeffreys' Prior with four parameters</i>
-------	---

Description

Jeffreys' Prior with four parameters

Usage

jpf4p(ggd, detg, ggi)

Arguments

ggd	gradient of the expected information matrix
detg	determinant of the expected information matrix
ggi	inverse of the expected information matrix

Value

Vector of 4 values

lnorm_cp	<i>Log-normal Distribution Predictions Based on a Calibrating Prior</i>
----------	---

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data *x*. For model **** the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlnorm_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rlnorm_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE, aderivs = TRUE)

dlnorm_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)

plnorm_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)

tlnorm_cp(n, x, debug = FALSE)
```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>d1</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the first parameter
<code>fd2</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the second parameter

means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times $-1/2$.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.

- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The log normal distribution has probability density function

$$f(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma x} e^{-(\log(x)-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

p**** optionally returns the following:

If rust=TRUE:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d35lnorm_example_data_v1
p=c(1:9)/10
q=qlnorm_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlnorm_cp)",
main="Log-normal: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

lnorm_dmgs_cp

*Log-normal Distribution Predictions Based on a Calibrating Prior, using DMGS (for testing only)***Description**

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlnorm_dmgs_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rlnorm_dmgs_cp(
  n,
```

```

    x,
    d1 = 0.01,
    fd2 = 0.01,
    mlcp = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

dlnorm_dmgs_cp(x, y = x, d1 = 0.01, fd2 = 0.01, debug = FALSE, aderivs = TRUE)

plnorm_dmgs_cp(x, y, d1 = 0.01, fd2 = 0.01, debug = FALSE, aderivs = TRUE)

```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.

- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of x

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The log normal distribution has probability density function

$$f(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma x} e^{-(\log(x)-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean (`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d35lnorm_example_data_v1
p=c(1:9)/10
q=qlnorm_dmgs_cp(x,p)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlnorm_dmgs_cp)",
main="Log-normal_DMGS: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
```

lnorm_dmgs_gg11	<i>One component of the second derivative of the expected log-likelihood</i>
-----------------	--

Description

One component of the second derivative of the expected log-likelihood

Usage

```
lnorm_dmgs_gg11(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Scalar value

<code>lnorm_dmgs_gg12</code>	<i>One component of the second derivative of the expected log-likelihood</i>
------------------------------	--

Description

One component of the second derivative of the expected log-likelihood

Usage

`lnorm_dmgs_gg12(alpha, v1, d1, v2, fd2)`

Arguments

- | | |
|--------------------|--|
| <code>alpha</code> | a vector of values of alpha (one minus probability) |
| <code>v1</code> | first parameter |
| <code>d1</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>v2</code> | second parameter |
| <code>fd2</code> | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Scalar value

<code>lnorm_dmgs_gg22</code>	<i>One component of the second derivative of the expected log-likelihood</i>
------------------------------	--

Description

One component of the second derivative of the expected log-likelihood

Usage

`lnorm_dmgs_gg22(alpha, v1, d1, v2, fd2)`

Arguments

- | | |
|--------------------|--|
| <code>alpha</code> | a vector of values of alpha (one minus probability) |
| <code>v1</code> | first parameter |
| <code>d1</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>v2</code> | second parameter |
| <code>fd2</code> | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Scalar value

lnorm_dmgs_loglik	<i>log-likelihood function</i>
-------------------	--------------------------------

Description

log-likelihood function

Usage

```
lnorm_dmgs_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

lnorm_dmgs_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
----------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lnorm_dmgs_logscores(logscores, x, d1 = 0.01, fd2 = 0.01)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Two scalars

<code>lnorm_dmgs_means</code>	<i>MLE and RHP predictive means</i>
-------------------------------	-------------------------------------

Description

MLE and RHP predictive means

Usage

```
lnorm_dmgs_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

<code>means</code>	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
<code>ml_params</code>	parameters
<code>lddi</code>	inverse observed information matrix
<code>lddd</code>	third derivative of log-likelihood
<code>lambdad_rhp</code>	derivative of the log RHP prior
<code>nx</code>	length of training data
<code>dim</code>	number of parameters

Value

Two scalars

<code>lnorm_dmgs_mu1f</code>	<i>DMGS equation 3.3, mu1 term</i>
------------------------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

```
lnorm_dmgs_mu1f(alpha, v1, d1, v2, fd2)
```

Arguments

<code>alpha</code>	a vector of values of alpha (one minus probability)
<code>v1</code>	first parameter
<code>d1</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>fd2</code>	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

<code>lnorm_dmgs_mu2f</code>	<i>DMGS equation 3.3, mu2 term</i>
------------------------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

`lnorm_dmgs_mu2f(alpha, v1, d1, v2, fd2)`

Arguments

- | | |
|--------------------|--|
| <code>alpha</code> | a vector of values of alpha (one minus probability) |
| <code>v1</code> | first parameter |
| <code>d1</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>v2</code> | second parameter |
| <code>fd2</code> | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

<code>lnorm_dmgs_p1f</code>	<i>DMGS equation 3.3, p1 term</i>
-----------------------------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

`lnorm_dmgs_p1f(y, v1, d1, v2, fd2)`

Arguments

- | | |
|------------------|--|
| <code>y</code> | a vector of values at which to calculate the density and distribution functions |
| <code>v1</code> | first parameter |
| <code>d1</code> | the delta used in the numerical derivatives with respect to the parameter |
| <code>v2</code> | second parameter |
| <code>fd2</code> | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Matrix

<code>lnorm_dmgs_p2f</code>	<i>DMGS equation 3.3, p2 term</i>
-----------------------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

`lnorm_dmgs_p2f(y, v1, d1, v2, fd2)`

Arguments

- `y` a vector of values at which to calculate the density and distribution functions
- `v1` first parameter
- `d1` the delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `fd2` the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

<code>lnorm_dmgs_waic</code>	<i>Waic</i>
------------------------------	-------------

Description

Waic

Usage

```
lnorm_dmgs_waic(  
  waicscores,  
  x,  
  v1hat,  
  d1,  
  v2hat,  
  fd2,  
  lddi,
```

```
      lddd,  
      lambdad,  
      aderivs  
    )
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

lnorm_f1f	<i>DMGS equation 3.3, f1 term</i>
-----------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

```
lnorm_f1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

<code>lnorm_f1fa</code>	<i>The first derivative of the density</i>
-------------------------	--

Description

The first derivative of the density

Usage

`lnorm_f1fa(x, v1, v2)`

Arguments

- `x` a vector of training data values
- `v1` first parameter
- `v2` second parameter

Value

Vector

<code>lnorm_f2f</code>	<i>DMGS equation 3.3, f2 term</i>
------------------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

`lnorm_f2f(y, v1, d1, v2, fd2)`

Arguments

- `y` a vector of values at which to calculate the density and distribution functions
- `v1` first parameter
- `d1` the delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `fd2` the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

lnorm_f2fa	<i>The second derivative of the density</i>
------------	---

Description

The second derivative of the density

Usage

```
lnorm_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

lnorm_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_1dd	<i>Second derivative matrix of the lnormalized log-likelihood</i>
-----------	---

Description

Second derivative matrix of the lnormalized log-likelihood

Usage

```
lnorm_1dd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

lnorm_ldda	<i>The second derivative of the normalized log-likelihood</i>
------------	---

Description

The second derivative of the normalized log-likelihood

Usage

lnorm_ldda(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Matrix

lnorm_lddd	<i>Third derivative tensor of the lnormalized log-likelihood</i>
------------	--

Description

Third derivative tensor of the lnormalized log-likelihood

Usage

lnorm_lddd(x, v1, d1, v2, fd2)

Arguments

- x a vector of training data values
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

<code>lnorm_lddda</code>	<i>The third derivative of the normalized log-likelihood</i>
--------------------------	--

Description

The third derivative of the normalized log-likelihood

Usage

`lnorm_lddda(x, v1, v2)`

Arguments

- `x` a vector of training data values
- `v1` first parameter
- `v2` second parameter

Value

3d array

<code>lnorm_lmn</code>	<i>One component of the second derivative of the normalized log-likelihood</i>
------------------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

`lnorm_lmn(x, v1, d1, v2, fd2, mm, nn)`

Arguments

- `x` a vector of training data values
- `v1` first parameter
- `d1` the delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `fd2` the fractional delta used in the numerical derivatives with respect to the parameter
- `mm` an index for which derivative to calculate
- `nn` an index for which derivative to calculate

Value

Scalar value

lnorm_lmp	<i>One component of the second derivative of the normalized log-likelihood</i>
-----------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

lnorm_lmp(x, v1, d1, v2, fd2, mm, nn, rr)

Arguments

- x a vector of training data values
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter
- mm an index for which derivative to calculate
- nn an index for which derivative to calculate
- rr an index for which derivative to calculate

Value

Scalar value

lnorm_logf	<i>Logf for RUST</i>
------------	----------------------

Description

Logf for RUST

Usage

lnorm_logf(params, x)

Arguments

- params model parameters for calculating logf
- x a vector of training data values

Value

Scalar value.

lnorm_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

lnorm_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
-----------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

lnorm_logscores(logscores, x)

Arguments

- | | |
|-----------|---|
| logscores | logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime) |
| x | a vector of training data values |

Value

Two scalars

lnorm_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
-------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

lnorm_mu1fa(alpha, v1, v2)

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| v2 | second parameter |

Value

Vector

lnorm_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
-------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

lnorm_mu2fa(alpha, v1, v2)

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| v2 | second parameter |

Value

Matrix

lnorm_p1fa	<i>The first derivative of the cdf</i>
------------	--

Description

The first derivative of the cdf

Usage

lnorm_p1fa(x, v1, v2)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |

Value

Vector

lnorm_p1_cp

*Log-normal Distribution with a Predictor, Predictions Based on a Calibrating Prior***Description**

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  centering = TRUE,
```

```

    debug = FALSE,
    aderivs = TRUE
  )

  rlnorm_p1_cp(
    n,
    x,
    t,
    t0 = NA,
    n0 = NA,
    rust = FALSE,
    mlcp = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  dlnorm_p1_cp(
    x,
    t,
    t0 = NA,
    n0 = NA,
    y = x,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  plnorm_p1_cp(
    x,
    t,
    t0 = NA,
    n0 = NA,
    y = x,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  tlnorm_p1_cp(n, x, t, debug = FALSE)

```

Arguments

<i>x</i>	a vector of training data values
<i>t</i>	a vector of predictors, such that <code>length(t)=length(x)</code>

t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.

- `adjustedx`: the detrended values of x

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The log normal distribution with a predictor has probability density function

$$f(x; a, b, \sigma) = \frac{1}{\sqrt{2\pi}x\sigma} e^{-(\log(x) - \mu(a,b))^2 / (2\sigma^2)}$$

where x is the random variable, $\mu = a + bt$ is the location parameter of the log of the random variable, modelled as a function of parameters a, b and predictor t , and $\sigma > 0$ is the scale parameter of the log of the random variable.

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b, \sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d61lnorm_p1_example_data_v1_x
tt=fitdistcp::d61lnorm_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qlnorm_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlnorm_p1_cp)",
main="Log-Normal w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

lnorm_p1_f1f	<i>DMGS equation 2.1, f1 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, f1 term

Usage

```
lnorm_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| v3 | third parameter |
| fd3 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Matrix

lnorm_p1_f1fa	<i>The first derivative of the density</i>
---------------	--

Description

The first derivative of the density

Usage

lnorm_p1_f1fa(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Vector

lnorm_p1_f2f	<i>DMGS equation 2.1, f2 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

lnorm_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- v3 third parameter
- fd3 the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

lnorm_p1_f2fa	<i>The second derivative of the density</i>
---------------	---

Description

The second derivative of the density

Usage

lnorm_p1_f2fa(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Matrix

lnorm_p1_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

lnorm_p1_fd(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Vector

lnorm_p1_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

lnorm_p1_fdd(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Matrix

lnorm_p1_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
--------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

lnorm_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

lnorm_p1_ldda	<i>The second derivative of the normalized log-likelihood</i>
---------------	---

Description

The second derivative of the normalized log-likelihood

Usage

lnorm_p1_ldda(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lnorm_p1_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
---------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

lnorm_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3)

Arguments

- | | |
|-----|--|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| v3 | third parameter |
| fd3 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Cubic scalar array

lnorm_p1_lddda	<i>The third derivative of the normalized log-likelihood</i>
----------------	--

Description

The third derivative of the normalized log-likelihood

Usage

lnorm_p1_lddda(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

3d array

lnorm_p1_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lnorm_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

lnorm_p1_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lnorm_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

lnorm_p1_logf	<i>Logf for RUST</i>
---------------	----------------------

Description

Logf for RUST

Usage

```
lnorm_p1_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

lnorm_p1_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

lnorm_p1_logfdd(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lnorm_p1_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

lnorm_p1_loglik	<i>Log-normal-with-p1 observed log-likelihood function</i>
-----------------	--

Description

Log-normal-with-p1 observed log-likelihood function

Usage

```
lnorm_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

<code>lnorm_p1_logscores</code>	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
---------------------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lnorm_p1_logscores(logscores, x, t)
```

Arguments

- | | |
|------------------------|---|
| <code>logscores</code> | logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime) |
| <code>x</code> | a vector of training data values |
| <code>t</code> | a vector or matrix of predictors |

Value

Two scalars

<code>lnorm_p1_mu1fa</code>	<i>Minus the first derivative of the cdf, at alpha</i>
-----------------------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

```
lnorm_p1_mu1fa(alpha, t, v1, v2, v3)
```

Arguments

- | | |
|--------------------|---|
| <code>alpha</code> | a vector of values of alpha (one minus probability) |
| <code>t</code> | a vector or matrix of predictors |
| <code>v1</code> | first parameter |
| <code>v2</code> | second parameter |
| <code>v3</code> | third parameter |

Value

Vector

lnorm_p1_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
----------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

lnorm_p1_mu2fa(alpha, t, v1, v2, v3)

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Matrix

lnorm_p1_p1fa	<i>The first derivative of the cdf</i>
---------------	--

Description

The first derivative of the cdf

Usage

lnorm_p1_p1fa(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Vector

lnorm_p1_p2fa	<i>The second derivative of the cdf</i>
---------------	---

Description

The second derivative of the cdf

Usage

lnorm_p1_p2fa(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Matrix

lnorm_p1_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

lnorm_p1_pd(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Vector

lnorm_p1_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

lnorm_p1_pdd(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Matrix

lnorm_p1_predictordata	<i>Predicted Parameter and Generalized Residuals</i>
------------------------	--

Description

Predicted Parameter and Generalized Residuals

Usage

lnorm_p1_predictordata(x, t, t0, params)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

lnorm_p1_waic	<i>Waic</i>
---------------	-------------

Description

Waic

Usage

```
lnorm_p1_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  v3hat,  
  fd3,  
  aderivs = TRUE  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

lnorm_p2fa	<i>The second derivative of the cdf</i>
------------	---

Description

The second derivative of the cdf

Usage

```
lnorm_p2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

<code>lnorm_pdd</code>	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

`lnorm_pdd(x, v1, v2)`

Arguments

- `x` a vector of training data values
- `v1` first parameter
- `v2` second parameter

Value

Matrix

<code>lnorm_waic</code>	<i>Waic for RUST</i>
-------------------------	----------------------

Description

Waic for RUST

Usage

`lnorm_waic(waiccores, x, v1hat, d1, v2hat, fd2, aderivs)`

Arguments

- `waiccores` logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
- `x` a vector of training data values
- `v1hat` first parameter
- `d1` the delta used in the numerical derivatives with respect to the parameter
- `v2hat` second parameter
- `fd2` the fractional delta used in the numerical derivatives with respect to the parameter
- `aderivs` logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

logis_cp

Logistic Distribution Predictions Based on a Calibrating Prior

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlogis_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
```

```

)

rlogis_cp(
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dlogis_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

plogis_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tlogis_cp(n, x, d1 = 0.01, fd2 = 0.01, debug = FALSE)

```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.

- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The logistic distribution has distribution function

$$f(x; \mu, \sigma) = \frac{1}{1 + e^{-(x-\mu)/\sigma}}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If rust=TRUE:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If rust=TRUE:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If rust=TRUE:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d40logis_example_data_v1
p=c(1:9)/10
q=qlogis_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlogis_cp)",
main="Logistic: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

logis_f1f	<i>DMGS equation 3.3, f1 term</i>
-----------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

logis_f1f(y, v1, d1, v2, fd2)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

logis_f1fa	<i>The first derivative of the density</i>
------------	--

Description

The first derivative of the density

Usage

logis_f1fa(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Vector

logis_f2f	<i>DMGS equation 3.3, f2 term</i>
-----------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

logis_f2f(y, v1, d1, v2, fd2)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

logis_f2fa	<i>The second derivative of the density</i>
------------	---

Description

The second derivative of the density

Usage

logis_f2fa(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Matrix

logis_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

logis_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

logis_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
-----------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
logis_ldd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

logis_ldda	<i>The second derivative of the normalized log-likelihood</i>
------------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
logis_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

logis_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
logis_lddd(x, v1, d1, v2, fd2)
```

Arguments

- x a vector of training data values
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

logis_lddda	<i>The third derivative of the normalized log-likelihood</i>
-------------	--

Description

The third derivative of the normalized log-likelihood

Usage

```
logis_lddda(x, v1, v2)
```

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

3d array

logis_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
-----------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
logis_lmn(x, v1, d1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

logis_lmnp	<i>One component of the third derivative of the normalized log-likelihood</i>
------------	---

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
logis_lmnp(x, v1, d1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

logis_logf	<i>Logf for RUST</i>
------------	----------------------

Description

Logf for RUST

Usage

logis_logf(params, x)

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

logis_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

logis_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

logis_loglik	<i>log-likelihood function</i>
--------------	--------------------------------

Description

log-likelihood function

Usage

logis_loglik(vv, x)

Arguments

- vv parameters
- x a vector of training data values

Value

Scalar value.

logis_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
-----------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

logis_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)

Arguments

- logscores logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
- x a vector of training data values
- d1 the delta used in the numerical derivatives with respect to the parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter
- aderivs logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

logis_mu1f	<i>DMGS equation 3.3, mu1 term</i>
------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

```
logis_mu1f(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

logis_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
-------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

```
logis_mu1fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Vector

logis_mu2f	<i>DMGS equation 3.3, mu2 term</i>
------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

logis_mu2f(alpha, v1, d1, v2, fd2)

Arguments

- | | |
|-------|--|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

logis_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
-------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

logis_mu2fa(alpha, v1, v2)

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| v2 | second parameter |

Value

Matrix

logis_p1f	<i>DMGS equation 3.3, p1 term</i>
-----------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

logis_p1f(y, v1, d1, v2, fd2)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

logis_p1fa	<i>The first derivative of the cdf</i>
------------	--

Description

The first derivative of the cdf

Usage

logis_p1fa(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Vector

logis_p1_cp

*Logistic Distribution with a Predictor, Predictions Based on a Calibrating Prior***Description**

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlogis_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
```

```
    predictordata = TRUE,  
    centering = TRUE,  
    debug = FALSE,  
    aderivs = TRUE  
  )
```

```
rlogis_p1_cp(  
  n,  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  d1 = 0.01,  
  d2 = 0.01,  
  fd3 = 0.01,  
  rust = FALSE,  
  mlcp = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
dlogis_p1_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  y = x,  
  d1 = 0.01,  
  d2 = 0.01,  
  fd3 = 0.01,  
  rust = FALSE,  
  nrust = 1000,  
  centering = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
plogis_p1_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  y = x,  
  d1 = 0.01,  
  d2 = 0.01,  
  fd3 = 0.01,  
  rust = FALSE,  
  nrust = 1000,
```

```

    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  tlogis_p1_cp(n, x, t, d1 = 0.01, d2 = 0.01, fd3 = 0.01, debug = FALSE)

```

Arguments

<code>x</code>	a vector of training data values
<code>t</code>	a vector of predictors, such that <code>length(t)=length(x)</code>
<code>t0</code>	a single value of the predictor (specify either <code>t0</code> or <code>n0</code> but not both)
<code>n0</code>	an index for the predictor (specify either <code>t0</code> or <code>n0</code> but not both)
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>d1</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the first parameter
<code>d2</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the second parameter
<code>fd3</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the third parameter
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>waicscores</code>	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
<code>logscores</code>	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
<code>dmgs</code>	logical that indicates whether DMGS calculations should be run or not (longer run time)
<code>rust</code>	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
<code>nrust</code>	the number of posterior samples used in the RUST calculations
<code>predictordata</code>	logical that indicates whether <code>predictordata</code> should be calculated
<code>centering</code>	logical that indicates whether the predictor should be centered
<code>debug</code>	logical for turning on debug messages
<code>aderivs</code>	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
<code>n</code>	the number of random samples required
<code>mlcp</code>	logical that indicates whether <code>maxlik</code> and parameter uncertainty calculations should be performed (turn off to speed up RUST)
<code>y</code>	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of x

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The logistic distribution with a predictor has distribution function

$$f(x; a, b, \sigma) = \frac{1}{1 + e^{-(x - \mu(a, b))/\sigma}}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),

- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d62logis_p1_example_data_v1_x
tt=fitdistcp::d62logis_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qlogis_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlogis_p1_cp)",
main="Logistic w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

logis_p1_f1f

DMGS equation 2.1, f1 term

Description

DMGS equation 2.1, f1 term

Usage

```
logis_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

logis_p1_f1fa	<i>The first derivative of the density</i>
---------------	--

Description

The first derivative of the density

Usage

logis_p1_f1fa(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

logis_p1_f2f	<i>DMGS equation 2.1, f2 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

logis_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| v3 | third parameter |
| fd3 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

logis_p1_f2fa	<i>The second derivative of the density</i>
---------------	---

Description

The second derivative of the density

Usage

logis_p1_f2fa(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Matrix

logis_p1_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_fd(x, t, v1, v2, v3)
```

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Vector

logis_p1_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

logis_p1_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
--------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
logis_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

logis_p1_ldda	<i>The second derivative of the normalized log-likelihood</i>
---------------	---

Description

The second derivative of the normalized log-likelihood

Usage

logis_p1_ldda(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Matrix

logis_p1_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
---------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

logis_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- v3 third parameter
- fd3 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

logis_p1_lddda	<i>The third derivative of the normalized log-likelihood</i>
----------------	--

Description

The third derivative of the normalized log-likelihood

Usage

```
logis_p1_lddda(x, t, v1, v2, v3)
```

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

3d array

logis_p1_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
logis_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

logis_p1_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

logis_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn, rr)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

logis_p1_logf	<i>Logf for RUST</i>
---------------	----------------------

Description

Logf for RUST

Usage

logis_p1_logf(params, x, t)

Arguments

- | | |
|--------|---------------------------------------|
| params | model parameters for calculating logf |
| x | a vector of training data values |
| t | a vector or matrix of predictors |

Value

Scalar value.

logis_p1_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

logis_p1_logfdd(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Scalar value.

logis_p1_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
--------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
logis_p1_logscores(logscores, x, t, d1, d2, fd3, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

logis_p1_means	<i>Logistic distribution: RHP mean</i>
----------------	--

Description

Logistic distribution: RHP mean

Usage

```
logis_p1_means(t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

logis_p1_mu1f	<i>DMGS equation 3.3, mu1 term</i>
---------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

logis_p1_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3)

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

logis_p1_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
----------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

```
logis_p1_mu1fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

logis_p1_mu2f	<i>DMGS equation 3.3, mu2 term</i>
---------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

```
logis_p1_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

logis_p1_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
----------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

```
logis_p1_mu2fa(alpha, t, v1, v2, v3)
```

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Matrix

logis_p1_p1f	<i>DMGS equation 2.1, p1 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, p1 term

Usage

```
logis_p1_p1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

logis_p1_p1fa	<i>The first derivative of the cdf</i>
---------------	--

Description

The first derivative of the cdf

Usage

logis_p1_p1fa(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

logis_p1_p2f	<i>DMGS equation 2.1, p2 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, p2 term

Usage

logis_p1_p2f(y, t0, v1, d1, v2, d2, v3, fd3)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| v3 | third parameter |
| fd3 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

logis_p1_p2fa	<i>The second derivative of the cdf</i>
---------------	---

Description

The second derivative of the cdf

Usage

logis_p1_p2fa(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Matrix

logis_p1_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_pd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

logis_p1_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_pdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

logis_p1_predictordata
<i>Predicted Parameter and Generalized Residuals</i>

Description

Predicted Parameter and Generalized Residuals

Usage

```
logis_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

logis_p1_waic	<i>Waic</i>
---------------	-------------

Description

Waic

Usage

```
logis_p1_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  v3hat,  
  fd3,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs = TRUE  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

logis_p2f	<i>DMGS equation 3.3, p2 term</i>
-----------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

logis_p2f(y, v1, d1, v2, fd2)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

logis_p2fa	<i>The second derivative of the cdf</i>
------------	---

Description

The second derivative of the cdf

Usage

logis_p2fa(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Vector

logis_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

logis_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

logis_waic	<i>Waic</i>
------------	-------------

Description

Waic

Usage

```
logis_waic(waiccores, x, v1hat, d1, v2hat, fd2, lddi, lddd, lambdad, aderivs)
```

Arguments

waiccores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

lst_k3_cp	<i>t Distribution Predictions Based on a Calibrating Prior</i>
-----------	--

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlst_k3_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kdf = 5,
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rlst_k3_cp(
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  kdf = 5,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
dlst_k3_cp(
  x,
  y = x,
```

```

    d1 = 0.01,
    fd2 = 0.01,
    kdf = 5,
    rust = FALSE,
    nrust = 1000,
    debug = FALSE,
    aderivs = TRUE
)

plst_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kdf = 5,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tlst_k3_cp(n, x, d1 = 0.01, fd2 = 0.01, kdf = 5, debug = FALSE)

```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>kdf</code>	the known degrees of freedom parameter
<code>d1</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the first parameter
<code>fd2</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the second parameter
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>waicscores</code>	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
<code>logscores</code>	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
<code>dmgs</code>	logical that indicates whether DMGS calculations should be run or not (longer run time)
<code>rust</code>	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
<code>nrust</code>	the number of posterior samples used in the RUST calculations
<code>debug</code>	logical for turning on debug messages
<code>aderivs</code>	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.

n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The t distribution (also known as the location-scale t distribution, hence the name `lst`), has probability density function

$$f(x; \mu, \sigma) = \frac{\Gamma((\nu + 1)/2)}{\sqrt{\pi\nu\sigma}\Gamma(\nu/2)} \left(1 + \frac{(x - \mu)^2}{\sigma^2\nu}\right)^{-(\nu+1)/2}$$

where x is the random variable, $\mu, \sigma > 0$ are the parameters, and we consider the degrees of freedom ν to be known (hence the `k3` in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),

- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d411st_k3_example_data_v1
p=c(1:9)/10
q=qlst_k3_cp(x,p,kdf=5,rust=TRUE,nrust=1000)
xmin=min(q$m1_quantiles,q$cp_quantiles);
xmax=max(q$m1_quantiles,q$cp_quantiles);
plot(q$m1_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlst_k3_cp)",
main="t: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

1st_k3_f1f

DMGS equation 3.3, f1 term

Description

DMGS equation 3.3, f1 term

Usage

```
1st_k3_f1f(y, v1, d1, v2, fd2, kdf)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Matrix

lst_k3_f1fa	<i>The first derivative of the density</i>
-------------	--

Description

The first derivative of the density

Usage

lst_k3_f1fa(x, v1, v2, kdf)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kdf	the known degrees of freedom parameter

Value

Vector

1st_k3_f2f	<i>DMGS equation 3.3, f2 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

1st_k3_f2f(y, v1, d1, v2, fd2, kdf)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| kdf | the known degrees of freedom parameter |

Value

3d array

1st_k3_f2fa	<i>The second derivative of the density</i>
-------------	---

Description

The second derivative of the density

Usage

1st_k3_f2fa(x, v1, v2, kdf)

Arguments

- | | |
|-----|--|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |
| kdf | the known degrees of freedom parameter |

Value

Matrix

1st_k3_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
1st_k3_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

1st_k3_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
1st_k3_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

l1st_k3_l1dd	<i>Second derivative matrix of the normalized log-likelihood</i>
--------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
l1st_k3_l1dd(x, v1, d1, v2, fd2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Square scalar matrix

l1st_k3_l1dda	<i>The second derivative of the normalized log-likelihood</i>
---------------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
l1st_k3_l1dda(x, v1, v2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kdf	the known degrees of freedom parameter

Value

Matrix

lst_k3_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
-------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
lst_k3_lddd(x, v1, d1, v2, fd2, kdf)
```

Arguments

- | | |
|-----|--|
| x | a vector of training data values |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| kdf | the known degrees of freedom parameter |

Value

Cubic scalar array

lst_k3_lddda	<i>The third derivative of the normalized log-likelihood</i>
--------------	--

Description

The third derivative of the normalized log-likelihood

Usage

```
lst_k3_lddda(x, v1, v2, kdf)
```

Arguments

- | | |
|-----|--|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |
| kdf | the known degrees of freedom parameter |

Value

3d array

1st_k3_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
1st_k3_lmn(x, v1, d1, v2, fd2, kdf, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

1st_k3_lmp	<i>One component of the third derivative of the normalized log-likelihood</i>
------------	---

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
1st_k3_lmp(x, v1, d1, v2, fd2, kdf, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

lst_k3_logf	<i>Logf for RUST</i>
-------------	----------------------

Description

Logf for RUST

Usage

lst_k3_logf(params, x, kdf)

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kdf	the known degrees of freedom parameter

Value

Scalar value.

1st_k3_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
1st_k3_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

1st_k3_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
1st_k3_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

lst_k3_loglik	<i>log-likelihood function</i>
---------------	--------------------------------

Description

log-likelihood function

Usage

```
lst_k3_loglik(vv, x, kdf)
```

Arguments

- vv parameters
- x a vector of training data values
- kdf the known degrees of freedom parameter

Value

Scalar value.

lst_k3_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lst_k3_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, kdf, aderivs = TRUE)
```

Arguments

- logscores logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
- x a vector of training data values
- d1 the delta used in the numerical derivatives with respect to the parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter
- kdf the known degrees of freedom parameter
- aderivs logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

1st_k3_mu1f	<i>DMGS equation 3.3, mu1 term</i>
-------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

```
1st_k3_mu1f(alpha, v1, d1, v2, fd2, kdf)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Matrix

1st_k3_mu2f	<i>DMGS equation 3.3, mu2 term</i>
-------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

```
1st_k3_mu2f(alpha, v1, d1, v2, fd2, kdf)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

lst_k3_p1f	<i>DMGS equation 3.3, p1 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

lst_k3_p1f(y, v1, d1, v2, fd2, kdf)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |
| kdf | the known degrees of freedom parameter |

Value

Matrix

lst_k3_p2f	<i>DMGS equation 3.3, p2 term</i>
------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

lst_k3_p2f(y, v1, d1, v2, fd2, kdf)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

lst_k3_waic	<i>Waic</i>
-------------	-------------

Description

Waic

Usage

```
lst_k3_waic(  
  waicscores,  
  x,  
  v1hat,  
  d1,  
  v2hat,  
  fd2,  
  kdf,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter

fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

lst_p1k3_cp	<i>t Distribution with a Predictor, Predictions Based on a Calibrating Prior</i>
-------------	--

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlst_p1k3_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  p = seq(0.1, 0.9, 0.1),  
  d1 = 0.01,  
  d2 = 0.01,  
  fd3 = 0.01,  
  kdf = 10,  
  means = FALSE,  
  waicscores = FALSE,  
  logscores = FALSE,  
  dmgs = TRUE,  
  rust = FALSE,  
  nrust = 1e+05,  
  predictordata = TRUE,  
  centering = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
r1st_p1k3_cp(  
  n,  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  d1 = 0.01,  
  d2 = 0.01,  
  fd3 = 0.01,  
  kdf = 10,  
  rust = FALSE,  
  mlcp = TRUE,  
  centering = TRUE,  
  debug = FALSE,  
  aderivs = TRUE  
)
```

```
dlst_p1k3_cp(  
  x,  
  t,  
  t0 = NA,  
  n0 = NA,  
  y = x,  
  d1 = 0.01,  
  d2 = 0.01,
```

```

    fd3 = 0.01,
    kdf = 10,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

```

```

plst_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kdf = 10,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

```

```

tlst_p1k3_cp(
  n,
  x,
  t,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kdf = 10,
  debug = FALSE
)

```

Arguments

x	a vector of training data values
t	a vector of predictors, such that <code>length(t)=length(x)</code>
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter

fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
kdf	the known degrees of freedom parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The t distribution with a predictor (also known as the location-scale t distribution with a predictor, hence the name `lst`), has probability density function

$$f(x; a, b, \sigma) = \frac{\Gamma((\nu + 1)/2)}{\sqrt{\pi\nu}\sigma\Gamma(\nu/2)} \left(1 + \frac{(x - \mu(a, b))^2}{\sigma^2\nu} \right)^{-(\nu+1)/2}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, and $\sigma > 0$ is the scale parameter. We consider the degrees of freedom ν to be known (hence the `k3` in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean (`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d63lst_p1k3_example_data_v1_x
tt=fitdistcp::d63lst_p1k3_example_data_v1_t
p=c(1:9)/10
n0=10
q=qlst_p1k3_cp(x,tt,n0=n0,p=p,kdf=5,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlst_p1k3_cp)",
main="t w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

lst_p1k3_f1f	<i>DMGS equation 2.1, f1 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, f1 term

Usage

```
lst_p1k3_f1f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Matrix

lst_p1k3_f1fa	<i>The first derivative of the density</i>
---------------	--

Description

The first derivative of the density

Usage

lst_p1k3_f1fa(x, t, v1, v2, v3, kdf)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter
- kdf the known degrees of freedom parameter

Value

Vector

lst_p1k3_f2f	<i>DMGS equation 2.1, f2 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

lst_p1k3_f2f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

lst_p1k3_f2fa	<i>The second derivative of the density</i>
---------------	---

Description

The second derivative of the density

Usage

lst_p1k3_f2fa(x, t, v1, v2, v3, kdf)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kdf	the known degrees of freedom parameter

Value

Matrix

1st_p1k3_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
1st_p1k3_fd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

1st_p1k3_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
1st_p1k3_fdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

lst_p1k3_1dd	<i>Second derivative matrix of the normalized log-likelihood</i>
--------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

lst_p1k3_1dd(x, t, v1, d1, v2, d2, v3, fd3, kdf)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Square scalar matrix

lst_p1k3_ldda	<i>The second derivative of the normalized log-likelihood</i>
---------------	---

Description

The second derivative of the normalized log-likelihood

Usage

lst_p1k3_ldda(x, t, v1, v2, v3, kdf)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter
- kdf the known degrees of freedom parameter

Value

Matrix

lst_p1k3_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
---------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

lst_p1k3_lddd(x, t, v1, d1, v2, d2, v3, fd3, kdf)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter

v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Cubic scalar array

lst_p1k3_lddda	<i>The third derivative of the normalized log-likelihood</i>
----------------	--

Description

The third derivative of the normalized log-likelihood

Usage

lst_p1k3_lddda(x, t, v1, v2, v3, kdf)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kdf	the known degrees of freedom parameter

Value

3d array

lst_p1k3_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lst_p1k3_lmn(x, t, v1, d1, v2, d2, v3, fd3, kdf, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

lst_p1k3_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
--------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lst_p1k3_lmn(x, t, v1, d1, v2, d2, v3, fd3, kdf, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

lst_p1k3_logf	<i>Logf for RUST</i>
---------------	----------------------

Description

Logf for RUST

Usage

lst_p1k3_logf(params, x, t, kdf)

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors
kdf	the known degrees of freedom parameter

Value

Scalar value.

1st_p1k3_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
1st_p1k3_logfdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

1st_p1k3_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
1st_p1k3_logfddd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

lst_p1k3_loglik	<i>LST-with-p1 observed log-likelihood function</i>
-----------------	---

Description

LST-with-p1 observed log-likelihood function

Usage

```
lst_p1k3_loglik(vv, x, t, kdf)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors
kdf	the known degrees of freedom parameter

Value

Scalar value.

lst_p1k3_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
--------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lst_p1k3_logscores(logscores, x, t, d1, d2, fd3, kdf, aderivs)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

lst_p1k3_mu1f	<i>DMGS equation 3.3, mu1 term</i>
---------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

```
lst_p1k3_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Matrix

lst_p1k3_mu2f	<i>DMGS equation 3.3, mu2 term</i>
---------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

lst_p1k3_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3, kdf)

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

lst_p1k3_p1f	<i>DMGS equation 2.1, p1 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, p1 term

Usage

lst_p1k3_p1f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)

Arguments

y	value of random variable
t0	value of predictor
v1	first parameter
d1	delta for numerical derivative
v2	second parameter
d2	delta for numerical derivative
v3	third parameter
fd3	fractional delta for numerical derivative
kdf	the known number of degrees of freedom

Value

Matrix

lst_p1k3_p2f	<i>DMGS equation 2.1, p2 term</i>
--------------	-----------------------------------

Description

DMGS equation 2.1, p2 term

Usage

lst_p1k3_p2f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

lst_p1k3_predictordata
<i>Predicted Parameter and Generalized Residuals</i>

Description

Predicted Parameter and Generalized Residuals

Usage

```
lst_p1k3_predictordata(predictordata, x, t, t0, params, kdf)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf
kdf	the known degrees of freedom parameter

Value

Two vectors

lst_p1k3_setics	<i>Set initial conditions</i>
-----------------	-------------------------------

Description

Set initial conditions

Usage

```
lst_p1k3_setics(x, t, ics)
```

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- ics initial conditions for the maximum likelihood search

Value

Vector

lst_p1k3_waic	<i>Waic</i>
---------------	-------------

Description

Waic

Usage

```
lst_p1k3_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  v3hat,  
  fd3,  
  kdf,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

<code>waicscores</code>	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
<code>x</code>	a vector of training data values
<code>t</code>	a vector or matrix of predictors
<code>v1hat</code>	first parameter
<code>d1</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v2hat</code>	second parameter
<code>d2</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v3hat</code>	third parameter
<code>fd3</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>kdf</code>	the known degrees of freedom parameter
<code>lddi</code>	inverse observed information matrix
<code>lddd</code>	third derivative of log-likelihood
<code>lambdad</code>	derivative of the log prior
<code>aderivs</code>	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

<code>makemuhat0</code>	<i>Make muhat0</i>
-------------------------	--------------------

Description

Make muhat0

Usage

`makemuhat0(t0, n0, t, mle_params)`

Arguments

<code>t0</code>	the value of the predictor vector at which to make the prediction (if <code>n0</code> not specified)
<code>n0</code>	the position in the predictor vector at which to make the prediction (positive integer less than or equal to the length of <i>x</i>) (if <code>t0</code> not specified)
<code>t</code>	predictor
<code>mle_params</code>	MLE params

Value

Scalar

makeq	<i>Calculates quantiles from simulations by inverting the Hazen CDF</i>
-------	---

Description

Calculates quantiles from simulations by inverting the Hazen CDF

Usage

makeq(yy, pp)

Arguments

- | | |
|----|-------------------------|
| yy | vector of samples |
| pp | vector of probabilities |

Value

Vector

maket0	<i>Determine t0</i>
--------	---------------------

Description

Determine t0

Usage

maket0(t0, n0, t)

Arguments

- | | |
|----|--|
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| n0 | an index for the predictor (specify either t0 or n0 but not both) |
| t | a vector or matrix of predictors |

Value

Scalar

maketa0	<i>Make ta0</i>
---------	-----------------

Description

Make ta0

Usage

```
maketa0(t0, n0, t)
```

Arguments

t0	the value of the predictor vector at which to make the prediction (if n0 not specified)
n0	the position in the predictor vector at which to make the prediction (positive integer less than or equal to the length of x) (if t0 not specified)
t	predictor

Value

Scalar

make_cwaic	<i>Make WAIC</i>
------------	------------------

Description

Make WAIC

Usage

```
make_cwaic(x, fhatx, lddi, lddd, f1f, lambdad, f2f, dim)
```

Arguments

x	the training data
fhatx	density of x at the maximum likelihood parameters
lddi	inverse of the second derivative log-likelihood matrix
lddd	the third derivative log-likelihood tensor
f1f	the f1 term from DMGS equation 2.1
lambdad	the slope of the log prior
f2f	the f2 term from DMGS equation 2.1
dim	number of free parameters

Value

Two scalars

make_maic	<i>Calculate MAIC</i>
-----------	-----------------------

Description

Calculate MAIC

Usage

```
make_maic(ml_value, nparams)
```

Arguments

ml_value	maximum of the likelihood
nparams	number of parameters

Value

Vector of 3 values Returns the two components of MAIC, and their sum

make_se	<i>Make Standard Errors from lddi</i>
---------	---------------------------------------

Description

Make Standard Errors from lddi

Usage

```
make_se(nx, lddi)
```

Arguments

nx	length of training data
lddi	the inverse log-likelihood matrix

Value

Vector

make_waic	<i>Make WAIC</i>
Description	
Make WAIC	
Usage	
make_waic(x, fhatx, lddi, lddd, f1f, lambdad, f2f, dim)	
Arguments	
x	the training data
fhatx	density of x at the maximum likelihood parameters
lddi	inverse of the second derivative log-likelihood matrix
lddd	the third derivative log-likelihood tensor
f1f	the f1 term from DMGS equation 2.1
lambdad	the slope of the log prior
f2f	the f2 term from DMGS equation 2.1
dim	number of free parameters
Value	
Two scalars	
man	<i>A blank function I use for setting up the man page information</i>

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`

- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
man(
  x,
  t,
  t1,
  t2,
  t3,
  t0,
  t01,
  t02,
  t03,
  t10,
  t20,
  n0,
  n01,
  n02,
  n03,
  n10,
  n20,
  p,
  n,
  y,
  ics,
  kloc,
  kscale,
  kshape,
  kdf,
  kbeta,
  d1,
  fd1,
  d2,
  fd2,
  d3,
  fd3,
  d4,
  fd4,
  d5,
```

```

    fd5,
    d6,
    fd6,
    fdalpha,
    minxi,
    maxxi,
    dlogpi,
    means,
    waicscores,
    logscores,
    extramodels,
    pdf,
    customprior,
    dmgs,
    mlcp,
    predictordata,
    centering,
    nonnegslopesonly,
    rnonnegslopesonly,
    prior,
    debug,
    rust,
    nrust,
    pwm,
    unbiasedv,
    aderivs
  )

```

Arguments

<code>x</code>	a vector of training data values
<code>t</code>	a vector of predictors, such that <code>length(t)=length(x)</code>
<code>t1</code>	a vector of predictors for the mean, such that <code>length(t1)=length(x)</code>
<code>t2</code>	a vector of predictors for the sd, such that <code>length(t2)=length(x)</code>
<code>t3</code>	a vector of predictors for the shape, such that <code>length(t3)=length(x)</code>
<code>t0</code>	a single value of the predictor (specify either <code>t0</code> or <code>n0</code> but not both)
<code>t01</code>	a single value of the predictor (specify either <code>t01</code> or <code>n01</code> but not both)
<code>t02</code>	a single value of the predictor (specify either <code>t02</code> or <code>n02</code> but not both)
<code>t03</code>	a single value of the predictor (specify either <code>t03</code> or <code>n03</code> but not both)
<code>t10</code>	a single value of the predictor for the mean (specify either <code>t10</code> or <code>n10</code> but not both)
<code>t20</code>	a single value of the predictor for the sd (specify either <code>t20</code> or <code>n20</code> but not both)
<code>n0</code>	an index for the predictor (specify either <code>t0</code> or <code>n0</code> but not both)
<code>n01</code>	an index for the predictor (specify either <code>t01</code> or <code>n01</code> but not both)
<code>n02</code>	an index for the predictor (specify either <code>t02</code> or <code>n02</code> but not both)

n03	an index for the predictor (specify either t03 or n03 but not both)
n10	an index for the predictor for the mean (specify either t10 or n10 but not both)
n20	an index for the predictor for the sd (specify either t20 or n20 but not both)
p	a vector of probabilities at which to generate predictive quantiles
n	the number of random samples required
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
kloc	the known location parameter
kscale	the known scale parameter
kshape	the known shape parameter
kdf	the known degrees of freedom parameter
kbeta	the known beta parameter
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
d3	if aderivs=FALSE, the delta used for numerical derivatives with respect to the third parameter
fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
d4	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fourth parameter
fd4	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the fourth parameter
d5	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fifth parameter
fd5	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the fourth parameter
d6	if aderivs=FALSE, the delta used for numerical derivatives with respect to the sixth parameter
fd6	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the fourth parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)

dlogpi	gradient of the log prior
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
customprior	a custom value for the slope of the log prior at the maxlik estimate
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
nonnegslopesonly	logical that indicates whether to disallow non-negative slopes
rnonnegslopesonly	logical that indicates whether to disallow non-negative slopes
prior	logical indicating which prior to use
debug	logical for turning on debug messages
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
pwm	logical for whether to include PWM results (longer runtime)
unbiasedv	logical for whether to include unbiased variance results in norm
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.

- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

*r***** optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

*d***** optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

*p***** optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

*q***** optionally returns the following, for EVT models only:

- `cp_pdf`: the density function at quantiles corresponding to input probabilities `p`. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

*q***** optionally returns the following, for some EVT models only:

If `extramodels=TRUE`:

- `flat_quantiles`: predictive quantiles calculated from Bayesian integration with a flat prior.
- `rh_ml_quantiles`: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximum likelihood estimate for the shape parameter.

- `jp_quantiles`: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

`r****` additionally returns the following, for some EVT models only:

If `extramodels=TRUE`:

- `flat_deviates`: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- `rh_ml_deviates`: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- `jp_deviates`: predictive random deviates calculated using a Bayesian analysis with the JP.

`d****` additionally returns the following, for some EVT models only:

If `extramodels=TRUE`:

- `flat_pdf`: predictive density function from a Bayesian analysis with the flat prior.
- `rh_ml_pdf`: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- `jp_pdf`: predictive density function from a Bayesian analysis with the JP.

`p****` additionally returns the following, for some EVT models only:

If `extramodels=TRUE`:

- `flat_cdf`: predictive distribution function from a Bayesian analysis with the flat prior.
- `rh_ml_cdf`: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- `jp_cdf`: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The `cp` outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean (`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

man1f	<i>Return message for f1f, p1f, mu1f</i>
-------	--

Description

Return message for f1f, p1f, mu1f

Usage

man1f()

Value

Matrix

man2f	<i>Return message for f2f, p2f, mu2f</i>
-------	--

Description

Return message for f2f, p2f, mu2f

Usage

man2f()

Value

3d array

mandsub	<i>Return message for dsub</i>
---------	--------------------------------

Description

Return message for dsub

Usage

mandsub()

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

manf	<i>Blank function I use for setting up the man page information for the functions</i>
------	---

Description

Blank function I use for setting up the man page information for the functions

Usage

```
manf(  
  dim,  
  vv,  
  ml_params,  
  nx,  
  nxx,  
  x,  
  xx,  
  t,  
  t1,  
  t2,  
  t3,  
  tt,  
  tt1,  
  tt2,  
  tt3,  
  tt2d,  
  tt3d,  
  t0,  
  t01,
```

t02,
t03,
t10,
t20,
t30,
n0,
n10,
n20,
p,
n,
y,
ics,
ta,
ta0,
muhat0,
v1,
v1hat,
v1h,
d1,
fd1,
v2,
v2hat,
v2h,
d2,
fd2,
v3,
v3hat,
v3h,
d3,
fd3,
v4,
v4hat,
v4h,
d4,
fd4,
v5,
v5hat,
v5h,
d5,
v6,
v6hat,
v6h,
d6,
minxi,
maxxi,
xmin,
ximax,
fdalpha,

kyscale,
kloc,
kshape,
kdf,
kbeta,
alpha,
ymn,
slope,
mu,
sigma,
sigma1,
sigma2,
scale,
shape,
xi,
xi1,
xi2,
lambda,
log,
mm,
nn,
rr,
lddi,
lddi_k2,
lddi_k3,
lddi_k4,
lddd,
lddd_k2,
lddd_k3,
lddd_k4,
lambdad,
lambdad_cp,
lambdad_rhp,
lambdad_flat,
lambdad_rh_mle,
lambdad_rh_flat,
lambdad_jp,
lambdad_custom,
means,
waicscores,
logscores,
extramodels,
pdf,
predictordata,
nonnegslopesonly,
rnonnegslopesonly,
customprior,
prior,

```

    params,
    yy,
    pp,
    dlogpi,
    debug,
    centering,
    aderivs
)

```

Arguments

dim	number of parameters
vv	parameters
ml_params	parameters
nx	length of training data
nxx	length of training data
x	a vector of training data values
xx	a vector of training data values
t	a vector or matrix of predictors
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
tt	a vector of predictors
tt1	a vector of predictors for the mean
tt2	a vector of predictors for the sd
tt3	a vector of predictors for the shape
tt2d	a matrix of predictors (nx by 2)
tt3d	a matrix of predictors (nx by 3)
t0	a single value of the predictor (specify either t0 or n0 but not both)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
t10	a single value of the predictor for the mean (specify either t10 or n10 but not both)
t20	a single value of the predictor for the sd (specify either t20 or n20 but not both)
t30	a single value of the predictor for the shape (specify either t30 or n30 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
n10	an index for the predictor for the mean (specify either t10 or n10 but not both)
n20	an index for the predictor for the sd (specify either t10 or n10 but not both)

p	a vector of probabilities at which to generate predictive quantiles
n	number of random samples required
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
ta	predictor residuals
ta0	predictor residual at the point being predicted
muhat0	muhat at the point being predicted
v1	first parameter
v1hat	first parameter
v1h	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
v2hat	second parameter
v2h	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
v3hat	third parameter
v3h	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
v4hat	fourth parameter
v4h	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
fd4	the fractional delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
v5hat	fifth parameter
v5h	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
v6hat	sixth parameter
v6h	sixth parameter

d6	the delta used in the numerical derivatives with respect to the parameter
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
ximin	minimum value of shape parameter xi
ximax	maximum value of shape parameter xi
fdalpha	the fractional delta used in the numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
kscale	the known scale parameter
kloc	the known location parameter
kshape	the known shape parameter
kdf	the known degrees of freedom parameter
kbeta	the known beta parameter
alpha	a vector of values of alpha (one minus probability)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
mu	the location parameter of the distribution
sigma	the sigma parameter of the distribution
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
scale	the scale parameter of the distribution
shape	the shape parameter of the distribution
xi	the shape parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution
lambda	the lambda parameter of the distribution
log	logical for the density evaluation
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate
lddi	inverse observed information matrix
lddi_k2	inverse observed information matrix, fixed shape parameter
lddi_k3	inverse observed information matrix, fixed shape parameter
lddi_k4	inverse observed information matrix, fixed shape parameter
lddd	third derivative of log-likelihood
lddd_k2	third derivative of log-likelihood, fixed shape parameter
lddd_k3	third derivative of log-likelihood, fixed shape parameter
lddd_k4	third derivative of log-likelihood, fixed shape parameter

lambdad	derivative of the log prior
lambdad_cp	derivative of the log prior
lambdad_rhp	derivative of the log RHP prior
lambdad_flat	derivative of the log flat prior
lambdad_rh_mle	derivative of the log CRHP-MLE prior
lambdad_rh_flat	derivative of the log CRHP-FLAT prior
lambdad_jp	derivative of the log JP prior
lambdad_custom	custom value of the derivative of the log prior
means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
extramodels	logical that indicates whether to add three additional prediction models
pdf	logical that indicates whether to return density functions evaluated at quantiles specified by input probabilities
predictordata	logical that indicates whether to calculate and return predictordata
nonnegslopesonly	logical that indicates whether to disallow non-negative slopes
rnonnegslopesonly	logical that indicates whether to disallow non-negative slopes
customprior	a custom value for the slope of the log prior at the maxlik estimate
prior	logical indicating which prior to use
params	model parameters for calculating logf
yy	vector of samples
pp	vector of probabilities
dlogpi	gradient of the log prior
debug	debug flag
centering	indicates whether the routine should center the data or not
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

No return value

manl _{dd}	<i>Return message for l_{dd}</i>
--------------------	--

Description

Return message for l_{dd}

Usage

manl_{dd}()

Value

Square scalar matrix

manl _{ddd}	<i>Return message for l_{ddd}</i>
---------------------	---

Description

Return message for l_{ddd}

Usage

manl_{ddd}()

Value

Cubic scalar array

manl _{nn}	<i>Return message for l_{nn}</i>
--------------------	--

Description

Return message for l_{nn}

Usage

manl_{nn}()

Value

Scalar value

manlnnn	<i>Return message for lnnn</i>
---------	--------------------------------

Description

Return message for lnnn

Usage

manlnnn()

Value

Scalar value

manlogf	<i>Return message for Logf</i>
---------	--------------------------------

Description

Return message for Logf

Usage

manlogf()

Value

Scalar value.

manloglik	<i>Return message for loglik</i>
-----------	----------------------------------

Description

Return message for loglik

Usage

manloglik()

Value

Scalar value.

manlogscores	<i>Return message for logscores</i>
--------------	-------------------------------------

Description

Return message for logscores

Usage

manlogscores()

Value

Two scalars

manmeans	<i>Return message for means</i>
----------	---------------------------------

Description

Return message for means

Usage

manmeans()

Value

Two scalars

manpredictor	<i>Return message for predictor.</i>
--------------	--------------------------------------

Description

Return message for predictor.

Usage

manpredictor()

Value

Two vectors

manvector	<i>Return message for vector</i>
-----------	----------------------------------

Description

Return message for vector

Usage

manvector()

Value

Vector

manwaic	<i>Return message for WAIC</i>
---------	--------------------------------

Description

Return message for WAIC

Usage

manwaic()

Value

Two numeric values.

movexiawayfromzero	<i>Move xi away from zero a bit</i>
--------------------	-------------------------------------

Description

Move xi away from zero a bit

Usage

movexiawayfromzero(xi)

Arguments

xi xi

Value

Scalar

`ms_flat_1tail`*Illustration of Model Selection Among 10 One Tail Distributions from the fitdistcp Package*

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x , for 10 one tailed models in the `fitdistcp` package (although for the GPD, the logscore is NA for mathematical reasons).

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the `fitdistcp` routines.

The input data may be automatically shifted so that the minimum value is positive.

For the Pareto, the data may be further shifted so that the minimum value is slightly greater than 1.

Usage

```
ms_flat_1tail(x)
```

Arguments

`x` data vector

Details

The 10 models are: `exp`, `pareto_k2`, `halfnorm`, `lnorm`, `frechet_k1`, `weibull`, `gamma`, `invgamma`, `invgauss` and `gpd_k1`.

Value

Plots QQ plots to the screen, for each of the models, and returns a data frame containing

- MLE parameter values
- AIC scores (times -0.5), AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores, logscore weights
- maximum likelihood and calibrating prior means
- maximum likelihood and calibrating prior standard deviations

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
# because it's too slow for CRAN
set.seed(1)
nx=50
x=rlnorm(nx)
print(ms_flat_1tail(x))
```

ms_flat_2tail	<i>Illustration of Model Selection Among 18 Distributions from the fitdistcp Package</i>
---------------	--

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x , for 7 two tailed models in the fitdistcp packages

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the fitdistcp routines.

Usage

```
ms_flat_2tail(x)
```

Arguments

x	data vector
---	-------------

Details

The 7 models are: norm, gnorm_k3, gumbel, logis, lst_k3, cauchy, gev

Value

Plots QQ plots to the screen, for each of the models, and returns a data frame containing

- AIC scores (times -0.5), AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores, logscore weights
- maximum likelihood and calibrating prior means
- maximum likelihood and calibrating prior standard deviations

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
# because it's too slow for CRAN
set.seed(1)
nx=50
x=rnorm(nx)
print(ms_flat_2tail(x))
```

ms_predictors_1tail	<i>Model Selection Among 5 Distributions with predictors from the fitdistcp Package</i>
---------------------	---

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x, t , for 5 one tailed models with predictors in the fitdistcp package.

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the fitdistcp routines.

The input data may be automatically shifted so that the minimum value is positive.

For the Pareto, the data is so that the minimum value is slightly greater than 1.

Usage

```
ms_predictors_1tail(x, t)
```

Arguments

x	data vector
t	predictor vector

Details

The 5 models are: exp_p1, pareto_p1k2, lnorm_p1, frechet_p2k1, weibull_p2.

Value

Plots QQ plots to the screen, for each of the 5 models, and returns a data frame containing

- AIC scores, AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores and logscore weights

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
# because it's too slow for CRAN
set.seed(2)
nx=100
predictor=c(1:nx)/nx
x=rlnorm(nx,meanlog=predictor,sdlog=0.1)
print(ms_predictors_1tail(x,predictor))
```

ms_predictors_2tail	<i>Model Selection Among 6 Distributions with predictors from the fitdistcp Package</i>
---------------------	---

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x, t , for 6 two tail models with predictors in the fitdistcp packages (although for the GEV, the logscore is NA for mathematical reasons).

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the fitdistcp routines.

GEVD is temperamental in that it doesn't work if the shape parameter is extreme.

Usage

```
ms_predictors_2tail(x, t)
```

Arguments

x	data vector
t	predictor vector

Details

The 11 models are: norm_p1, gumbel_p1, logis_p1, lst_k3_p1, cauchy_p1 and gev_p1.

Value

Plots QQ plots to the screen, for each of the 6 models, and returns a data frame containing

- AIC scores, AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores and logscore weights

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
# because it's too slow for CRAN
set.seed(2)
nx=100
predictor=c(1:nx)/nx
x=rnorm(nx,mean=predictor,sd=1)
print(ms_predictors_2tail(x,predictor))
```

nopdfcdfmsg

*Message to explain why GEV and GPD d*** and p*** routines don't return DMGS pdfs and cdfs*

Description

Message to explain why GEV and GPD d*** and p*** routines don't return DMGS pdfs and cdfs

Usage

```
nopdfcdfmsg(yy, pp)
```

Arguments

yy	vector of samples
pp	vector of probabilities

Value

String

norm_cp

*Normal Distribution Predictions Based on a Calibrating Prior***Description**

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qnorm_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  unbiasedv = FALSE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rnorm_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE, aderivs = TRUE)
```

```
dnorm_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)
```

```
pnorm_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)
```

```
tnorm_cp(n, x, debug = FALSE)
```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>d1</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the first parameter
<code>fd2</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the second parameter
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>waicscores</code>	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
<code>logscores</code>	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
<code>rust</code>	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
<code>nrust</code>	the number of posterior samples used in the RUST calculations
<code>unbiasedv</code>	logical for whether to include unbiased variance results in norm
<code>debug</code>	logical for turning on debug messages
<code>aderivs</code>	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
<code>n</code>	the number of random samples required
<code>mlcp</code>	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
<code>y</code>	a vector of values at which to calculate the density and distribution functions

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.

- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of x

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The normal distribution has probability density function

$$f(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d30norm_example_data_v1
p=c(1:9)/10
q=qnorm_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qnorm_cp)",
main="Normal: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

norm_dmgs_cp

Normal Distribution Predictions Based on a Calibrating Prior, using DMGS (for testing only)

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model **** the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```

qnorm_dmgs_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rnorm_dmgs_cp(
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dnorm_dmgs_cp(x, y = x, d1 = 0.01, fd2 = 0.01, debug = FALSE, aderivs = TRUE)

pnorm_dmgs_cp(x, y = x, d1 = 0.01, fd2 = 0.01, debug = FALSE, aderivs = TRUE)

```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>d1</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the first parameter
<code>fd2</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the second parameter
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>waicscores</code>	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
<code>logscores</code>	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
<code>dmgs</code>	logical that indicates whether DMGS calculations should be run or not (longer run time)
<code>debug</code>	logical for turning on debug messages

<code>aderivs</code>	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
<code>n</code>	the number of random samples required
<code>mlcp</code>	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
<code>y</code>	a vector of values at which to calculate the density and distribution functions

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The normal distribution has probability density function

$$f(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),

- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d30norm_example_data_v1
p=c(1:9)/10
q=qnorm_dmgs_cp(x,p)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qnorm_dmgs_cp)",
main="Normal_DMGS: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
```

norm_dmgs_loglik	<i>log-likelihood function</i>
------------------	--------------------------------

Description

log-likelihood function

Usage

```
norm_dmgs_loglik(vv, x)
```

Arguments

- vv parameters
- x a vector of training data values

Value

Scalar value.

norm_dmgs_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
---------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
norm_dmgs_logscores(logscores, x, d1 = 0.01, fd2 = 0.01)
```

Arguments

- logscores logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
- x a vector of training data values
- d1 the delta used in the numerical derivatives with respect to the parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Two scalars

norm_dmgs_means	<i>MLE and RHP predictive means</i>
-----------------	-------------------------------------

Description

MLE and RHP predictive means

Usage

```
norm_dmgs_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

norm_dmgs_mu1f	<i>DMGS equation 3.3, mu1 term</i>
----------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

norm_dmgs_mu1f(alpha, v1, d1, v2, fd2)

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

norm_dmgs_mu2f	<i>DMGS equation 3.3, mu2 term</i>
----------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

norm_dmgs_mu2f(alpha, v1, d1, v2, fd2)

Arguments

- | | |
|-------|--|
| alpha | a vector of values of alpha (one minus probability) |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

norm_dmgs_p1f	<i>DMGS equation 3.3, p1 term</i>
---------------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

norm_dmgs_p1f(y, v1, d1, v2, fd2)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

Matrix

<code>norm_dmgs_p2f</code>	<i>DMGS equation 3.3, p2 term</i>
----------------------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

```
norm_dmgs_p2f(y, v1, d1, v2, fd2)
```

Arguments

- `y` a vector of values at which to calculate the density and distribution functions
- `v1` first parameter
- `d1` the delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `fd2` the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

<code>norm_dmgs_waic</code>	<i>Waic</i>
-----------------------------	-------------

Description

Waic

Usage

```
norm_dmgs_waic(  
  waicscores,  
  x,  
  v1hat,  
  d1,  
  v2hat,  
  fd2,  
  lddi,
```

```
      lddd,  
      lambdad,  
      aderivs  
    )
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

norm_f1f	<i>DMGS equation 3.3, f1 term</i>
----------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

```
norm_f1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

norm_f1fa	<i>The first derivative of the density</i>
-----------	--

Description

The first derivative of the density

Usage

norm_f1fa(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Vector

norm_f2f	<i>DMGS equation 3.3, f2 term</i>
----------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

norm_f2f(y, v1, d1, v2, fd2)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

norm_f2fa	<i>The second derivative of the density</i>
-----------	---

Description

The second derivative of the density

Usage

```
norm_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

norm_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_gg	<i>Second derivative matrix of the expected per-observation log-likelihood</i>
---------	--

Description

Second derivative matrix of the expected per-observation log-likelihood

Usage

```
norm_gg(nx, v1, d1, v2, fd2)
```

Arguments

nx	length of training data
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

norm_gmn	<i>One component of the second derivative of the expected log-likelihood</i>
----------	--

Description

One component of the second derivative of the expected log-likelihood

Usage

```
norm_gmn(alpha, v1, d1, v2, fd2, mm, nn)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

norm_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
----------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
norm_ldd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

norm_ldda	<i>The second derivative of the normalized log-likelihood</i>
-----------	---

Description

The second derivative of the normalized log-likelihood

Usage

norm_ldda(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Matrix

norm_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
-----------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

norm_lddd(x, v1, d1, v2, fd2)

Arguments

- x a vector of training data values
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

norm_lddda	<i>The third derivative of the normalized log-likelihood</i>
------------	--

Description

The third derivative of the normalized log-likelihood

Usage

norm_lddda(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

3d array

norm_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
----------	--

Description

One component of the second derivative of the normalized log-likelihood
One component of the second derivative of the normalized log-likelihood

Usage

norm_lmn(x, v1, d1, v2, fd2, mm, nn)

norm_lmn(x, v1, d1, v2, fd2, mm, nn)

Arguments

- x a vector of training data values
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter
- mm an index for which derivative to calculate
- nn an index for which derivative to calculate

Value

Scalar value

<code>norm_lmp</code>	<i>One component of the second derivative of the normalized log-likelihood</i>
-----------------------	--

Description

One component of the second derivative of the normalized log-likelihood

One component of the third derivative of the normalized log-likelihood

Usage

```
norm_lmp(x, v1, d1, v2, fd2, mm, nn, rr)
```

```
norm_lmp(x, v1, d1, v2, fd2, mm, nn, rr)
```

Arguments

<code>x</code>	a vector of training data values
<code>v1</code>	first parameter
<code>d1</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>fd2</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>mm</code>	an index for which derivative to calculate
<code>nn</code>	an index for which derivative to calculate
<code>rr</code>	an index for which derivative to calculate

Value

Scalar value

norm_logf	<i>Logf for RUST</i>
-----------	----------------------

Description

Logf for RUST

Usage

```
norm_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

norm_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

norm_logfddd(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

3d array

norm_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
----------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

norm_logscores(logscores, x)

Arguments

- logscores logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
- x a vector of training data values

Value

Two scalars

norm_ml_params	<i>Maximum likelihood estimator</i>
----------------	-------------------------------------

Description

Maximum likelihood estimator

Usage

```
norm_ml_params(x)
```

Arguments

x	a vector of training data values
---	----------------------------------

Value

Scalar value.

norm_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

```
norm_mu1fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Vector

<code>norm_mu2fa</code>	<i>Minus the second derivative of the cdf, at alpha</i>
-------------------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

`norm_mu2fa(alpha, v1, v2)`

Arguments

- | | |
|--------------------|---|
| <code>alpha</code> | a vector of values of alpha (one minus probability) |
| <code>v1</code> | first parameter |
| <code>v2</code> | second parameter |

Value

Matrix

<code>norm_p1fa</code>	<i>The first derivative of the cdf</i>
------------------------	--

Description

The first derivative of the cdf

Usage

`norm_p1fa(x, v1, v2)`

Arguments

- | | |
|-----------------|----------------------------------|
| <code>x</code> | a vector of training data values |
| <code>v1</code> | first parameter |
| <code>v2</code> | second parameter |

Value

Vector

norm_p1_cp

*Normal Distribution with a Predictor, Predictions Based on a Calibrating Prior***Description**

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  centering = TRUE,
```

```
    debug = FALSE,
    aderivs = TRUE
  )

  rnorm_p1_cp(
    n,
    x,
    t,
    t0 = NA,
    n0 = NA,
    rust = FALSE,
    mlcp = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  dnorm_p1_cp(
    x,
    t,
    t0 = NA,
    n0 = NA,
    y = x,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  pnorm_p1_cp(
    x,
    t,
    t0 = NA,
    n0 = NA,
    y = x,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  tnorm_p1_cp(n, x, t, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that <code>length(t)=length(x)</code>

t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.

- `adjustedx`: the detrended values of x

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The normal distribution with a predictor has probability density function

$$f(x; a, b, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu(a,b))^2/(2\sigma^2)}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, modelled as a function of parameters a, b and predictor t , and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b, \sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `retest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d60norm_p1_example_data_v1_x
tt=fitdistcp::d60norm_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qnorm_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qnorm_p1_cp)",
main="Normal w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

norm_p1_f1f	<i>DMGS equation 2.1, f1 term</i>
-------------	-----------------------------------

Description

DMGS equation 2.1, f1 term

Usage

```
norm_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

- y a vector of values at which to calculate the density and distribution functions
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- v3 third parameter
- fd3 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

norm_p1_f1fa	<i>The first derivative of the density</i>
--------------	--

Description

The first derivative of the density
The first derivative of the density

Usage

norm_p1_f1fa(x, t, v1, v2, v3)
norm_p1_f1fa(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

norm_p1_f2f	<i>DMGS equation 2.1, f2 term</i>
-------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

norm_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

norm_p1_f2fa	<i>The second derivative of the density</i>
--------------	---

Description

The second derivative of the density
The second derivative of the density

Usage

norm_p1_f2fa(x, t, v1, v2, v3)
norm_p1_f2fa(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_fd(x, t, v1, v2, v3)
```

```
norm_p1_fd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

norm_p1_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_fdd(x, t, v1, v2, v3)
```

```
norm_p1_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
-------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
norm_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

norm_p1_ldda	<i>The second derivative of the normalized log-likelihood</i>
--------------	---

Description

The second derivative of the normalized log-likelihood
The second derivative of the normalized log-likelihood

Usage

norm_p1_ldda(x, t, v1, v2, v3)
norm_p1_ldda(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
--------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

norm_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

norm_p1_lddda	<i>The third derivative of the normalized log-likelihood</i>
---------------	--

Description

The third derivative of the normalized log-likelihood
The third derivative of the normalized log-likelihood

Usage

norm_p1_lddda(x, t, v1, v2, v3)
norm_p1_lddda(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

norm_p1_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
-------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
norm_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

norm_p1_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
-------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
norm_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

norm_p1_logf	<i>Logf for RUST</i>
--------------	----------------------

Description

Logf for RUST

Usage

norm_p1_logf(params, x, t)

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

norm_p1_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_logfdd(x, t, v1, v2, v3)
```

```
norm_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_logfddd(x, t, v1, v2, v3)
```

```
norm_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

norm_p1_loglik	<i>Normal-with-p1 observed log-likelihood function</i>
----------------	--

Description

Normal-with-p1 observed log-likelihood function

Usage

```
norm_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

norm_p1_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
-------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
norm_p1_logscores(logscores, x, t, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

norm_p1_mlparams	<i>Maximum likelihood estimator</i>
------------------	-------------------------------------

Description

Maximum likelihood estimator

Usage

```
norm_p1_mlparams(x, t)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors

Value

Vector

norm_p1_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
---------------	--

Description

Minus the first derivative of the cdf, at alpha
Minus the first derivative of the cdf, at alpha

Usage

norm_p1_mu1fa(alpha, t, v1, v2, v3)

norm_p1_mu1fa(alpha, t, v1, v2, v3)

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

norm_p1_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
---------------	---

Description

Minus the second derivative of the cdf, at alpha
Minus the second derivative of the cdf, at alpha

Usage

norm_p1_mu2fa(alpha, t, v1, v2, v3)

norm_p1_mu2fa(alpha, t, v1, v2, v3)

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_p1fa	<i>The first derivative of the cdf</i>
--------------	--

Description

The first derivative of the cdf
The first derivative of the cdf

Usage

norm_p1_p1fa(x, t, v1, v2, v3)
norm_p1_p1fa(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

norm_p1_p2fa	<i>The second derivative of the cdf</i>
--------------	---

Description

The second derivative of the cdf
The second derivative of the cdf

Usage

norm_p1_p2fa(x, t, v1, v2, v3)

norm_p1_p2fa(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol
First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

norm_p1_pd(x, t, v1, v2, v3)

norm_p1_pd(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

norm_p1_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

norm_p1_pdd(x, t, v1, v2, v3)

norm_p1_pdd(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_predictordata	<i>Predicted Parameter and Generalized Residuals</i>
-----------------------	--

Description

Predicted Parameter and Generalized Residuals

Usage

```
norm_p1_predictordata(x, t, t0, params)
```

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- params model parameters for calculating logf

Value

Two vectors

norm_p1_waic	<i>Waic</i>
--------------	-------------

Description

Waic

Usage

```
norm_p1_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  v3hat,  
  fd3,  
  aderivs = TRUE  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

norm_p2fa	<i>The second derivative of the cdf</i>
-----------	---

Description

The second derivative of the cdf

Usage

norm_p2fa(x, v1, v2)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

norm_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_unbiasedv_params	<i>Method of moments estimator</i>
-----------------------	------------------------------------

Description

Method of moments estimator

Usage

norm_unbiasedv_params(x)

Arguments

x a vector of training data values

Value

Vector

norm_waic	<i>Waic</i>
-----------	-------------

Description

Waic

Usage

norm_waic(waiccores, x, v1hat, d1, v2hat, fd2, aderivs)

Arguments

waiccores logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)

x a vector of training data values

v1hat first parameter

d1 the delta used in the numerical derivatives with respect to the parameter

v2hat second parameter

fd2 the fractional delta used in the numerical derivatives with respect to the parameter

aderivs logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qpareto_k2_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kscale = 1,
  fd1 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rpareto_k2_cp(
  n,
```

```

    x,
    kscale = 1,
    rust = FALSE,
    mlcp = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

dpareto_k2_cp(
  x,
  y = x,
  kscale = 1,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

ppareto_k2_cp(
  x,
  y = x,
  kscale = 1,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tpareto_k2_cp(n, x, kscale = 1, debug = FALSE)

```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>kscale</code>	the known scale parameter
<code>fd1</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the first parameter
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>waicscores</code>	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
<code>logscores</code>	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
<code>rust</code>	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
<code>nrust</code>	the number of posterior samples used in the RUST calculations

debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times $-1/2$.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.

- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Pareto distribution has various forms. The form we are using has exceedance distribution function

$$S(x; \alpha) = \left(\frac{\sigma}{x}\right)^\alpha$$

where $x \geq \sigma$ is the random variable and $\alpha > 0, \sigma > 0$ are the shape and scale parameters. We consider the scale parameter σ to be known (hence the k2 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\alpha) \propto \frac{1}{\alpha}$$

as given in Jewson et al. (2025). Some others authors may refer to the shape and scale parameters as the scale and location parameters, respectively.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

p**** optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),

- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d11pareto_k2_example_data_v1
p=c(1:9)/10
q=qpareto_k2_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles)
xmax=max(q$ml_quantiles,q$cp_quantiles)
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qpareto_k2_cp)",
main="Pareto: quantile estimates")
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

pareto_k2_f1f

DMGS equation 2.1, f1 term

Description

DMGS equation 2.1, f1 term

Usage

```
pareto_k2_f1f(y, v1, fd1, kscale)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

Matrix

pareto_k2_f1fa	<i>The first derivative of the density</i>
----------------	--

Description

The first derivative of the density

Usage

pareto_k2_f1fa(x, v1, kscale)

Arguments

x	a vector of training data values
v1	first parameter
kscale	the known scale parameter

Value

Vector

pareto_k2_f2f	<i>DMGS equation 2.1, f2 term</i>
---------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

pareto_k2_f2f(y, v1, fd1, kscale)

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

3d array

pareto_k2_f2fa	<i>The second derivative of the density</i>
----------------	---

Description

The second derivative of the density

Usage

pareto_k2_f2fa(x, v1, kscale)

Arguments

x	a vector of training data values
v1	first parameter
kscale	the known scale parameter

Value

Matrix

pareto_k2_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

pareto_k2_fd(x, v1, v2)

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

pareto_k2_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

pareto_k2_l111	<i>Third derivative of the normalized log-likelihood</i>
----------------	--

Description

Third derivative of the normalized log-likelihood

Usage

```
pareto_k2_l111(x, v1, fd1, kscale)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

Scalar value

pareto_k2_ldd	<i>The second derivative of the normalized log-likelihood</i>
---------------	---

Description

The second derivative of the normalized log-likelihood

Usage

pareto_k2_ldd(x, v1, fd1, kscale)

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

Square scalar matrix

pareto_k2_ldda	<i>The second derivative of the normalized log-likelihood</i>
----------------	---

Description

The second derivative of the normalized log-likelihood

Usage

pareto_k2_ldda(x, v1, kscale)

Arguments

- x a vector of training data values
- v1 first parameter
- kscale the known scale parameter

Value

Matrix

pareto_k2_lddd	<i>Third derivative tensor of the log-likelihood</i>
----------------	--

Description

Third derivative tensor of the log-likelihood

Usage

pareto_k2_lddd(x, v1, fd1, kscale)

Arguments

- x a vector of training data values
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- kscale the known scale parameter

Value

Cubic scalar array

pareto_k2_lddda	<i>The third derivative of the normalized log-likelihood</i>
-----------------	--

Description

The third derivative of the normalized log-likelihood

Usage

pareto_k2_lddda(x, v1, kscale)

Arguments

- x a vector of training data values
- v1 first parameter
- kscale the known scale parameter

Value

3d array

pareto_k2_logf	<i>Logf for RUST</i>
----------------	----------------------

Description

Logf for RUST

Usage

pareto_k2_logf(params, x, kscale)

Arguments

- params model parameters for calculating logf
- x a vector of training data values
- kscale the known scale parameter

Value

Scalar value.

pareto_k2_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

pareto_k2_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

pareto_k2_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
---------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

pareto_k2_logscores(logscores, x, kscale)

Arguments

- | | |
|-----------|---|
| logscores | logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime) |
| x | a vector of training data values |
| kscale | the known scale parameter |

Value

Two scalars

pareto_k2_ml_params	<i>Maximum likelihood estimator</i>
---------------------	-------------------------------------

Description

Maximum likelihood estimator

Usage

pareto_k2_ml_params(x, kscale)

Arguments

- | | |
|--------|----------------------------------|
| x | a vector of training data values |
| kscale | the known scale parameter |

Value

Scalar value.

pareto_k2_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
-----------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

pareto_k2_mu1fa(alpha, v1, kscale)

Arguments

- alpha a vector of values of alpha (one minus probability)
- v1 first parameter
- kscale the known scale parameter

Value

Vector

pareto_k2_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
-----------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

pareto_k2_mu2fa(alpha, v1, kscale)

Arguments

- alpha a vector of values of alpha (one minus probability)
- v1 first parameter
- kscale the known scale parameter

Value

Matrix

pareto_k2_p1fa	<i>The first derivative of the cdf</i>
----------------	--

Description

The first derivative of the cdf

Usage

pareto_k2_p1fa(x, v1, kscale)

Arguments

- x a vector of training data values
- v1 first parameter
- kscale the known scale parameter

Value

Vector

pareto_k2_p2fa	<i>The second derivative of the cdf</i>
----------------	---

Description

The second derivative of the cdf

Usage

pareto_k2_p2fa(x, v1, kscale)

Arguments

- x a vector of training data values
- v1 first parameter
- kscale the known scale parameter

Value

Matrix

pareto_k2_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

pareto_k2_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

pareto_k2_waic	<i>Waic</i>
----------------	-------------

Description

Waic

Usage

pareto_k2_waic(waiccores, x, v1hat, fd1, kscale, aderivs)

Arguments

waiccores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

pareto_p1k2_cp	<i>Pareto Distribution with a Predictor, Predictions Based on a Calibrating Prior</i>
----------------	---

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.

- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qpareto_p1k2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  kscale = 1,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rpareto_p1k2_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  kscale = 1,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```

)

dpareto_p1k2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  kscale = 1,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

ppareto_p1k2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  kscale = 1,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tpareto_p1k2_cp(n, x, t, d1 = 0.01, d2 = 0.01, kscale = 1, debug = FALSE)

```

Arguments

x	a vector of training data values
t	a vector of predictors, such that <code>length(t)=length(x)</code>
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter

kscale	the known scale parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times $-1/2$.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.

- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Pareto distribution with a predictor has various forms. The form we are using has exceedance distribution function

$$S(x; a, b) = \left(\frac{\sigma}{x}\right)^{\alpha(a, b)}$$

where $x \geq \sigma$ is the random variable, $\alpha = \exp(-a - bt)$ is the shape parameter, modelled as a function of parameters a, b , and σ is the scale parameter. We consider the scale parameter σ to be known (hence the k2 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b) \propto 1$$

as given in Jewson et al. (2025). Note that others authors have referred to the shape and scale parameters as the scale and location parameters, respectively.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d56pareto_p1k2_example_data_v1_x
tt=fitdistcp::d56pareto_p1k2_example_data_v1_t
p=c(1:9)/10
n0=10
```

```
q=qpareto_p1k2_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qpareto_p1k2_cp)",
main="Pareto w/ p2: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

pareto_p1k2_f1f	<i>DMGS equation 2.1, f1 term</i>
-----------------	-----------------------------------

Description

DMGS equation 2.1, f1 term

Usage

pareto_p1k2_f1f(y, t0, v1, d1, v2, d2, kscale)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- kscale the known scale parameter

Value

Matrix

pareto_p1k2_f1fa	<i>The first derivative of the density</i>
------------------	--

Description

The first derivative of the density

Usage

```
pareto_p1k2_f1fa(x, t, v1, v2, kscale)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Vector

pareto_p1k2_f2f	<i>DMGS equation 2.1, f2 term</i>
-----------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

```
pareto_p1k2_f2f(y, t0, v1, d1, v2, d2, kscale)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

3d array

pareto_p1k2_f2fa	<i>The second derivative of the density</i>
------------------	---

Description

The second derivative of the density

Usage

pareto_p1k2_f2fa(x, t, v1, v2, kscale)

Arguments

- | | |
|--------|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| kscale | the known scale parameter |

Value

Matrix

pareto_p1k2_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

pareto_p1k2_fd(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Vector

pareto_p1k2_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

pareto_p1k2_1dd	<i>Second derivative matrix of the normalized log-likelihood</i>
-----------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
pareto_p1k2_1dd(x, t, v1, d1, v2, d2, kscale)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

Square scalar matrix

pareto_p1k2_ldda	<i>The second derivative of the normalized log-likelihood</i>
------------------	---

Description

The second derivative of the normalized log-likelihood

Usage

pareto_p1k2_ldda(x, t, v1, v2, kscale)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Matrix

pareto_p1k2_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
------------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

pareto_p1k2_lddd(x, t, v1, d1, v2, d2, kscale)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- kscale the known scale parameter

Value

Cubic scalar array

pareto_p1k2_lddda	<i>The third derivative of the normalized log-likelihood</i>
-------------------	--

Description

The third derivative of the normalized log-likelihood

Usage

pareto_p1k2_lddda(x, t, v1, v2, kscale)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- kscale the known scale parameter

Value

3d array

pareto_p1k2_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
-----------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

pareto_p1k2_lmn(x, t, v1, d1, v2, d2, kscale, mm, nn)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- kscale the known scale parameter
- mm an index for which derivative to calculate
- nn an index for which derivative to calculate

Value

Scalar value

pareto_p1k2_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
-----------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

pareto_p1k2_lmn(x, t, v1, d1, v2, d2, kscale, mm, nn, rr)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

pareto_p1k2_logf	<i>Logf for RUST</i>
------------------	----------------------

Description

Logf for RUST

Usage

pareto_p1k2_logf(params, x, t, kscale)

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors
kscale	the known scale parameter

Value

Scalar value.

pareto_p1k2_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

pareto_p1k2_logfdd(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Matrix

pareto_p1k2_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

pareto_p1k2_logfddd(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

3d array

pareto_p1k2_loglik	<i>observed log-likelihood function</i>
--------------------	---

Description

observed log-likelihood function

Usage

pareto_p1k2_loglik(vv, x, t, kscale)

Arguments

- | | |
|--------|----------------------------------|
| vv | parameters |
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| kscale | the known scale parameter |

Value

Scalar value.

pareto_p1k2_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
-----------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

pareto_p1k2_logscores(logscores, x, t, d1, d2, kscale, aderivs, debug)

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)
debug	debug flag

Value

Two scalars

pareto_p1k2_means	<i>pareto_k1 distribution: RHP mean</i>
-------------------	---

Description

pareto_k1 distribution: RHP mean

Usage

```
pareto_p1k2_means(  
  means,  
  t0,  
  ml_params,  
  lddi,  
  lddd,  
  lambdad_rhp,  
  nx,  
  dim = 2,  
  kscale  
)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix

lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kscale	the known scale parameter

Value

Two scalars

pareto_p1k2_mu1f	<i>DMGS equation 3.3, mu1 term</i>
------------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

pareto_p1k2_mu1f(alpha, t0, v1, d1, v2, d2, kscale)

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

Matrix

pareto_p1k2_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
-------------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

pareto_p1k2_mu1fa(alpha, t, v1, v2, kscale)

Arguments

- | | |
|--------|---|
| alpha | a vector of values of alpha (one minus probability) |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| kscale | the known scale parameter |

Value

Vector

pareto_p1k2_mu2f	<i>DMGS equation 3.3, mu2 term</i>
------------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

pareto_p1k2_mu2f(alpha, t0, v1, d1, v2, d2, kscale)

Arguments

- | | |
|--------|---|
| alpha | a vector of values of alpha (one minus probability) |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| kscale | the known scale parameter |

Value

3d array

pareto_p1k2_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
-------------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

pareto_p1k2_mu2fa(alpha, t, v1, v2, kscale)

Arguments

- | | |
|--------|---|
| alpha | a vector of values of alpha (one minus probability) |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| kscale | the known scale parameter |

Value

Matrix

pareto_p1k2_p1f	<i>DMGS equation 2.1, p1 term</i>
-----------------	-----------------------------------

Description

DMGS equation 2.1, p1 term

Usage

pareto_p1k2_p1f(y, t0, v1, d1, v2, d2, kscale)

Arguments

- | | |
|--------|---|
| y | a vector of values at which to calculate the density and distribution functions |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| v1 | first parameter |
| d1 | the delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| kscale | the known scale parameter |

Value

Matrix

pareto_p1k2_p1fa	<i>The first derivative of the cdf</i>
------------------	--

Description

The first derivative of the cdf

Usage

pareto_p1k2_p1fa(x, t, v1, v2, kscale)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- kscale the known scale parameter

Value

Vector

pareto_p1k2_p2f	<i>DMGS equation 2.1, p2 term</i>
-----------------	-----------------------------------

Description

DMGS equation 2.1, p2 term

Usage

pareto_p1k2_p2f(y, t0, v1, d1, v2, d2, kscale)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- v1 first parameter
- d1 the delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- kscale the known scale parameter

Value

3d array

pareto_p1k2_p2fa	<i>The second derivative of the cdf</i>
------------------	---

Description

The second derivative of the cdf

Usage

pareto_p1k2_p2fa(x, t, v1, v2, kscale)

Arguments

- | | |
|--------|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| kscale | the known scale parameter |

Value

Matrix

pareto_p1k2_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

pareto_p1k2_pd(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Vector

pareto_p1k2_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

pareto_p1k2_pdd(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Matrix

pareto_p1k2_predictordata	<i>Predicted Parameter and Generalized Residuals</i>
---------------------------	--

Description

Predicted Parameter and Generalized Residuals

Usage

pareto_p1k2_predictordata(predictordata, x, t, t0, params, kscale)

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf
kscale	the known scale parameter

Value

Two vectors

pareto_p1k2_waic	<i>Waic</i>
------------------	-------------

Description

Waic

Usage

```
pareto_p1k2_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  d1,  
  v2hat,  
  d2,  
  kscale,  
  lddi,  
  lddd,  
  lambdad  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter

d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

pcauchy_p1	<i>Cauchy-with-p1 distribution function</i>
------------	---

Description

Cauchy-with-p1 distribution function

Usage

pcauchy_p1(x, t0, ymn, slope, scale)

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution

Value

Vector

pexp_p1	<i>Exponential-with-p1 distribution function</i>
---------	--

Description

Exponential-with-p1 distribution function

Usage

```
pexp_p1(x, t0, ymn, slope)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor

Value

Vector

pfrechet_p2k1	<i>Frechet_k1-with-p2 distribution function</i>
---------------	---

Description

Frechet_k1-with-p2 distribution function

Usage

```
pfrechet_p2k1(x, t0, ymn, slope, lambda, kloc)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
lambda	the lambda parameter of the distribution
kloc	the known location parameter

Value

Vector

pgev_p1	<i>GEVD-with-p1: Distribution function</i>
---------	--

Description

GEVD-with-p1: Distribution function

Usage

```
pgev_p1(y, t0, ymn, slope, sigma, xi)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution

Value

Vector

pgev_p12	<i>GEVD-with-p1: Distribution function</i>
----------	--

Description

GEVD-with-p1: Distribution function

Usage

```
pgev_p12(y, t1, t2, ymn, slope, sigma1, sigma2, xi)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi	the shape parameter of the distribution

Value

Vector

pgev_p123	<i>GEVD-with-p1: Distribution function</i>
-----------	--

Description

GEVD-with-p1: Distribution function

Usage

pgev_p123(y, t1, t2, t3, ymn, slope, sigma1, sigma2, xi1, xi2)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution

Value

Vector

<code>pgev_p1k3</code>	<i>GEV-with-known-shape-with-p1 distribution function</i>
------------------------	---

Description

GEV-with-known-shape-with-p1 distribution function

Usage

`pgev_p1k3(x, t0, ymn, slope, sigma, kshape)`

Arguments

- | | |
|---------------------|--|
| <code>x</code> | a vector of training data values |
| <code>t0</code> | a single value of the predictor (specify either <code>t0</code> or <code>n0</code> but not both) |
| <code>ymn</code> | the location parameter of the function of the predictor |
| <code>slope</code> | the slope of the function of the predictor |
| <code>sigma</code> | the sigma parameter of the distribution |
| <code>kshape</code> | the known shape parameter |

Value

Vector

<code>pgumbel_p1</code>	<i>Gumbel-with-p1 distribution function</i>
-------------------------	---

Description

Gumbel-with-p1 distribution function

Usage

`pgumbel_p1(x, t0, ymn, slope, sigma)`

Arguments

- | | |
|--------------------|--|
| <code>x</code> | a vector of training data values |
| <code>t0</code> | a single value of the predictor (specify either <code>t0</code> or <code>n0</code> but not both) |
| <code>ymn</code> | the location parameter of the function of the predictor |
| <code>slope</code> | the slope of the function of the predictor |
| <code>sigma</code> | the sigma parameter of the distribution |

Value

Vector

plnorm_p1	<i>Normal-with-p1 distribution function</i>
-----------	---

Description

Normal-with-p1 distribution function

Usage

```
plnorm_p1(x, t0, ymn, slope, sigma)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

plogis_p1	<i>Logistic-with-p1 distribution function</i>
-----------	---

Description

Logistic-with-p1 distribution function

Usage

```
plogis_p1(x, t0, ymn, slope, scale)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution

Value

Vector

plst_p1k3	<i>LST-with-p1 distribution function</i>
-----------	--

Description

LST-with-p1 distribution function

Usage

```
plst_p1k3(x, t0, ymn, slope, sigma, kdf)
```

Arguments

- | | |
|-------|--|
| x | a vector of training data values |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| ymn | the location parameter of the function of the predictor |
| slope | the slope of the function of the predictor |
| sigma | the sigma parameter of the distribution |
| kdf | the known degrees of freedom parameter |

Value

Vector

pnorm_p1	<i>Normal-with-p1 distribution function</i>
----------	---

Description

Normal-with-p1 distribution function

Usage

```
pnorm_p1(x, t0, ymn, slope, sigma)
```

Arguments

- | | |
|-------|--|
| x | a vector of training data values |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| ymn | the location parameter of the function of the predictor |
| slope | the slope of the function of the predictor |
| sigma | the sigma parameter of the distribution |

Value

Vector

pnorm_p1_formula	<i>Linear regression formula, densities</i>
------------------	---

Description

Linear regression formula, densities

Usage

```
pnorm_p1_formula(y, ta, ta0, nx, muhat0, v3hat)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
ta	predictor residuals
ta0	predictor residual at the point being predicted
nx	length of training data
muhat0	muhat at the point being predicted
v3hat	third parameter

Value

Vector

ppareto_p1k2	<i>pareto_k1-with-p2 distribution function</i>
--------------	--

Description

pareto_k1-with-p2 distribution function

Usage

```
ppareto_p1k2(x, t0, ymn, slope, kscale)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
kscale	the known scale parameter

Value

Vector

<code>punif_formula</code>	<i>Predictive CDFs</i>
----------------------------	------------------------

Description

Predictive CDFs

Usage

```
punif_formula(x, y)
```

Arguments

- `x` a vector of training data values
- `y` a vector of values at which to calculate the density and distribution functions

Value

Two vectors

<code>pweibull_p2</code>	<i>Weibull-with-p1 distribution function</i>
--------------------------	--

Description

Weibull-with-p1 distribution function

Usage

```
pweibull_p2(x, t0, shape, ymn, slope)
```

Arguments

- `x` a vector of training data values
- `t0` a single value of the predictor (specify either `t0` or `n0` but not both)
- `shape` the shape parameter of the distribution
- `ymn` the location parameter of the function of the predictor
- `slope` the slope of the function of the predictor

Value

Vector

qcauchy_p1	<i>Cauchy-with-p1 quantile function</i>
------------	---

Description

Cauchy-with-p1 quantile function

Usage

qcauchy_p1(p, t0, ymn, slope, scale)

Arguments

- p a vector of probabilities at which to generate predictive quantiles
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- ymn the location parameter of the function of the predictor
- slope the slope of the function of the predictor
- scale the scale parameter of the distribution

Value

Vector

qexp_p1	<i>-with-p1 quantile function</i>
---------	-----------------------------------

Description

-with-p1 quantile function

Usage

qexp_p1(p, t0, ymn, slope)

Arguments

- p a vector of probabilities at which to generate predictive quantiles
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- ymn the location parameter of the function of the predictor
- slope the slope of the function of the predictor

Value

Vector

qfrechet_p2k1	<i>Frechet_k1-with-p2 quantile function</i>
---------------	---

Description

Frechet_k1-with-p2 quantile function

Usage

qfrechet_p2k1(p, t0, ymn, slope, lambda, kloc)

Arguments

- p a vector of probabilities at which to generate predictive quantiles
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- ymn the location parameter of the function of the predictor
- slope the slope of the function of the predictor
- lambda the lambda parameter of the distribution
- kloc the known location parameter

Value

Vector

qgamma_k1_ppm	<i>Temporary dummy for one of the cp models</i>
---------------	---

Description

Temporary dummy for one of the cp models

Usage

qgamma_k1_ppm(x, p)

Arguments

- x a vector of training data values
- p a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

qgamma_ppm

Temporary dummy for one of the ppm models

Description

Temporary dummy for one of the ppm models

Usage

```
qgamma_ppm(x, p)
```

Arguments

x a vector of training data values
p a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.

- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

qgev_k12_ppm

Temporary dummy for one of the ppm models

Description

Temporary dummy for one of the ppm models

Usage

```
qgev_k12_ppm(x, p)
```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times -1/2.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

qgev_mpd_ppm	<i>Temporary dummy for one of the ppm models</i>
--------------	--

Description

Temporary dummy for one of the ppm models

Usage

qgev_mpd_ppm(x, p)

Arguments

- | | |
|---|---|
| x | a vector of training data values |
| p | a vector of probabilities at which to generate predictive quantiles |

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

qgev_p1	<i>GEVD-with-p1: Quantile function</i>
---------	--

Description

GEVD-with-p1: Quantile function

Usage

```
qgev_p1(p, t0, ymn, slope, sigma, xi)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution

Value

Vector

qgev_p12	<i>GEVD-with-p1: Quantile function</i>
----------	--

Description

GEVD-with-p1: Quantile function

Usage

```
qgev_p12(p, t1, t2, ymn, slope, sigma1, sigma2, xi)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi	the shape parameter of the distribution

Value

Vector

qgev_p123	<i>GEVD-with-p1: Quantile function</i>
-----------	--

Description

GEVD-with-p1: Quantile function

Usage

```
qgev_p123(p, t1, t2, t3, ymn, slope, sigma1, sigma2, xi1, xi2)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution

Value

Vector

qgev_p1k3	<i>GEV-with-known-shape-with-p1 quantile function</i>
-----------	---

Description

GEV-with-known-shape-with-p1 quantile function

Usage

qgev_p1k3(p, t0, ymn, slope, sigma, kshape)

Arguments

- | | |
|--------|--|
| p | a vector of probabilities at which to generate predictive quantiles |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| ymn | the location parameter of the function of the predictor |
| slope | the slope of the function of the predictor |
| sigma | the sigma parameter of the distribution |
| kshape | the known shape parameter |

Value

Vector

qgev_p1_ppm	<i>Temporary dummy for one of the ppm models</i>
-------------	--

Description

Temporary dummy for one of the ppm models

Usage

qgev_p1_ppm(x, t, n0, p)

Arguments

- | | |
|----|---|
| x | a vector of training data values |
| t | a vector of predictors, such that length(t)=length(x) |
| n0 | an index for the predictor (specify either t0 or n0 but not both) |
| p | a vector of probabilities at which to generate predictive quantiles |

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

qgev_ppm

Temporary dummy for one of the ppm models

Description

Temporary dummy for one of the ppm models

Usage

```
qgev_ppm(x, p)
```

Arguments

x a vector of training data values
p a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.

- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

qgpd_k1_ppm

Temporary dummy for one of the ppm models

Description

Temporary dummy for one of the ppm models

Usage

```
qgpd_k1_ppm(x, p)
```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times -1/2.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

qgumbel_p1

Gumbel-with-p1 quantile function

Description

Gumbel-with-p1 quantile function

Usage

```
qgumbel_p1(p, t0, ymn, slope, sigma)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

qlnorm_p1	<i>Normal-with-p1 quantile function</i>
-----------	---

Description

Normal-with-p1 quantile function

Usage

qlnorm_p1(p, t0, ymn, slope, sigma)

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

qlogis_p1	<i>Logistic-with-p1 quantile function</i>
-----------	---

Description

Logistic-with-p1 quantile function

Usage

```
qlogis_p1(p, t0, ymn, slope, scale)
```

Arguments

- | | |
|-------|--|
| p | a vector of probabilities at which to generate predictive quantiles |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| ymn | the location parameter of the function of the predictor |
| slope | the slope of the function of the predictor |
| scale | the scale parameter of the distribution |

Value

Vector

qlst_p1k3	<i>LST-with-p1 quantile function</i>
-----------	--------------------------------------

Description

LST-with-p1 quantile function

Usage

```
qlst_p1k3(p, t0, ymn, slope, sigma, kdf)
```

Arguments

- | | |
|-------|--|
| p | a vector of probabilities at which to generate predictive quantiles |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| ymn | the location parameter of the function of the predictor |
| slope | the slope of the function of the predictor |
| sigma | the sigma parameter of the distribution |
| kdf | the known degrees of freedom parameter |

Value

Vector

qnorm_p1	<i>Normal-with-p1 quantile function</i>
----------	---

Description

Normal-with-p1 quantile function

Usage

qnorm_p1(p, t0, ymn, slope, sigma)

Arguments

- | | |
|-------|--|
| p | a vector of probabilities at which to generate predictive quantiles |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| ymn | the location parameter of the function of the predictor |
| slope | the slope of the function of the predictor |
| sigma | the sigma parameter of the distribution |

Value

Vector

qnorm_p1_formula	<i>Linear regression formula, quantiles</i>
------------------	---

Description

Linear regression formula, quantiles

Usage

qnorm_p1_formula(alpha, ta, ta0, nx, muhat0, v3hat)

Arguments

- | | |
|--------|---|
| alpha | a vector of values of alpha (one minus probability) |
| ta | predictor residuals |
| ta0 | predictor residual at the point being predicted |
| nx | length of training data |
| muhat0 | muhat at the point being predicted |
| v3hat | third parameter |

Value

Vector

qntt_ppm

Temporary dummy for one of the ppm models

Description

Temporary dummy for one of the ppm models

Usage

```
qntt_ppm(x, p)
```

Arguments

x a vector of training data values
p a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.

- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

qpareto_p1k2	<i>pareto_k1-with-p2 quantile function</i>
--------------	--

Description

pareto_k1-with-p2 quantile function

Usage

qpareto_p1k2(p, t0, ymn, slope, kscale)

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
kscale	the known scale parameter

Value

Vector

qunif_formula	<i>Predictive Quantiles</i>
---------------	-----------------------------

Description

Predictive Quantiles

Usage

```
qunif_formula(x, p)
```

Arguments

- x a vector of training data values
- p a vector of probabilities at which to generate predictive quantiles

Value

Two vectors

qweibull_p2	<i>Weibull-with-p1 quantile function</i>
-------------	--

Description

Weibull-with-p1 quantile function

Usage

```
qweibull_p2(p, t0, shape, ymn, slope)
```

Arguments

- p a vector of probabilities at which to generate predictive quantiles
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- shape the shape parameter of the distribution
- ymn the location parameter of the function of the predictor
- slope the slope of the function of the predictor

Value

Vector

reltest

*Evaluation of Reliability for Models in the fitdistcp Package***Description**

Uses simulations to evaluate the reliability of the predictive quantiles produced by the q****_cp routines in the fitdistcp package.

Usage

```
reltest(
  model = "exp",
  ntrials = 1000,
  nrepeats = 3,
  nx = 20,
  params = c(1),
  alpha = seq(0.005, 0.995, 0.005),
  plotflag = TRUE,
  verbose = TRUE,
  dmgs = TRUE,
  debug = FALSE,
  aderivs = TRUE,
  unbiasedv = FALSE,
  pwm = FALSE,
  minxi = -10,
  maxx = 10
)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k1", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k3", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k4", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k4", "norm_p12", "lst_p12k5", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1", "gev_p12", "gev_p123".
ntrials	the number of trials to run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence. 3 is a good choice.
nx	the length of the training data to use.
params	values for the parameters for the specified distribution
alpha	the exceedance probability values at which to test
plotflag	logical to turn the plotting on and off
verbose	logical to turn loop counting on and off

dmgs	logical to turn DMGS calculations on and off (to optimize speed for maxlik only calculations)
debug	logical for turning debug messages on and off
aderivs	logical for whether to use analytic derivatives (instead of numerical)
unbiasedv	logical for whether to use the unbiased variance instead of maxlik (for the normal)
pwm	logical for whether to use PWM instead of maxlik (for the GEV)
minxi	minimum value for EVT shape parameter
maxxi	maximum value for EVT shape parameter

Details

The maximum likelihood quantiles (plotted in blue) do not give good reliability. They typically underestimate the tails (see panel (f)).

For "exp", "pareto_k1", "unif", "norm", "lnorm", "norm_p1" and "lnorm_p1", the calibrating prior quantiles are calculated using the right Haar prior and an exact solution for the Bayesian prediction integral. They will converge towards exact reliability with a large enough number of trials, for any sample size.

For "halfnorm", "norm_dmgs", "lnorm_dmgs", "gnorm_k3", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k3", "gumbel_p1", "logis_p1" and "lst_p1k4" "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k4", "norm_p12", "lst_p12k5" the calibrating prior quantiles are calculated using the right Haar prior, with the DMGS asymptotic solution for the Bayesian prediction integral. They will converge towards good reliability with a large enough number of trials, with the only deviation from exact reliability being due to the neglect of higher order terms in the asymptotic expansion. They will converge towards exact reliability with a large enough number of trials and a large enough sample size.

For "gamma", "invgamma", "invgauss", "gev", "gpd_k1" and "gev_p1", "gev_p12", "gev_p123", the calibrating prior quantiles are calculated using the "fitdistcp" recommended calibrating priors, with the DMGS asymptotic solution for the Bayesian prediction integral. The chosen priors give reasonably good reliability with a large enough number of trials, and for large sample sizes, but may give poor reliability for small sample sizes (e.g., $n < 20$).

Value

A plot showing 9 different reliability checks, and a list containing various outputs, including the probabilities shown in the plot.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`1st_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`1st_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),

- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
set.seed(1)
# example 1
# -runs the default settings, which test reliability for the exponential distribution
reltest()
```

reltest2

*Evaluation of Reliability for Certain Additional Models in the
fitdistcp Package*

Description

This routine is mainly for reproducing certain results in Jewson et al. (2025), and not of general interest.

It uses simulations to evaluate the reliability of the predictive quantiles produced by the `qgev_cp`, `ggpd_cp` and `qgev_p1_cp` routines in the `fitdistcp` package. For each model, results for 5 models are calculated. This is to illustrate that the calibrating prior predictions dominate the `ml`, `flat`, `crhp_ml` and `jp` predictions, in terms of reliability.

Usage

```
reltest2(
  model = "gev",
  ntrials = 100,
  nrepeats = 3,
  nx = 50,
  params = c(0, 1, 0),
  alpha = seq(0.005, 0.995, 0.005),
  plotflag = TRUE,
  verbose = TRUE
)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_p1".
ntrials	the number of trials to run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence. 3 is a good choice.
nx	the length of the training data.
params	values for the parameters for the specified distribution
alpha	the alpha values at which to test
plotflag	logical to turn the plotting on and off
verbose	logical to turn loop counting on and off

Details

The maximum likelihood quantiles (plotted in blue) do not give good reliability. They typically underestimate the tails (see panel (f)).

The cp predictive quantiles generally give reasonably good reliability, especially for sample sizes of ~100. The other predictions generally give poor reliability.

Value

A plot showing 9 different reliability checks, and a list containing various outputs, including the probabilities shown in the plot.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to fitdistcp, with more examples, is given [on this webpage](#).

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),

- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
set.seed(1)
# example 1
# -runs the default settings, which test reliability for the GEV distribution
reltest2(nrepeats=1)
```

reltest2_cases	Cases
----------------	-------

Description

Cases

Usage

```
reltest2_cases(model = "gev", nx = 50, params)
```

Arguments

- model which distribution to test. Possibles values are "gev", "gpd_k1", "gev_pred1".
- nx length of training data
- params model parameters

Value

Two integers

reltest2_makeup	Cases
-----------------	-------

Description

Cases

Usage

```
reltest2_makeup(model, pred1, tt0, params)
```

Arguments

- model which distribution to test. Possibles values are "gev", "gpd_k1", "gev_pred1".
- pred1 quantile predictions
- tt0 value of predictor vector
- params model parameters

Value

Vector

reltest2_plot	<i>Plotting routine for reltest2</i>
---------------	--------------------------------------

Description

Plots 9 diagnostics related to predictive probability matching.

Usage

```
reltest2_plot(  
  model,  
  ntrials,  
  nrepeats,  
  nx,  
  params,  
  nmethods,  
  alpha,  
  freqexceeded,  
  case  
)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd", "gev_p1".
ntrials	the number of trials o run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence
nx	the length of the training data.
params	values for the parameters for the specified distribution
nmethods	the number of methods being tested
alpha	the values of alpha being tested
freqexceeded	the exceedance counts
case	there are 3 cases (must be set to case=1 except for my testing)

Value

Plots the results of reliability testing

reltest2_predict	<i>Make prediction from one model</i>
------------------	---------------------------------------

Description

Make prediction from one model

Usage

```
reltest2_predict(model = "gev", xx, tt, n0, pp, params, case, nmethods)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k1", "halfnorm", "norm", "lnorm", "gumbel", "frechet_k1", "weibull", "gev_k3", "logis", "lst_k3", "cauchy", "norm_p1", "lnorm_p1", "logis_p1", "lst_k3p1", "gumbel_p1", "norm_p12", "gev", "gpd", "gev_p1".
xx	training data
tt	predictor vector
n0	index for predictor vector
pp	probabilities to predict
params	model parameters
case	the case number: different models have different lists of methods
nmethods	the number of methods: different models have different numbers of methods

Value

Vector

reltest2_simulate	<i>Random training data from one model</i>
-------------------	--

Description

Random training data from one model

Usage

```
reltest2_simulate(model = "gev", nx = 50, tt, params)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_pred1".
nx	the length of the training data.
tt	the predictor
params	values for the parameters for the specified distribution

Value

Vector

reltest_makeep	<i>Calculate EP from one model</i>
----------------	------------------------------------

Description

Calculate EP from one model

Usage

```
reltest_makeep(model, pred1, tt0, tt10, tt20, tt30, params)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k2", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k2", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k3", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k3", "norm_p12", "lst_p12k3", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1", "gev_p12", "gev_p123".
pred1	quantile predictions
tt0	value of the predictor
tt10	value of predictor 1
tt20	value of predictor 2
tt30	value of predictor 3
params	the model parameters

Value

Vector

reltest_makemaxep	<i>Calculate MaxEP from one model</i>
-------------------	---------------------------------------

Description

Calculate MaxEP from one model

Usage

```
reltest_makemaxep(model, ml_max, tt0, tt10, tt20, tt30, params)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
ml_max	predicted max value
tt0	value of the predictor
tt10	value of predictor 1
tt20	value of predictor 2
tt30	value of predictor 3
params	the model parameters

Value

Vector

reltest_predict	<i>Make prediction from one model</i>
-----------------	---------------------------------------

Description

Make prediction from one model

Usage

```
reltest_predict(  
  model,  
  xx,  
  tt,  
  tt1,  
  tt2,  
  tt3,  
  n0,  
  n10,
```

```

n20,
n30,
pp,
params,
dmgs = TRUE,
debug = FALSE,
aderivs = TRUE,
unbiasedv = FALSE,
pwm = FALSE,
minxi = -10,
maxxi = 10
)

```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k2", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k2", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k3", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "exp_p1k4", "norm_p12", "lst_p12k3", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1", "gev_p12", "gev_p123".
xx	training data
tt	predictor vector
tt1	predictor vector 1
tt2	predictor vector 2
tt3	predictor vector 3
n0	index for predictor vector
n10	index for predictor vector 1
n20	index for predictor vector 2
n30	index for predictor vector 2
pp	probabilites at which to make quantile predictions
params	model parameters
dmgs	flag for whether to run dmgs calculations or not
debug	flag for turning debug messages on
aderivs	a logical for whether to use analytic derivatives (instead of numerical)
unbiasedv	a logical for whether to use the unbiased variance instead of maxlik (for the normal)
pwm	a logical for whether to use PWM instead of maxlik (for the GEV)
minxi	minimum value for EVT shape parameter
maxxi	maximum value for EVT shape parameter

Value

Two vectors

reltest_simulate	<i>Random training data from one model</i>
------------------	--

Description

Random training data from one model

Usage

```
reltest_simulate(  
  model = "exp",  
  nx = 20,  
  tt,  
  tt1,  
  tt2,  
  tt3,  
  params,  
  minxi = -10,  
  maxx = -10  
)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k2", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k2", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k3", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k3", "norm_p12", "lst_p12k3", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
nx	the length of the training data to use.
tt	predictor vector
tt1	predictor vector 1
tt2	predictor vector 2
tt3	predictor vector 2
params	values for the parameters for the specified distribution
minxi	minimum value for EVT shape parameter
maxxi	maximum value for EVT shape parameter

Value

Vector

rgev_minmax	<i>rgev but with maxlik xi guaranteed within bounds</i>
-------------	---

Description

rgev but with maxlik xi guaranteed within bounds

Usage

```
rgev_minmax(nx, mu, sigma, xi, minxi = -0.45, maxx = 0.45)
```

Arguments

- nx length of training data
- mu the location parameter of the distribution
- sigma the sigma parameter of the distribution
- xi the shape parameter of the distribution
- minxi minimum value of shape parameter xi
- maxxi maximum value of shape parameter xi

Value

Vector

rgev_p123_minmax	<i>rgev for gev_p123 but with maxlik xi within bounds</i>
------------------	---

Description

rgev for gev_p123 but with maxlik xi within bounds

Usage

```
rgev_p123_minmax(  
  nx,  
  mu,  
  sigma,  
  xi,  
  t1,  
  t2,  
  t3,  
  minxi = -0.45,  
  maxx = 0.45,  
  centering = TRUE  
)
```

Arguments

nx	length of training data
mu	the location parameter of the distribution
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
centering	indicates whether the routine should center the data or not

Value

Vector

rgev_p12_minmax	<i>rgev for gev_p12 but with maxlik xi within bounds</i>
-----------------	--

Description

rgev for gev_p12 but with maxlik xi within bounds

Usage

```
rgev_p12_minmax(
  nx,
  mu,
  sigma,
  xi,
  t1,
  t2,
  minxi = -0.45,
  maxxi = 0.45,
  centering = TRUE
)
```

Arguments

nx	length of training data
mu	the location parameter of the distribution
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution

t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
centering	indicates whether the routine should center the data or not

Value

Vector

rgev_p1_minmax	<i>rgev for gev_p1 but with maxlik xi within bounds</i>
----------------	---

Description

rgev for gev_p1 but with maxlik xi within bounds

Usage

```
rgev_p1_minmax(  
  nx,  
  mu,  
  sigma,  
  xi,  
  tt,  
  minxi = -0.45,  
  maxxi = 0.45,  
  centering = TRUE  
)
```

Arguments

nx	length of training data
mu	the location parameter of the distribution
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
tt	a vector of predictors
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
centering	indicates whether the routine should center the data or not

Value

Vector

rgpd_k1_minmax	<i>rgpd for gpd_k1 but with maxlik xi within bounds</i>
----------------	---

Description

rgpd for gpd_k1 but with maxlik xi within bounds

Usage

```
rgpd_k1_minmax(nx, kloc, sigma, xi, minxi = -0.45, maxxi = 0.45)
```

Arguments

nx	length of training data
kloc	the known location parameter
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

Vector

rhp_dmgs_cpmethod	<i>Generates a comment about the method</i>
-------------------	---

Description

Generates a comment about the method

Usage

```
rhp_dmgs_cpmethod()
```

Value

String

rust_pumethod	<i>Generates a comment about the method</i>
---------------	---

Description

Generates a comment about the method

Usage

```
rust_pumethod()
```

Value

String

testppm_plot	<i>Plotting routine for testppm</i>
--------------	-------------------------------------

Description

Plots 9 diagnostics related to predictive probability matching.

Usage

```
testppm_plot(  
  model,  
  ntrials,  
  nrepeats,  
  nx,  
  params,  
  nmethods,  
  alpha,  
  freqexceeded  
)
```

Arguments

model	which distribution to test. Possibles values are
ntrials	the number of trials to run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence
nx	the length of the training data.
params	values for the parameters for the specified distribution
nmethods	the number of methods being tested
alpha	the values of alpha being tested
freqexceeded	the exceedance counts

Value

Plots the results of reliability testing

unif_cp

Uniform Distribution Predictions Based on a Calibrating Prior

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qunif_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  debug = FALSE,
  aderivs = TRUE
)

runif_cp(n, x, mlcp = TRUE, debug = FALSE, aderivs = TRUE)

dunif_cp(x, y = x, debug = FALSE, aderivs = TRUE)

punif_cp(x, y = x, debug = FALSE, aderivs = TRUE)
```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>debug</code>	logical for turning on debug messages
<code>aderivs</code>	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
<code>n</code>	the number of random samples required
<code>mlcp</code>	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
<code>y</code>	a vector of values at which to calculate the density and distribution functions

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The uniform distribution has probability density function

$$f(x; min, max) = \frac{1}{max - min}$$

and zero otherwise, where $min \leq x \leq max$ is the random variable and min, max are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\lambda) \propto \frac{1}{max - min}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),

- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d25unif_example_data_v1
cat("length(x)=",length(x),"\\n")
p=c(1:9)/10
q=qunif_cp(x,p)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qunif_cp)",
main="unif: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
```

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x . For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities p , and various other diagnostics.
- `r****_cp` returns n random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values y
- `p****_cp` returns the predictive distribution function at the specified values y
- `t****_cp` returns n random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qweibull_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rweibull_cp(
```

```

    n,
    x,
    fd1 = 0.01,
    fd2 = 0.01,
    rust = FALSE,
    mlcp = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

dweibull_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pweibull_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tweibull_cp(n, x, fd1 = 0.01, fd2 = 0.01, debug = FALSE)

```

Arguments

<code>x</code>	a vector of training data values
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>fd1</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the first parameter
<code>fd2</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the second parameter
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
<code>waicscores</code>	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
<code>logscores</code>	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)

dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times $-1/2$.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- `cp_method`: a comment about the method used to generate the cp prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Weibull distribution has exceedance distribution function

$$S(x; k, \sigma) = \exp \left(- \left(\frac{x}{\sigma} \right)^k \right)$$

where $x \geq 0$ is the random variable and $k > 0, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(k, \sigma) \propto \frac{1}{k\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible

- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d52weibull_example_data_v1
p=c(1:9)/10
q=qweibull_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qweibull_cp)",
main="Weibull: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

weibull_f1f	<i>DMGS equation 3.3, f1 term</i>
-------------	-----------------------------------

Description

DMGS equation 3.3, f1 term

Usage

```
weibull_f1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

weibull_f1fa	<i>The first derivative of the density</i>
--------------	--

Description

The first derivative of the density

Usage

```
weibull_f1fa(x, v1, v2)
```

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |

Value

Vector

weibull_f2f	<i>DMGS equation 3.3, f2 term</i>
-------------	-----------------------------------

Description

DMGS equation 3.3, f2 term

Usage

weibull_f2f(y, v1, fd1, v2, fd2)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| fd2 | the fractional delta used in the numerical derivatives with respect to the parameter |

Value

3d array

weibull_f2fa	<i>The second derivative of the density</i>
--------------	---

Description

The second derivative of the density

Usage

weibull_f2fa(x, v1, v2)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |

Value

Matrix

weibull_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

weibull_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

weibull_1dd	<i>Second derivative matrix of the normalized log-likelihood</i>
-------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

weibull_1dd(x, v1, fd1, v2, fd2)

Arguments

- x a vector of training data values
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

weibull_1dda	<i>The second derivative of the normalized log-likelihood</i>
--------------	---

Description

The second derivative of the normalized log-likelihood

Usage

weibull_1dda(x, v1, v2)

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

Matrix

weibull_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
--------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
weibull_lddd(x, v1, fd1, v2, fd2)
```

Arguments

- x a vector of training data values
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

weibull_lddda	<i>The third derivative of the normalized log-likelihood</i>
---------------	--

Description

The third derivative of the normalized log-likelihood

Usage

```
weibull_lddda(x, v1, v2)
```

Arguments

- x a vector of training data values
- v1 first parameter
- v2 second parameter

Value

3d array

weibull_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
-------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
weibull_lmn(x, v1, fd1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

weibull_lmnp	<i>One component of the third derivative of the normalized log-likelihood</i>
--------------	---

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
weibull_lmnp(x, v1, fd1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

<code>weibull_logf</code>	<i>Logf for RUST</i>
---------------------------	----------------------

Description

Logf for RUST

Usage

`weibull_logf(params, x)`

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

weibull_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

weibull_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-----------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

weibull_loglik	<i>log-likelihood function</i>
----------------	--------------------------------

Description

log-likelihood function

Usage

```
weibull_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

weibull_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
-------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
weibull_logscores(logscores, x, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

weibull_means	<i>MLE and RHP predictive means</i>
---------------	-------------------------------------

Description

MLE and RHP predictive means

Usage

```
weibull_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

weibull_mu1f	<i>DMGS equation 3.3, mu1 term</i>
--------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

```
weibull_mu1f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

<code>weibull_mu1fa</code>	<i>Minus the first derivative of the cdf, at alpha</i>
----------------------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

`weibull_mu1fa(alpha, v1, v2)`

Arguments

- `alpha` a vector of values of alpha (one minus probability)
- `v1` first parameter
- `v2` second parameter

Value

Vector

<code>weibull_mu2f</code>	<i>DMGS equation 3.3, mu2 term</i>
---------------------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

`weibull_mu2f(alpha, v1, fd1, v2, fd2)`

Arguments

- `alpha` a vector of values of alpha (one minus probability)
- `v1` first parameter
- `fd1` the fractional delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `fd2` the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

weibull_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
---------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

weibull_mu2fa(alpha, v1, v2)

Arguments

- alpha a vector of values of alpha (one minus probability)
- v1 first parameter
- v2 second parameter

Value

Matrix

weibull_p1f	<i>DMGS equation 3.3, p1 term</i>
-------------	-----------------------------------

Description

DMGS equation 3.3, p1 term

Usage

weibull_p1f(y, v1, fd1, v2, fd2)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- fd2 the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

<code>weibull_p1fa</code>	<i>The first derivative of the cdf</i>
---------------------------	--

Description

The first derivative of the cdf

Usage

`weibull_p1fa(x, v1, v2)`

Arguments

- `x` a vector of training data values
- `v1` first parameter
- `v2` second parameter

Value

Vector

<code>weibull_p2f</code>	<i>DMGS equation 3.3, p2 term</i>
--------------------------	-----------------------------------

Description

DMGS equation 3.3, p2 term

Usage

`weibull_p2f(y, v1, fd1, v2, fd2)`

Arguments

- `y` a vector of values at which to calculate the density and distribution functions
- `v1` first parameter
- `fd1` the fractional delta used in the numerical derivatives with respect to the parameter
- `v2` second parameter
- `fd2` the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

weibull_p2fa	<i>The second derivative of the cdf</i>
--------------	---

Description

The second derivative of the cdf

Usage

weibull_p2fa(x, v1, v2)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| v1 | first parameter |
| v2 | second parameter |

Value

Matrix

weibull_p2_cp	<i>weibull Distribution with a Predictor on the Scale Parameter, Predictions Based on a Calibrating Prior</i>
---------------	---

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y

- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qweibull_p2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rweibull_p2_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  fd1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
dweibull_p2_cp(
  x,
```

```

    t,
    t0 = NA,
    n0 = NA,
    y = x,
    fd1 = 0.01,
    d2 = 0.01,
    d3 = 0.01,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
)

```

```

pweibull_p2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  fd1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

```

```
tweibull_p2_cp(n, x, t, fd1 = 0.01, d2 = 0.01, d3 = 0.01, debug = FALSE)
```

Arguments

<code>x</code>	a vector of training data values
<code>t</code>	a vector of predictors, such that <code>length(t)=length(x)</code>
<code>t0</code>	a single value of the predictor (specify either <code>t0</code> or <code>n0</code> but not both)
<code>n0</code>	an index for the predictor (specify either <code>t0</code> or <code>n0</code> but not both)
<code>p</code>	a vector of probabilities at which to generate predictive quantiles
<code>fd1</code>	if <code>aderivs=FALSE</code> , the fractional delta used for numerical derivatives with respect to the first parameter
<code>d2</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the second parameter
<code>d3</code>	if <code>aderivs=FALSE</code> , the delta used for numerical derivatives with respect to the third parameter
<code>means</code>	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times $-1/2$.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of x

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Weibull distribution with predictor on the scale parameter has exceedance distribution function

$$S(x; k, a, b) = \exp \left(- \left(\frac{x}{\sigma(a, b)} \right)^k \right)$$

where $x \geq 0$ is the random variable, $k > 0$ is the shape parameter and $\sigma = e^{a+bt}$ is the scale parameter, modelled as a function of parameters a, b and predictor t .

The calibrating prior is given by the right Haar prior, which is

$$\pi(k, \sigma) \propto \frac{1}{k}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- `ru_deviates`: `nrust` predictive random deviates calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

p**** optionally returns the following:

If rust=TRUE:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., $n < 20$), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), <https://ascmo.copernicus.org/articles/11/1/2025/>.

See Also

An introduction to `fitdistcp`, with more examples, is given [on this webpage](#).

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean (gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (1st_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (1st_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d73weibull_p2_example_data_v1_x
tt=fitdistcp::d73weibull_p2_example_data_v1_t
p=c(1:9)/10
n0=10
```

```
q=qweibull_p2_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qweibull_p2_cp)",
main="Weibull w/ p2: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

weibull_p2_f1f	<i>DMGS equation 2.1, f1 term</i>
----------------	-----------------------------------

Description

DMGS equation 2.1, f1 term

Usage

```
weibull_p2_f1f(y, t0, v1, fd1, v2, d2, v3, d3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

weibull_p2_f1fa	<i>The first derivative of the density</i>
-----------------	--

Description

The first derivative of the density

Usage

weibull_p2_f1fa(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Vector

weibull_p2_f2f	<i>DMGS equation 2.1, f2 term</i>
----------------	-----------------------------------

Description

DMGS equation 2.1, f2 term

Usage

weibull_p2_f2f(y, t0, v1, fd1, v2, d2, v3, d3)

Arguments

- y a vector of values at which to calculate the density and distribution functions
- t0 a single value of the predictor (specify either t0 or n0 but not both)
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- v3 third parameter
- d3 the delta used in the numerical derivatives with respect to the parameter

Value

3d array

weibull_p2_f2fa	<i>The second derivative of the density</i>
-----------------	---

Description

The second derivative of the density

Usage

```
weibull_p2_f2fa(x, t, v1, v2, v3)
```

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Matrix

weibull_p2_fd	<i>First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	--

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_fd(x, t, v1, v2, v3)
```

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Vector

weibull_p2_fdd	<i>Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	---

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

weibull_p2_fdd(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Matrix

weibull_p2_ldd	<i>Second derivative matrix of the normalized log-likelihood</i>
----------------	--

Description

Second derivative matrix of the normalized log-likelihood

Usage

weibull_p2_ldd(x, t, v1, fd1, v2, d2, v3, d3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

weibull_p2_ldda	<i>The second derivative of the normalized log-likelihood</i>
-----------------	---

Description

The second derivative of the normalized log-likelihood

Usage

```
weibull_p2_ldda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

weibull_p2_lddd	<i>Third derivative tensor of the normalized log-likelihood</i>
-----------------	---

Description

Third derivative tensor of the normalized log-likelihood

Usage

weibull_p2_lddd(x, t, v1, fd1, v2, d2, v3, d3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- fd1 the fractional delta used in the numerical derivatives with respect to the parameter
- v2 second parameter
- d2 the delta used in the numerical derivatives with respect to the parameter
- v3 third parameter
- d3 the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

weibull_p2_lddda	<i>The third derivative of the normalized log-likelihood</i>
------------------	--

Description

The third derivative of the normalized log-likelihood

Usage

weibull_p2_lddda(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

3d array

weibull_p2_lmn	<i>One component of the second derivative of the normalized log-likelihood</i>
----------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
weibull_p2_lmn(x, t, v1, fd1, v2, d2, v3, d3, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

<code>weibull_p2_lmp</code>	<i>One component of the second derivative of the normalized log-likelihood</i>
-----------------------------	--

Description

One component of the second derivative of the normalized log-likelihood

Usage

`weibull_p2_lmp(x, t, v1, fd1, v2, d2, v3, d3, mm, nn, rr)`

Arguments

<code>x</code>	a vector of training data values
<code>t</code>	a vector or matrix of predictors
<code>v1</code>	first parameter
<code>fd1</code>	the fractional delta used in the numerical derivatives with respect to the parameter
<code>v2</code>	second parameter
<code>d2</code>	the delta used in the numerical derivatives with respect to the parameter
<code>v3</code>	third parameter
<code>d3</code>	the delta used in the numerical derivatives with respect to the parameter
<code>mm</code>	an index for which derivative to calculate
<code>nn</code>	an index for which derivative to calculate
<code>rr</code>	an index for which derivative to calculate

Value

Scalar value

<code>weibull_p2_logf</code>	<i>Logf for RUST</i>
------------------------------	----------------------

Description

Logf for RUST

Usage

`weibull_p2_logf(params, x, t)`

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

weibull_p2_logfdd	<i>Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------------	---

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

weibull_p2_logfdd(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

weibull_p2_logfddd	<i>Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
--------------------	--

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

weibull_p2_loglik	<i>observed log-likelihood function</i>
-------------------	---

Description

observed log-likelihood function

Usage

```
weibull_p2_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

weibull_p2_logscores	<i>Log scores for MLE and RHP predictions calculated using leave-one-out</i>
----------------------	--

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
weibull_p2_logscores(logscores, x, t, fd1, d2, d3, aderivs)
```

Arguments

- | | |
|-----------|---|
| logscores | logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime) |
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| d3 | the delta used in the numerical derivatives with respect to the parameter |
| aderivs | logical for whether to use analytic derivatives (instead of numerical) |

Value

Two scalars

weibull_p2_means	<i>weibull distribution: RHP mean</i>
------------------	---------------------------------------

Description

weibull distribution: RHP mean

Usage

```
weibull_p2_means(means, t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

weibull_p2_mu1f	<i>DMGS equation 3.3, mu1 term</i>
-----------------	------------------------------------

Description

DMGS equation 3.3, mu1 term

Usage

`weibull_p2_mu1f(alpha, t0, v1, fd1, v2, d2, v3, d3)`

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

weibull_p2_mu1fa	<i>Minus the first derivative of the cdf, at alpha</i>
------------------	--

Description

Minus the first derivative of the cdf, at alpha

Usage

weibull_p2_mu1fa(alpha, t, v1, v2, v3)

Arguments

- | | |
|-------|---|
| alpha | a vector of values of alpha (one minus probability) |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Vector

weibull_p2_mu2f	<i>DMGS equation 3.3, mu2 term</i>
-----------------	------------------------------------

Description

DMGS equation 3.3, mu2 term

Usage

weibull_p2_mu2f(alpha, t0, v1, fd1, v2, d2, v3, d3)

Arguments

- | | |
|-------|--|
| alpha | a vector of values of alpha (one minus probability) |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| v3 | third parameter |
| d3 | the delta used in the numerical derivatives with respect to the parameter |

Value

3d array

weibull_p2_mu2fa	<i>Minus the second derivative of the cdf, at alpha</i>
------------------	---

Description

Minus the second derivative of the cdf, at alpha

Usage

weibull_p2_mu2fa(alpha, t, v1, v2, v3)

Arguments

- alpha a vector of values of alpha (one minus probability)
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Matrix

weibull_p2_p1f	<i>DMGS equation 2.1, p1 term</i>
----------------	-----------------------------------

Description

DMGS equation 2.1, p1 term

Usage

weibull_p2_p1f(y, t0, v1, fd1, v2, d2, v3, d3)

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

weibull_p2_p1fa	<i>The first derivative of the cdf</i>
-----------------	--

Description

The first derivative of the cdf

Usage

weibull_p2_p1fa(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

weibull_p2_p2f	<i>DMGS equation 2.1, p2 term</i>
----------------	-----------------------------------

Description

DMGS equation 2.1, p2 term

Usage

weibull_p2_p2f(y, t0, v1, fd1, v2, d2, v3, d3)

Arguments

- | | |
|-----|--|
| y | a vector of values at which to calculate the density and distribution functions |
| t0 | a single value of the predictor (specify either t0 or n0 but not both) |
| v1 | first parameter |
| fd1 | the fractional delta used in the numerical derivatives with respect to the parameter |
| v2 | second parameter |
| d2 | the delta used in the numerical derivatives with respect to the parameter |
| v3 | third parameter |
| d3 | the delta used in the numerical derivatives with respect to the parameter |

Value

3d array

weibull_p2_p2fa	<i>The second derivative of the cdf</i>
-----------------	---

Description

The second derivative of the cdf

Usage

weibull_p2_p2fa(x, t, v1, v2, v3)

Arguments

- | | |
|----|----------------------------------|
| x | a vector of training data values |
| t | a vector or matrix of predictors |
| v1 | first parameter |
| v2 | second parameter |
| v3 | third parameter |

Value

Matrix

weibull_p2_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
---------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

weibull_p2_pd(x, t, v1, v2, v3)

Arguments

- x a vector of training data values
- t a vector or matrix of predictors
- v1 first parameter
- v2 second parameter
- v3 third parameter

Value

Vector

weibull_p2_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
----------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

weibull_p2_pdd(x, t, v1, v2, v3)

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

weibull_p2_predictordata
<i>Predicted Parameter and Generalized Residuals</i>

Description

Predicted Parameter and Generalized Residuals

Usage

weibull_p2_predictordata(predictordata, x, t, t0, params)

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

weibull_p2_waic	<i>Waic</i>
-----------------	-------------

Description

Waic

Usage

```
weibull_p2_waic(  
  waicscores,  
  x,  
  t,  
  v1hat,  
  fd1,  
  v2hat,  
  d2,  
  v3hat,  
  d3,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs = TRUE  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

weibull_pd	<i>First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
------------	--

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

weibull_pdd	<i>Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol</i>
-------------	---

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

weibull_waic	<i>Waic for RUST</i>
--------------	----------------------

Description

Waic for RUST

Usage

```
weibull_waic(  
  waicscores,  
  x,  
  v1hat,  
  fd1,  
  v2hat,  
  fd2,  
  lddi,  
  lddd,  
  lambdad,  
  aderivs  
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

Index

adhoc_dmgs_cpmethod, 26
analytic_cpmethod, 26

bayesian_dq_4terms_v1, 27

calc_revert2m1, 27
cauchy_cp, 28
cauchy_f1f, 35
cauchy_f1fa, 35
cauchy_f2f, 36
cauchy_f2fa, 36
cauchy_fd, 37
cauchy_fdd, 37
cauchy_ldd, 38
cauchy_ldda, 38
cauchy_lddd, 39
cauchy_lddda, 39
cauchy_lmn, 40
cauchy_lmnp, 40
cauchy_logf, 41
cauchy_logfdd, 42
cauchy_logfddd, 42
cauchy_loglik, 43
cauchy_logscores, 43
cauchy_mu1f, 44
cauchy_mu2f, 44
cauchy_p1_cp, 45
cauchy_p1_f1f, 53
cauchy_p1_f1fa, 54
cauchy_p1_f2f, 54
cauchy_p1_f2fa, 55
cauchy_p1_fd, 55
cauchy_p1_fdd, 56
cauchy_p1_ldd, 56
cauchy_p1_ldda, 57
cauchy_p1_lddd, 58
cauchy_p1_lddda, 58
cauchy_p1_lmn, 59
cauchy_p1_lmnp, 60
cauchy_p1_logf, 60

cauchy_p1_logfdd, 61
cauchy_p1_logfddd, 62
cauchy_p1_loglik, 62
cauchy_p1_logscores, 63
cauchy_p1_means, 63
cauchy_p1_mu1f, 64
cauchy_p1_mu2f, 65
cauchy_p1_p1f, 65
cauchy_p1_p2f, 66
cauchy_p1_predictordata, 67
cauchy_p1_waic, 67
cauchy_p1f, 45
cauchy_p2f, 68
cauchy_waic, 69
crhpflat_dmgs_cpmethod, 70

d100gamma_example_data_v1, 70
d101invgamma_example_data_v1, 70
d102invgauss_example_data_v1, 70
d105burr_example_data_v1, 71
d10exp_example_data_v1, 71
d110gev_example_data_v1, 71
d11pareto_k2_example_data_v1, 71
d120gpd_k1_example_data_v1, 71
d150gev_p1_example_data_v1_t, 72
d150gev_p1_example_data_v1_x, 72
d151gev_p12_example_data_v1_t, 72
d151gev_p12_example_data_v1_x, 72
d152gev_p123_example_data_v1_t, 72
d152gev_p123_example_data_v1_x, 73
d20halfnorm_example_data_v1, 73
d25unif_example_data_v1, 73
d30norm_example_data_v1, 73
d31norm_dmgs_example_data_v1, 73
d32gnorm_k3_example_data_v1, 74
d35lnorm_example_data_v1, 74
d36lnorm_dmgs_example_data_v1, 74
d40logis_example_data_v1, 74
d411st_k3_example_data_v1, 74
d42cauchy_example_data_v1, 75

- d50gumbel_example_data_v1, 75
- d51frechet_k1_example_data_v1, 75
- d52weibull_example_data_v1, 75
- d53gev_k3_example_data_v1, 75
- d55exp_p1_example_data_v1_t, 76
- d55exp_p1_example_data_v1_x, 76
- d56pareto_p1k2_example_data_v1_t, 76
- d56pareto_p1k2_example_data_v1_x, 76
- d60norm_p1_example_data_v1_t, 76
- d60norm_p1_example_data_v1_x, 77
- d61lnorm_p1_example_data_v1_t, 77
- d61lnorm_p1_example_data_v1_x, 77
- d62logis_p1_example_data_v1_t, 77
- d62logis_p1_example_data_v1_x, 77
- d63lst_p1k3_example_data_v1_t, 78
- d63lst_p1k3_example_data_v1_x, 78
- d64cauchy_p1_example_data_v1_t, 78
- d64cauchy_p1_example_data_v1_x, 78
- d70gumbel_p1_example_data_v1_t, 78
- d70gumbel_p1_example_data_v1_x, 79
- d71frechet_p2k1_example_data_v1_t, 79
- d71frechet_p2k1_example_data_v1_x, 79
- d72weibull_p1_example_data_v1_t, 79
- d72weibull_p1_example_data_v1_x, 79
- d73weibull_p2_example_data_v1_t, 80
- d73weibull_p2_example_data_v1_x, 80
- d74gev_p1k3_example_data_v1_t, 80
- d74gev_p1k3_example_data_v1_x, 80
- d80norm_p12_example_data_v1_t1, 80
- d80norm_p12_example_data_v1_t2, 81
- d80norm_p12_example_data_v1_x, 81
- d81lst_p12k3_example_data_v1_t1, 81
- d81lst_p12k3_example_data_v1_t2, 81
- d81lst_p12k3_example_data_v1_x, 81
- d82weibull_p12_example_data_v1_t1, 82
- d82weibull_p12_example_data_v1_t2, 82
- d82weibull_p12_example_data_v1_x, 82
- dcauchy_cp (cauchy_cp), 28
- dcauchy_p1, 83
- dcauchy_p1_cp (cauchy_p1_cp), 45
- dcauchy_p1sub, 83
- dcauchysub, 82
- deriv_copyfdd, 84
- deriv_copyld2, 85
- deriv_copyldd, 85
- deriv_copylddd, 86
- dexp_cp (exp_cp), 116
- dexp_p1, 87
- dexp_p1_cp (exp_p1_cp), 130
- dexp_p1sub, 87
- dexpsub, 86
- dfrechet_k1_cp (frechet_k1_cp), 157
- dfrechet_p2k1, 88
- dfrechet_p2k1_cp (frechet_p2k1_cp), 180
- dfrechet_p2k1sub, 89
- dfrechetsub, 88
- dgamma_cp (gamma_cp), 209
- dgamma_sub, 90
- dgev_cp (gev_cp), 230
- dgev_k3_cp (gev_k3_cp), 245
- dgev_k3sub, 91
- dgev_p1, 92
- dgev_p12, 92
- dgev_p123, 93
- dgev_p123_cp (gev_p123_cp), 277
- dgev_p123sub, 94
- dgev_p12_cp (gev_p12_cp), 321
- dgev_p12sub, 95
- dgev_p1_cp (gev_p1_cp), 374
- dgev_p1k3, 96
- dgev_p1k3_cp (gev_p1k3_cp), 349
- dgev_p1k3sub, 97
- dgev_p1sub, 97
- dgevs, 90
- dgnorm_k3_cp (gnorm_k3_cp), 405
- dgnorm_k3sub, 99
- dgpd_k1_cp (gpd_k1_cp), 437
- dgpdsub, 99
- dgumbel_cp (gumbel_cp), 462
- dgumbel_p1, 101
- dgumbel_p1_cp (gumbel_p1_cp), 481
- dgumbel_p1sub, 101
- dgumbelsub, 100
- dhalfnorm_cp (halfnorm_cp), 510
- dhalfnormsub, 102
- dinvgamma_cp (invgamma_cp), 529
- dinvgamma_sub, 102
- dinvgauss_cp (invgauss_cp), 549
- dinvgauss_sub, 103
- dlnorm_cp (lnorm_cp), 570
- dlnorm_dmgs_cp (lnorm_dmgs_cp), 577
- dlnorm_dmgs_sub, 104
- dlnorm_p1, 104
- dlnorm_p1_cp (lnorm_p1_cp), 599
- dlnorm_p1sub, 105
- dlnormsub, 103

- dlogis2sub, 105
- dlogis_cp (logis_cp), 623
- dlogis_p1, 106
- dlogis_p1_cp (logis_p1_cp), 642
- dlogis_p1sub, 106
- dlst_k3_cp (lst_k3_cp), 670
- dlst_k3sub, 107
- dlst_p1k3, 108
- dlst_p1k3_cp (lst_p1k3_cp), 690
- dlst_p1k3sub, 108
- dmgs, 109
- dnorm_cp (norm_cp), 745
- dnorm_dmgs_cp (norm_dmgs_cp), 750
- dnorm_dmgs_sub, 110
- dnorm_p1, 111
- dnorm_p1_cp (norm_p1_cp), 773
- dnorm_p1_formula, 112
- dnorm_p1sub, 111
- dnormsub, 110
- dpareto_k2_cp (pareto_k2_cp), 800
- dpareto_k2_sub, 112
- dpareto_p1k2, 113
- dpareto_p1k2_cp (pareto_p1k2_cp), 818
- dpareto_p1k2sub, 113
- dunif_cp (unif_cp), 891
- dunif_formula, 114
- dweibull_cp (weibull_cp), 897
- dweibull_p2, 115
- dweibull_p2_cp (weibull_p2_cp), 917
- dweibull_p2sub, 115
- dweibullsub, 114

- exp_cp, 116
- exp_f1f, 122
- exp_f1fa, 122
- exp_f2f, 123
- exp_f2fa, 123
- exp_fd, 124
- exp_fdd, 124
- exp_l111, 125
- exp_ldd, 125
- exp_ldda, 126
- exp_lddd, 126
- exp_lddda, 127
- exp_logf, 127
- exp_logfdd, 128
- exp_logfddd, 128
- exp_logscores, 129
- exp_p1_cp, 130
- exp_p1_f1f, 137
- exp_p1_f1fa, 138
- exp_p1_f2f, 138
- exp_p1_f2fa, 139
- exp_p1_fd, 139
- exp_p1_fdd, 140
- exp_p1_ldd, 140
- exp_p1_ldda, 141
- exp_p1_lddd, 141
- exp_p1_lddda, 142
- exp_p1_lmn, 142
- exp_p1_lmnp, 143
- exp_p1_logf, 143
- exp_p1_logfdd, 144
- exp_p1_logfddd, 145
- exp_p1_loglik, 145
- exp_p1_logscores, 146
- exp_p1_means, 146
- exp_p1_mu1f, 147
- exp_p1_mu1fa, 148
- exp_p1_mu2f, 148
- exp_p1_mu2fa, 149
- exp_p1_p1f, 149
- exp_p1_p1fa, 150
- exp_p1_p2f, 150
- exp_p1_p2fa, 151
- exp_p1_pd, 151
- exp_p1_pdd, 152
- exp_p1_predictordata, 152
- exp_p1_waic, 153
- exp_p1fa, 129
- exp_p2fa, 154
- exp_pd, 154
- exp_pdd, 155
- exp_waic, 155

- fixgevrage, 156
- fixgpdrange, 156
- frechet_k1_cp, 157
- frechet_k1_f1f, 164
- frechet_k1_f1fa, 164
- frechet_k1_f2f, 165
- frechet_k1_f2fa, 166
- frechet_k1_fd, 166
- frechet_k1_fdd, 167
- frechet_k1_ldd, 167
- frechet_k1_ldda, 168
- frechet_k1_lddd, 168
- frechet_k1_lddda, 169

- frechet_k1_lmn, 169
- frechet_k1_lmnp, 170
- frechet_k1_logf, 171
- frechet_k1_logfdd, 171
- frechet_k1_logfddd, 172
- frechet_k1_mu1f, 172
- frechet_k1_mu1fa, 173
- frechet_k1_mu2f, 173
- frechet_k1_mu2fa, 174
- frechet_k1_p1f, 174
- frechet_k1_p1fa, 175
- frechet_k1_p2f, 175
- frechet_k1_p2fa, 176
- frechet_k1_pd, 176
- frechet_k1_pdd, 177
- frechet_k1_waic, 177
- frechet_loglik, 178
- frechet_logscores, 179
- frechet_means, 179
- frechet_p2k1_cp, 180
- frechet_p2k1_f1f, 188
- frechet_p2k1_f1fa, 189
- frechet_p2k1_f2f, 189
- frechet_p2k1_f2fa, 190
- frechet_p2k1_fd, 191
- frechet_p2k1_fdd, 191
- frechet_p2k1_ddd, 192
- frechet_p2k1_ldda, 193
- frechet_p2k1_lddd, 193
- frechet_p2k1_lddda, 194
- frechet_p2k1_lmn, 195
- frechet_p2k1_lmnp, 195
- frechet_p2k1_logf, 196
- frechet_p2k1_logfdd, 197
- frechet_p2k1_logfddd, 197
- frechet_p2k1_loglik, 198
- frechet_p2k1_logscores, 199
- frechet_p2k1_means, 199
- frechet_p2k1_mu1f, 200
- frechet_p2k1_mu1fa, 201
- frechet_p2k1_mu2f, 202
- frechet_p2k1_mu2fa, 202
- frechet_p2k1_p1f, 203
- frechet_p2k1_p1fa, 204
- frechet_p2k1_p2f, 204
- frechet_p2k1_p2fa, 205
- frechet_p2k1_pd, 206
- frechet_p2k1_pdd, 206
- frechet_p2k1_predictordata, 207
- frechet_p2k1_waic, 208
- gamma_cp, 209
- gamma_f1f, 216
- gamma_f1fa, 216
- gamma_f2f, 217
- gamma_f2fa, 217
- gamma_fd, 218
- gamma_fdd, 218
- gamma_gg, 219
- gamma_gmn, 219
- gamma_ddd, 220
- gamma_ldda, 220
- gamma_lddd, 221
- gamma_lddda, 221
- gamma_lmn, 222
- gamma_lmnp, 222
- gamma_logf, 223
- gamma_logfdd, 224
- gamma_logfddd, 224
- gamma_loglik, 225
- gamma_logscores, 225
- gamma_means, 226
- gamma_mu1f, 226
- gamma_mu2f, 227
- gamma_p1f, 227
- gamma_p2f, 228
- gamma_waic, 229
- gev_checkmle, 229
- gev_cp, 230
- gev_f1f, 239
- gev_f1fa, 240
- gev_f2f, 240
- gev_f2fa, 241
- gev_fd, 241
- gev_fdd, 242
- gev_ggd_mev, 242
- gev_ggid_mev, 243
- gev_k12_ppm_minusloglik, 244
- gev_k3_cp, 245
- gev_k3_f1f, 253
- gev_k3_f1fa, 253
- gev_k3_f2f, 254
- gev_k3_f2fa, 254
- gev_k3_fd, 255
- gev_k3_fdd, 255
- gev_k3_ddd, 256
- gev_k3_ldda, 256

gev_k3_lddd, 257
gev_k3_lddda, 257
gev_k3_lmn, 258
gev_k3_lmnp, 258
gev_k3_logf, 259
gev_k3_logfdd, 260
gev_k3_logfddd, 260
gev_k3_loglik, 261
gev_k3_means, 261
gev_k3_mu1f, 262
gev_k3_mu1fa, 262
gev_k3_mu2f, 263
gev_k3_mu2fa, 263
gev_k3_pd, 264
gev_k3_pdd, 264
gev_k3_waic, 265
gev_ld12a, 266
gev_lda, 266
gev_ldd, 267
gev_ldda, 267
gev_ldddd, 268
gev_lddda, 268
gev_lmn, 269
gev_lmnp, 270
gev_logf, 270
gev_logfd, 271
gev_logfdd, 271
gev_logfddd, 272
gev_loglik, 273
gev_means, 273
gev_mu1f, 274
gev_mu1fa, 275
gev_mu2f, 275
gev_mu2fa, 276
gev_p123_checkmle, 276
gev_p123_cp, 277
gev_p123_f1f, 287
gev_p123_f1fa, 288
gev_p123_f2f, 288
gev_p123_f2fa, 289
gev_p123_fd, 290
gev_p123_fdd, 291
gev_p123_ldd, 291
gev_p123_ldda, 292
gev_p123_lddd, 293
gev_p123_lddda, 294
gev_p123_lmn, 294
gev_p123_lmnp, 296
gev_p123_logf, 297
gev_p123_logfdd, 298
gev_p123_logfddd, 298
gev_p123_loglik, 299
gev_p123_means, 300
gev_p123_mu1f, 300
gev_p123_mu1fa, 301
gev_p123_mu2f, 302
gev_p123_mu2fa, 303
gev_p123_pd, 304
gev_p123_pdd, 305
gev_p123_predictordata, 305
gev_p123_setics, 306
gev_p123_waic, 307
gev_p12_checkmle, 320
gev_p12_cp, 321
gev_p12_f1f, 330
gev_p12_f1fa, 331
gev_p12_f2f, 331
gev_p12_f2fa, 332
gev_p12_fd, 333
gev_p12_fdd, 333
gev_p12_ggd, 334
gev_p12_ldd, 335
gev_p12_ldda, 335
gev_p12_lddd, 336
gev_p12_lddda, 337
gev_p12_lmn, 337
gev_p12_lmnp, 338
gev_p12_logf, 339
gev_p12_logfdd, 340
gev_p12_logfddd, 340
gev_p12_loglik, 341
gev_p12_means, 342
gev_p12_mu1f, 342
gev_p12_mu1fa, 343
gev_p12_mu2f, 344
gev_p12_mu2fa, 344
gev_p12_pd, 345
gev_p12_pdd, 346
gev_p12_predictordata, 346
gev_p12_setics, 347
gev_p12_waic, 348
gev_p12k3_f1f, 308
gev_p12k3_f1fa, 309
gev_p12k3_f2f, 309
gev_p12k3_f2fa, 310
gev_p12k3_fd, 311

gev_p12k3_fdd, 311
gev_p12k3_ldd, 312
gev_p12k3_ldda, 313
gev_p12k3_lddd, 313
gev_p12k3_lddda, 314
gev_p12k3_logfdd, 315
gev_p12k3_logfddd, 315
gev_p12k3_mu1f, 316
gev_p12k3_mu1fa, 317
gev_p12k3_mu2f, 317
gev_p12k3_mu2fa, 318
gev_p12k3_pd, 319
gev_p12k3_pdd, 319
gev_p1_checkm1e, 374
gev_p1_cp, 374
gev_p1_f1f, 384
gev_p1_f1fa, 385
gev_p1_f2f, 385
gev_p1_f2fa, 386
gev_p1_fd, 387
gev_p1_fdd, 387
gev_p1_ggd, 388
gev_p1_ldd, 389
gev_p1_ldda, 389
gev_p1_lddd, 390
gev_p1_lddda, 391
gev_p1_lmn, 391
gev_p1_lmnp, 392
gev_p1_logf, 393
gev_p1_logfdd, 393
gev_p1_logfddd, 394
gev_p1_loglik, 395
gev_p1_means, 395
gev_p1_mu1f, 396
gev_p1_mu1fa, 397
gev_p1_mu2f, 397
gev_p1_mu2fa, 398
gev_p1_pd, 399
gev_p1_pdd, 399
gev_p1_predictordata, 400
gev_p1_setics, 401
gev_p1_waic, 401
gev_p1k3_cp, 349
gev_p1k3_f1f, 357
gev_p1k3_f1fa, 358
gev_p1k3_f2f, 359
gev_p1k3_f2fa, 359
gev_p1k3_fd, 360
gev_p1k3_fdd, 361
gev_p1k3_ldd, 361
gev_p1k3_ldda, 362
gev_p1k3_lddd, 363
gev_p1k3_lddda, 363
gev_p1k3_lmn, 364
gev_p1k3_lmnp, 365
gev_p1k3_logf, 365
gev_p1k3_logfdd, 366
gev_p1k3_logfddd, 367
gev_p1k3_loglik, 367
gev_p1k3_means, 368
gev_p1k3_mu1f, 368
gev_p1k3_mu1fa, 369
gev_p1k3_mu2f, 370
gev_p1k3_mu2fa, 370
gev_p1k3_pd, 371
gev_p1k3_pdd, 372
gev_p1k3_predictordata, 372
gev_p1k3_waic, 373
gev_pd, 402
gev_pdd, 403
gev_pwm_params, 403
gev_setics, 404
gev_waic, 404
gnorm_k3_cp, 405
gnorm_k3_f1f, 412
gnorm_k3_f1fa, 413
gnorm_k3_f2f, 414
gnorm_k3_f2fa, 414
gnorm_k3_fd, 415
gnorm_k3_fdd, 415
gnorm_k3_ldd, 416
gnorm_k3_ldda, 417
gnorm_k3_lddd, 417
gnorm_k3_lddda, 418
gnorm_k3_lmn, 418
gnorm_k3_logf, 419
gnorm_k3_logfdd, 419
gnorm_k3_logfddd, 420
gnorm_k3_loglik, 420
gnorm_k3_logscores, 421
gnorm_k3_mu1f, 421
gnorm_k3_mu2f, 422
gnorm_k3_p1f, 423
gnorm_k3_p2f, 423
gnorm_lmnp, 424
gnorm_waic, 425

gpd_k13_f1f, 426
 gpd_k13_f1fa, 426
 gpd_k13_f2f, 427
 gpd_k13_f2fa, 427
 gpd_k13_fd, 428
 gpd_k13_fdd, 428
 gpd_k13_l11, 429
 gpd_k13_l111, 429
 gpd_k13_ldd, 430
 gpd_k13_ldda, 430
 gpd_k13_lddd, 431
 gpd_k13_lddda, 431
 gpd_k13_logfdd, 432
 gpd_k13_logfddd, 432
 gpd_k13_mu1f, 433
 gpd_k13_mu1fa, 433
 gpd_k13_mu2f, 434
 gpd_k13_mu2fa, 434
 gpd_k13_p1f, 435
 gpd_k13_p2f, 435
 gpd_k13_pd, 436
 gpd_k13_pdd, 436
 gpd_k1_checkmle, 437
 gpd_k1_cp, 437
 gpd_k1_f1f, 446
 gpd_k1_f1fa, 447
 gpd_k1_f2f, 447
 gpd_k1_f2fa, 448
 gpd_k1_fd, 448
 gpd_k1_fdd, 449
 gpd_k1_ggd_mev, 449
 gpd_k1_ldd, 450
 gpd_k1_ldda, 450
 gpd_k1_lddd, 451
 gpd_k1_lddda, 451
 gpd_k1_lmn, 452
 gpd_k1_lmnp, 452
 gpd_k1_logf, 453
 gpd_k1_logfdd, 454
 gpd_k1_logfddd, 454
 gpd_k1_loglik, 455
 gpd_k1_means, 455
 gpd_k1_mu1f, 456
 gpd_k1_mu1fa, 457
 gpd_k1_mu2f, 457
 gpd_k1_mu2fa, 458
 gpd_k1_p1f, 458
 gpd_k1_p2f, 459
 gpd_k1_pd, 459
 gpd_k1_pdd, 460
 gpd_k1_setics, 460
 gpd_k1_waic, 461
 gumbel_cp, 462
 gumbel_f1f, 469
 gumbel_f1fa, 469
 gumbel_f2f, 470
 gumbel_f2fa, 470
 gumbel_fd, 471
 gumbel_fdd, 471
 gumbel_ldd, 472
 gumbel_ldda, 472
 gumbel_lddd, 473
 gumbel_lddda, 473
 gumbel_lmn, 474
 gumbel_lmnp, 474
 gumbel_logf, 475
 gumbel_logfdd, 476
 gumbel_logfddd, 476
 gumbel_loglik, 477
 gumbel_logscores, 477
 gumbel_means, 478
 gumbel_mu1f, 478
 gumbel_mu1fa, 479
 gumbel_mu2f, 479
 gumbel_mu2fa, 480
 gumbel_p1_cp, 481
 gumbel_p1_f1f, 489
 gumbel_p1_f1fa, 490
 gumbel_p1_f2f, 490
 gumbel_p1_f2fa, 491
 gumbel_p1_fd, 491
 gumbel_p1_fdd, 492
 gumbel_p1_ldd, 492
 gumbel_p1_ldda, 493
 gumbel_p1_lddd, 494
 gumbel_p1_lddda, 494
 gumbel_p1_lmn, 495
 gumbel_p1_lmnp, 496
 gumbel_p1_logf, 496
 gumbel_p1_logfdd, 497
 gumbel_p1_logfddd, 498
 gumbel_p1_loglik, 498
 gumbel_p1_logscores, 499
 gumbel_p1_means, 499
 gumbel_p1_mu1f, 500
 gumbel_p1_mu1fa, 501

gumbel_p1_mu2f, 501
gumbel_p1_mu2fa, 502
gumbel_p1_p1f, 502
gumbel_p1_p1fa, 503
gumbel_p1_p2f, 504
gumbel_p1_p2fa, 504
gumbel_p1_pd, 505
gumbel_p1_pdd, 505
gumbel_p1_predictordata, 506
gumbel_p1_waic, 507
gumbel_p1f, 480
gumbel_p1fa, 481
gumbel_p2f, 508
gumbel_p2fa, 508
gumbel_pd, 509
gumbel_pdd, 509
gumbel_waic, 510

halfnorm_cp, 510
halfnorm_f1f, 517
halfnorm_f1fa, 518
halfnorm_f2f, 518
halfnorm_f2fa, 519
halfnorm_fd, 519
halfnorm_fdd, 520
halfnorm_gg, 520
halfnorm_gg11, 521
halfnorm_l111, 521
halfnorm_ldd, 522
halfnorm_ldda, 522
halfnorm_lddd, 523
halfnorm_lddda, 523
halfnorm_logf, 524
halfnorm_logfdd, 524
halfnorm_logfddd, 525
halfnorm_loglik, 525
halfnorm_logscores, 526
halfnorm_means, 526
halfnorm_mu1f, 527
halfnorm_mu2f, 527
halfnorm_p1f, 528
halfnorm_p2f, 528
halfnorm_waic, 529

invgamma_cp, 529
invgamma_f1f, 536
invgamma_f1fa, 537
invgamma_f2f, 538
invgamma_f2fa, 538

invgamma_fd, 539
invgamma_fdd, 539
invgamma_ldd, 540
invgamma_ldda, 540
invgamma_lddd, 541
invgamma_lddda, 541
invgamma_lmn, 542
invgamma_lmnp, 542
invgamma_logf, 543
invgamma_logfdd, 544
invgamma_logfddd, 544
invgamma_loglik, 545
invgamma_logscores, 545
invgamma_mu1f, 546
invgamma_mu2f, 546
invgamma_p1f, 547
invgamma_p2f, 547
invgamma_waic, 548
invgauss_cp, 549
invgauss_f1f, 556
invgauss_f1fa, 557
invgauss_f2f, 557
invgauss_f2fa, 558
invgauss_fd, 558
invgauss_fdd, 559
invgauss_ldd, 559
invgauss_ldda, 560
invgauss_lddd, 560
invgauss_lddda, 561
invgauss_lmn, 561
invgauss_lmnp, 562
invgauss_logf, 562
invgauss_logfdd, 563
invgauss_logfddd, 563
invgauss_loglik, 564
invgauss_logscores, 564
invgauss_means, 565
invgauss_mu1f, 566
invgauss_mu2f, 566
invgauss_p1f, 567
invgauss_p2f, 567
invgauss_waic, 568

jpf2p, 569
jpf3p, 569
jpf4p, 570

lnorm_cp, 570
lnorm_dmgs_cp, 577

lnorm_dmgs_gg11, 583
lnorm_dmgs_gg12, 584
lnorm_dmgs_gg22, 584
lnorm_dmgs_loglik, 585
lnorm_dmgs_logscores, 585
lnorm_dmgs_means, 586
lnorm_dmgs_mu1f, 586
lnorm_dmgs_mu2f, 587
lnorm_dmgs_p1f, 587
lnorm_dmgs_p2f, 588
lnorm_dmgs_waic, 588
lnorm_f1f, 589
lnorm_f1fa, 590
lnorm_f2f, 590
lnorm_f2fa, 591
lnorm_fd, 591
lnorm_fdd, 592
lnorm_ldd, 592
lnorm_ldda, 593
lnorm_lddd, 593
lnorm_lddda, 594
lnorm_lmn, 594
lnorm_lmnp, 595
lnorm_logf, 595
lnorm_logfdd, 596
lnorm_logfddd, 596
lnorm_logscores, 597
lnorm_mu1fa, 597
lnorm_mu2fa, 598
lnorm_p1_cp, 599
lnorm_p1_f1f, 606
lnorm_p1_f1fa, 607
lnorm_p1_f2f, 607
lnorm_p1_f2fa, 608
lnorm_p1_fd, 608
lnorm_p1_fdd, 609
lnorm_p1_ldd, 609
lnorm_p1_ldda, 610
lnorm_p1_lddd, 611
lnorm_p1_lddda, 611
lnorm_p1_lmn, 612
lnorm_p1_lmnp, 613
lnorm_p1_logf, 613
lnorm_p1_logfdd, 614
lnorm_p1_logfddd, 615
lnorm_p1_loglik, 615
lnorm_p1_logscores, 616
lnorm_p1_mu1fa, 616
lnorm_p1_mu2fa, 617
lnorm_p1_p1fa, 617
lnorm_p1_p2fa, 618
lnorm_p1_pd, 618
lnorm_p1_pdd, 619
lnorm_p1_predictordata, 619
lnorm_p1_waic, 620
lnorm_p1fa, 598
lnorm_p2fa, 621
lnorm_pd, 621
lnorm_pdd, 622
lnorm_waic, 622
logis_cp, 623
logis_f1f, 630
logis_f1fa, 630
logis_f2f, 631
logis_f2fa, 631
logis_fd, 632
logis_fdd, 632
logis_ldd, 633
logis_ldda, 633
logis_lddd, 634
logis_lddda, 634
logis_lmn, 635
logis_lmnp, 635
logis_logf, 636
logis_logfdd, 637
logis_logfddd, 637
logis_loglik, 638
logis_logscores, 638
logis_mu1f, 639
logis_mu1fa, 639
logis_mu2f, 640
logis_mu2fa, 640
logis_p1_cp, 642
logis_p1_f1f, 649
logis_p1_f1fa, 650
logis_p1_f2f, 651
logis_p1_f2fa, 651
logis_p1_fd, 652
logis_p1_fdd, 652
logis_p1_ldd, 653
logis_p1_ldda, 654
logis_p1_lddd, 654
logis_p1_lddda, 655
logis_p1_lmn, 655
logis_p1_lmnp, 656
logis_p1_logf, 657

logis_p1_logfdd, 657
logis_p1_logfddd, 658
logis_p1_loglik, 658
logis_p1_logscores, 659
logis_p1_means, 659
logis_p1_mu1f, 660
logis_p1_mu1fa, 661
logis_p1_mu2f, 661
logis_p1_mu2fa, 662
logis_p1_p1f, 662
logis_p1_p1fa, 663
logis_p1_p2f, 664
logis_p1_p2fa, 664
logis_p1_pd, 665
logis_p1_pdd, 665
logis_p1_predictordata, 666
logis_p1_waic, 667
logis_p1f, 641
logis_p1fa, 641
logis_p2f, 668
logis_p2fa, 668
logis_pd, 669
logis_pdd, 669
logis_waic, 670
lst_k3_cp, 670
lst_k3_f1f, 677
lst_k3_f1fa, 678
lst_k3_f2f, 679
lst_k3_f2fa, 679
lst_k3_fd, 680
lst_k3_fdd, 680
lst_k3_ldd, 681
lst_k3_ldda, 681
lst_k3_lddd, 682
lst_k3_lddda, 682
lst_k3_lmn, 683
lst_k3_lmp, 683
lst_k3_logf, 684
lst_k3_logfdd, 685
lst_k3_logfddd, 685
lst_k3_loglik, 686
lst_k3_logscores, 686
lst_k3_mu1f, 687
lst_k3_mu2f, 687
lst_k3_p1f, 688
lst_k3_p2f, 688
lst_k3_waic, 689
lst_p1k3_cp, 690
lst_p1k3_f1f, 698
lst_p1k3_f1fa, 699
lst_p1k3_f2f, 699
lst_p1k3_f2fa, 700
lst_p1k3_fd, 701
lst_p1k3_fdd, 701
lst_p1k3_ldd, 702
lst_p1k3_ldda, 703
lst_p1k3_lddd, 703
lst_p1k3_lddda, 704
lst_p1k3_lmn, 705
lst_p1k3_lmp, 705
lst_p1k3_logf, 706
lst_p1k3_logfdd, 707
lst_p1k3_logfddd, 707
lst_p1k3_loglik, 708
lst_p1k3_logscores, 709
lst_p1k3_mu1f, 709
lst_p1k3_mu2f, 710
lst_p1k3_p1f, 711
lst_p1k3_p2f, 711
lst_p1k3_predictordata, 712
lst_p1k3_setics, 713
lst_p1k3_waic, 713
make_cwaic, 716
make_maic, 717
make_se, 717
make_waic, 718
makemuhat0, 714
makeq, 715
maket0, 715
maketa0, 716
man, 718
man1f, 728
man2f, 728
mandsub, 729
manf, 729
manldd, 736
manlddd, 736
manlenn, 736
manlennn, 737
manlogf, 737
manloglik, 737
manlogscores, 738
manmeans, 738
manpredictor, 738
manvector, 739
manwaic, 739

movexiawayfromzero, 739
ms_flat_1tail, 740
ms_flat_2tail, 741
ms_predictors_1tail, 742
ms_predictors_2tail, 743

nopdfcdfmsg, 744
norm_cp, 745
norm_dmgs_cp, 750
norm_dmgs_loglik, 756
norm_dmgs_logscores, 757
norm_dmgs_means, 757
norm_dmgs_mu1f, 758
norm_dmgs_mu2f, 759
norm_dmgs_p1f, 759
norm_dmgs_p2f, 760
norm_dmgs_waic, 760
norm_f1f, 761
norm_f1fa, 762
norm_f2f, 762
norm_f2fa, 763
norm_fd, 763
norm_fdd, 764
norm_gg, 764
norm_gmn, 765
norm_ldd, 765
norm_ldda, 766
norm_lddd, 766
norm_lddda, 767
norm_lmn, 767
norm_lmp, 768
norm_logf, 769
norm_logfdd, 769
norm_logfddd, 770
norm_logscores, 770
norm_ml_params, 771
norm_mu1fa, 771
norm_mu2fa, 772
norm_p1_cp, 773
norm_p1_f1f, 780
norm_p1_f1fa, 781
norm_p1_f2f, 781
norm_p1_f2fa, 782
norm_p1_fd, 783
norm_p1_fdd, 783
norm_p1_ldd, 784
norm_p1_ldda, 785
norm_p1_lddd, 785
norm_p1_lddda, 786

norm_p1_lmn, 787
norm_p1_lmp, 787
norm_p1_logf, 788
norm_p1_logfdd, 789
norm_p1_logfddd, 789
norm_p1_loglik, 790
norm_p1_logscores, 791
norm_p1_mlparams, 791
norm_p1_mu1fa, 792
norm_p1_mu2fa, 792
norm_p1_p1fa, 793
norm_p1_p2fa, 794
norm_p1_pd, 794
norm_p1_pdd, 795
norm_p1_predictordata, 796
norm_p1_waic, 796
norm_p1fa, 772
norm_p2fa, 797
norm_pd, 798
norm_pdd, 798
norm_unbiasedv_params, 799
norm_waic, 799

pareto_k2_cp, 800
pareto_k2_f1f, 806
pareto_k2_f1fa, 807
pareto_k2_f2f, 807
pareto_k2_f2fa, 808
pareto_k2_fd, 808
pareto_k2_fdd, 809
pareto_k2_l111, 809
pareto_k2_ldd, 810
pareto_k2_ldda, 811
pareto_k2_lddd, 811
pareto_k2_lddda, 812
pareto_k2_logf, 812
pareto_k2_logfdd, 813
pareto_k2_logfddd, 813
pareto_k2_logscores, 814
pareto_k2_ml_params, 814
pareto_k2_mu1fa, 815
pareto_k2_mu2fa, 815
pareto_k2_p1fa, 816
pareto_k2_p2fa, 816
pareto_k2_pd, 817
pareto_k2_pdd, 817
pareto_k2_waic, 818
pareto_p1k2_cp, 818
pareto_p1k2_f1f, 826

- pareto_p1k2_f1fa, [827](#)
- pareto_p1k2_f2f, [827](#)
- pareto_p1k2_f2fa, [828](#)
- pareto_p1k2_fd, [828](#)
- pareto_p1k2_fdd, [829](#)
- pareto_p1k2_ddd, [829](#)
- pareto_p1k2_ldda, [830](#)
- pareto_p1k2_ddd, [831](#)
- pareto_p1k2_ldddd, [831](#)
- pareto_p1k2_lmn, [832](#)
- pareto_p1k2_lmp, [832](#)
- pareto_p1k2_logf, [833](#)
- pareto_p1k2_logfdd, [834](#)
- pareto_p1k2_logfddd, [834](#)
- pareto_p1k2_loglik, [835](#)
- pareto_p1k2_logscores, [835](#)
- pareto_p1k2_means, [836](#)
- pareto_p1k2_mu1f, [837](#)
- pareto_p1k2_mu1fa, [838](#)
- pareto_p1k2_mu2f, [838](#)
- pareto_p1k2_mu2fa, [839](#)
- pareto_p1k2_p1f, [839](#)
- pareto_p1k2_p1fa, [840](#)
- pareto_p1k2_p2f, [840](#)
- pareto_p1k2_p2fa, [841](#)
- pareto_p1k2_pd, [841](#)
- pareto_p1k2_pdd, [842](#)
- pareto_p1k2_predictordata, [842](#)
- pareto_p1k2_waic, [843](#)
- pcauchy_cp (cauchy_cp), [28](#)
- pcauchy_p1, [844](#)
- pcauchy_p1_cp (cauchy_p1_cp), [45](#)
- pexp_cp (exp_cp), [116](#)
- pexp_p1, [845](#)
- pexp_p1_cp (exp_p1_cp), [130](#)
- pfrechet_k1_cp (frechet_k1_cp), [157](#)
- pfrechet_p2k1, [845](#)
- pfrechet_p2k1_cp (frechet_p2k1_cp), [180](#)
- pgamma_cp (gamma_cp), [209](#)
- pgev_cp (gev_cp), [230](#)
- pgev_k3_cp (gev_k3_cp), [245](#)
- pgev_p1, [846](#)
- pgev_p12, [846](#)
- pgev_p123, [847](#)
- pgev_p123_cp (gev_p123_cp), [277](#)
- pgev_p12_cp (gev_p12_cp), [321](#)
- pgev_p1_cp (gev_p1_cp), [374](#)
- pgev_p1k3, [848](#)
- pgev_p1k3_cp (gev_p1k3_cp), [349](#)
- pgnorm_k3_cp (gnorm_k3_cp), [405](#)
- pgpd_k1_cp (gpd_k1_cp), [437](#)
- pgumbel_cp (gumbel_cp), [462](#)
- pgumbel_p1, [848](#)
- pgumbel_p1_cp (gumbel_p1_cp), [481](#)
- phalfnorm_cp (halfnorm_cp), [510](#)
- pinvgamma_cp (invgamma_cp), [529](#)
- pinvgauss_cp (invgauss_cp), [549](#)
- plnorm_cp (lnorm_cp), [570](#)
- plnorm_dmgs_cp (lnorm_dmgs_cp), [577](#)
- plnorm_p1, [849](#)
- plnorm_p1_cp (lnorm_p1_cp), [599](#)
- plogis_cp (logis_cp), [623](#)
- plogis_p1, [849](#)
- plogis_p1_cp (logis_p1_cp), [642](#)
- plst_k3_cp (lst_k3_cp), [670](#)
- plst_p1k3, [850](#)
- plst_p1k3_cp (lst_p1k3_cp), [690](#)
- pnorm_cp (norm_cp), [745](#)
- pnorm_dmgs_cp (norm_dmgs_cp), [750](#)
- pnorm_p1, [850](#)
- pnorm_p1_cp (norm_p1_cp), [773](#)
- pnorm_p1_formula, [851](#)
- ppareto_k2_cp (pareto_k2_cp), [800](#)
- ppareto_p1k2, [851](#)
- ppareto_p1k2_cp (pareto_p1k2_cp), [818](#)
- punif_cp (unif_cp), [891](#)
- punif_formula, [852](#)
- pweibull_cp (weibull_cp), [897](#)
- pweibull_p2, [852](#)
- pweibull_p2_cp (weibull_p2_cp), [917](#)
- qcauchy_cp (cauchy_cp), [28](#)
- qcauchy_p1, [853](#)
- qcauchy_p1_cp (cauchy_p1_cp), [45](#)
- qexp_cp (exp_cp), [116](#)
- qexp_p1, [853](#)
- qexp_p1_cp (exp_p1_cp), [130](#)
- qfrechet_k1_cp (frechet_k1_cp), [157](#)
- qfrechet_p2k1, [854](#)
- qfrechet_p2k1_cp (frechet_p2k1_cp), [180](#)
- qgamma_cp (gamma_cp), [209](#)
- qgamma_k1_ppm, [854](#)
- qgamma_ppm, [856](#)
- qgev_cp (gev_cp), [230](#)
- qgev_k12_ppm, [857](#)
- qgev_k3_cp (gev_k3_cp), [245](#)
- qgev_mpd_ppm, [858](#)

- qgev_p1, 860
- qgev_p12, 860
- qgev_p123, 861
- qgev_p123_cp (gev_p123_cp), 277
- qgev_p12_cp (gev_p12_cp), 321
- qgev_p1_cp (gev_p1_cp), 374
- qgev_p1_ppm, 862
- qgev_p1k3, 862
- qgev_p1k3_cp (gev_p1k3_cp), 349
- qgev_ppm, 864
- qgnorm_k3_cp (gnorm_k3_cp), 405
- qgpd_k1_cp (gpd_k1_cp), 437
- qgpd_k1_ppm, 865
- qgumbel_cp (gumbel_cp), 462
- qgumbel_p1, 866
- qgumbel_p1_cp (gumbel_p1_cp), 481
- qhalfnorm_cp (halfnorm_cp), 510
- qinvgamma_cp (invgamma_cp), 529
- qinvgauss_cp (invgauss_cp), 549
- qlnorm_cp (lnorm_cp), 570
- qlnorm_dmgs_cp (lnorm_dmgs_cp), 577
- qlnorm_p1, 867
- qlnorm_p1_cp (lnorm_p1_cp), 599
- qllogis_cp (logis_cp), 623
- qllogis_p1, 868
- qllogis_p1_cp (logis_p1_cp), 642
- qlst_k3_cp (lst_k3_cp), 670
- qlst_p1k3, 868
- qlst_p1k3_cp (lst_p1k3_cp), 690
- qnorm_cp (norm_cp), 745
- qnorm_dmgs_cp (norm_dmgs_cp), 750
- qnorm_p1, 869
- qnorm_p1_cp (norm_p1_cp), 773
- qnorm_p1_formula, 869
- qntt_ppm, 870
- qpareto_k2_cp (pareto_k2_cp), 800
- qpareto_p1k2, 871
- qpareto_p1k2_cp (pareto_p1k2_cp), 818
- qunif_cp (unif_cp), 891
- qunif_formula, 872
- qweibull_cp (weibull_cp), 897
- qweibull_p2, 872
- qweibull_p2_cp (weibull_p2_cp), 917

- rcauchy_cp (cauchy_cp), 28
- rcauchy_p1_cp (cauchy_p1_cp), 45
- reltest, 873
- reltest2, 876
- reltest2_cases, 879
- reltest2_makeup, 879
- reltest2_plot, 880
- reltest2_predict, 881
- reltest2_simulate, 881
- reltest_makeup, 882
- reltest_makemaxep, 883
- reltest_predict, 883
- reltest_simulate, 885
- rexp_cp (exp_cp), 116
- rexp_p1_cp (exp_p1_cp), 130
- rfrechets_k1_cp (frechet_k1_cp), 157
- rfrechets_p2k1_cp (frechet_p2k1_cp), 180
- rgamma_cp (gamma_cp), 209
- rgev_cp (gev_cp), 230
- rgev_k3_cp (gev_k3_cp), 245
- rgev_minmax, 886
- rgev_p123_cp (gev_p123_cp), 277
- rgev_p123_minmax, 886
- rgev_p12_cp (gev_p12_cp), 321
- rgev_p12_minmax, 887
- rgev_p1_cp (gev_p1_cp), 374
- rgev_p1_minmax, 888
- rgev_p1k3_cp (gev_p1k3_cp), 349
- rgnorm_k3_cp (gnorm_k3_cp), 405
- rgpd_k1_cp (gpd_k1_cp), 437
- rgpd_k1_minmax, 889
- rgumbel_cp (gumbel_cp), 462
- rgumbel_p1_cp (gumbel_p1_cp), 481
- rhalfnorm_cp (halfnorm_cp), 510
- rhp_dmgs_cpmethod, 889
- rinvgamma_cp (invgamma_cp), 529
- rinvgauss_cp (invgauss_cp), 549
- rlnorm_cp (lnorm_cp), 570
- rlnorm_dmgs_cp (lnorm_dmgs_cp), 577
- rlnorm_p1_cp (lnorm_p1_cp), 599
- rlogis_cp (logis_cp), 623
- rlogis_p1_cp (logis_p1_cp), 642
- rlst_k3_cp (lst_k3_cp), 670
- rlst_p1k3_cp (lst_p1k3_cp), 690
- rnorm_cp (norm_cp), 745
- rnorm_dmgs_cp (norm_dmgs_cp), 750
- rnorm_p1_cp (norm_p1_cp), 773
- rpareto_k2_cp (pareto_k2_cp), 800
- rpareto_p1k2_cp (pareto_p1k2_cp), 818
- runif_cp (unif_cp), 891
- rust_pumethod, 890
- rweibull_cp (weibull_cp), 897
- rweibull_p2_cp (weibull_p2_cp), 917

- tcauchy_cp (cauchy_cp), 28
- tcauchy_p1_cp (cauchy_p1_cp), 45
- testppm_plot, 890
- texp_cp (exp_cp), 116
- texp_p1_cp (exp_p1_cp), 130
- tfrechet_k1_cp (frechet_k1_cp), 157
- tfrechet_p2k1_cp (frechet_p2k1_cp), 180
- tgamma_cp (gamma_cp), 209
- tgev_cp (gev_cp), 230
- tgev_k3_cp (gev_k3_cp), 245
- tgev_p123_cp (gev_p123_cp), 277
- tgev_p12_cp (gev_p12_cp), 321
- tgev_p1_cp (gev_p1_cp), 374
- tgev_p1k3_cp (gev_p1k3_cp), 349
- tgnorm_k3_cp (gnorm_k3_cp), 405
- tgpd_k1_cp (gpd_k1_cp), 437
- tgumbel_cp (gumbel_cp), 462
- tgumbel_p1_cp (gumbel_p1_cp), 481
- thalfnorm_cp (halfnorm_cp), 510
- tingamma_cp (invgamma_cp), 529
- tinvgauss_cp (invgauss_cp), 549
- tlnorm_cp (lnorm_cp), 570
- tlnorm_p1_cp (lnorm_p1_cp), 599
- tlogis_cp (logis_cp), 623
- tlogis_p1_cp (logis_p1_cp), 642
- tlst_k3_cp (lst_k3_cp), 670
- tlst_p1k3_cp (lst_p1k3_cp), 690
- tnorm_cp (norm_cp), 745
- tnorm_p1_cp (norm_p1_cp), 773
- tpareto_k2_cp (pareto_k2_cp), 800
- tpareto_p1k2_cp (pareto_p1k2_cp), 818
- tweibull_cp (weibull_cp), 897
- tweibull_p2_cp (weibull_p2_cp), 917
- unif_cp, 891
- weibull_cp, 897
- weibull_f1f, 904
- weibull_f1fa, 904
- weibull_f2f, 905
- weibull_f2fa, 905
- weibull_fd, 906
- weibull_fdd, 906
- weibull_ldd, 907
- weibull_ldda, 907
- weibull_lddd, 908
- weibull_lddda, 908
- weibull_lmn, 909
- weibull_lmnp, 909
- weibull_logf, 910
- weibull_logfdd, 911
- weibull_logfddd, 911
- weibull_loglik, 912
- weibull_logscores, 912
- weibull_means, 913
- weibull_mu1f, 913
- weibull_mu1fa, 914
- weibull_mu2f, 914
- weibull_mu2fa, 915
- weibull_p1f, 915
- weibull_p1fa, 916
- weibull_p2_cp, 917
- weibull_p2_f1f, 925
- weibull_p2_f1fa, 926
- weibull_p2_f2f, 926
- weibull_p2_f2fa, 927
- weibull_p2_fd, 927
- weibull_p2_fdd, 928
- weibull_p2_ldd, 928
- weibull_p2_ldda, 929
- weibull_p2_lddd, 930
- weibull_p2_lddda, 930
- weibull_p2_lmn, 931
- weibull_p2_lmnp, 932
- weibull_p2_logf, 932
- weibull_p2_logfdd, 933
- weibull_p2_logfddd, 934
- weibull_p2_loglik, 934
- weibull_p2_logscores, 935
- weibull_p2_means, 935
- weibull_p2_mu1f, 936
- weibull_p2_mu1fa, 937
- weibull_p2_mu2f, 937
- weibull_p2_mu2fa, 938
- weibull_p2_p1f, 938
- weibull_p2_p1fa, 939
- weibull_p2_p2f, 940
- weibull_p2_p2fa, 940
- weibull_p2_pd, 941
- weibull_p2_pdd, 941
- weibull_p2_predictordata, 942
- weibull_p2_waic, 943
- weibull_p2f, 916
- weibull_p2fa, 917
- weibull_pd, 944
- weibull_pdd, 944
- weibull_waic, 945