VanillaICE: Hidden Markov Models for the Assessment of Chromosomal Alterations using High-throughput SNP Arrays

Robert Scharpf

April 21, 2009

Abstract

Chromosomal DNA is characterized by variation between individuals at the level of entire chromosomes (e.g. aneuploidy in which the chromosome copy number is altered), segmental changes (including insertions, deletions, inversions, and translocations), and changes to small genomic regions (including single nucleotide polymorphisms). A variety of alterations that occur in chromosomal DNA, many of which can be detected using high density single nucleotide polymorphism (SNP) microarrays, are linked to normal variation as well as disease and therefore of particular interest. These include changes in copy number (deletions and duplications) and genotype (e.g. the occurrence of regions of homozygosity). Hidden Markov models (HMM) are particularly useful for detecting such abnormalities, modeling the spatial dependence between neighboring SNPs. Here, we extend previous approaches that utilize HMM frameworks for inference in high throughput SNP arrays by integrating copy number, genotype calls, and the corresponding measures of uncertainty when available. Using simulated and experimental data, we demonstrate how confidence scores control smoothing in a probabilistic framework.

1 Overview

This vignette describes how to fit a hidden Markov model to locus-level estimates of genotype or copy number. This vignette does not describe how to preprocess, genotype, or estimate copy number. We assume that locus-level estimates of genotype and/or copy number have been obtained by software such as the R package *crlmm* (preferred). Several HMM implementations are now available for the joint analysis of copy number and genotype, including QuantiSNP [2] and PennCNV [4].

Citing this software. Robert B Scharpf, Giovanni Parmigiani, Jonathan Pevsner, and Ingo Ruczinski. Hidden Markov models for the assessment of chromosomal alterations using high-throughput SNP arrays. *Annals of Applied Statistics*, 2(2):687–713, 2008.

2 Data organization

2.1 Basic data elements

The basic elements required for fitting the vanilla hidden Markov model in this vignette are

- a matrix of copy number estimates
- a matrix of genotype calls (represented as integers: 1=AA, 2=AB, 3=BB)
- a mapping of platform identifiers to physical position and chromosome

When available, uncertainty estimates of copy number and genotypes (CRLMM uncertainties) should also be represented as matrices. As several platforms now have polymorphic and nonpolymorphic probes, the dimensions of the genotype and copy number matrices do not necessarily coincide. However, for this release of *VanillaICE* we do impose this requirement. The genotypes for the nonpolymorphic probes can be represented as NAs. The assay data provided with this package contains the basic elements for the HMM from an older Affymetrix platform (100k):

```
> library(VanillaICE)
> data(locusLevelData)
> copynumber <- locusLevelData[["copynumber"]]/100
> genotypes <- locusLevelData[["genotypes"]]
```

The copy number estimates provided in the example data were saved as [copy number estimate]*100 and stored as an integer – we divided the estimates by 100 above to revert to the original scale. Genotypes should be represented as an integer: 1= AA, 2 = AB, 3=BB. NA or the integer 4 can be used to represent missing values. Confidence scores for the assay data are included in the `locusLevelData`, but for now we ignore these.

Meta-data on the locus-level estimates are also required. In particular, we require a mapping of the platform identifiers to the chromosome and physical position. The preferred representation for the chromosome is an integer. Currently, the only supported chromosomes are 1-22, X, and Y.

```
> chromosome <- locusLevelData[["annotation"]][, "chromosome"]
> position <- locusLevelData[["annotation"]][, "position"]
> names(position) <- names(chromosome) <- rownames(locusLevelData[["annotation"]])
```

Other chromosomes could potentially be supported, but the user will be expected to obtain centromere start and stop sites for alternative chromosomes. For details on how chromosome annotation should be organized, see

```
> require(SNPchip)
> data(chromosomeAnnotation, package = "SNPchip")
```

```
> all(c(identical(names(position), rownames(copynumber)), identical(names(position),
+     rownames(genotypes)), identical(colnames(genotypes), colnames(copynumber))))
```

```
[1] TRUE
```

The assay data elements should be ordered by chromosome and physical position:

```
> ordering <- order(chromosome, position)
> chromosome <- chromosome[ordering]
> position <- position[ordering]
> genotypes <- genotypes[ordering, , drop = FALSE]
> copynumber <- copynumber[ordering, , drop = FALSE]
```

2.2 Using *Biobase*-derived classes

A principal advantage of using classes instead of matrices is that the assay data elements are bound to the meta-data on the samples and loci. In this section, we show how the above assay data and meta-data can be encapsulated in a single object. The classes required for this operation are defined in the *oligoClasses* package available at BioConductor. These class definitions are all extensions of *eSet*. The methods for manipulating `eSets` in the R package *Biobase* are now available to us. We will instantiate these objects in two-steps: (i) instantiate an object that contains the locus-level meta-data and (ii) instantiate an object that contains the assay-data and the meta-data.

For (i), the `AnnotatedDataFrame` class defined in the *Biobase* package is used as a container for the meta-data on the locus-level summaries:

```
> require(oligoClasses)
> locusAnnotation <- data.frame(list(chromosome = chromosome, position = position),
+     row.names = names(chromosome))
> featuredata <- new("AnnotatedDataFrame", data = locusAnnotation,
+     varMetadata = data.frame(labelDescription = colnames(locusAnnotation)))
> stopifnot(all(c("chromosome", "position") %in% varLabels(featuredata)))
```

The labels 'chromosome' and 'position' are controlled vocabularies for the locus-level metadata that are later required when fitting the HMM. For (ii), three options are possible and depend on the assay data available:

1. SnpCallSet (genotypes-only)

   ```
   > new("SnpCallSet", calls = genotypes, phenoData = annotatedDataFrameFrom(genotypes,
   +     byrow = FALSE), featureData = featuredata)
   ```

2. SnpCopyNumberSet (copy number-only)

   ```
   > new("SnpCopyNumberSet", copyNumber = copynumber, phenoData = annotatedDataFrameFrom(copynumber,
   +     byrow = FALSE), featureData = featuredata)
   ```

3. oligoSnpSet (genotypes and copy number)

   ```
   > locusset <- new("oligoSnpSet", copyNumber = copynumber, calls = genotypes,
   +     phenoData = annotatedDataFrameFrom(copynumber, byrow = FALSE),
   +     featureData = featuredata)
   ```

Successfully instantiating an object of the class ensures that the assay data elements and the meta-data are consistent, greatly simplifying operations such as subsetting and reordering. For instance, the assay data and meta-data can be ordered by chromosome and physical position with one line of R code:

```
> locusset <- locusset[order(chromosome(locusset), position(locusset)),
+     ]
```

3 Fitting the Vanilla HMM

We use the Vanilla HMM to smooth the locus-level summaries as a function of physical position when confidence estimates of the assay data are unavailable. In Section 3.1, *Biobase*-derived class representations from the previous section and pre-defined wrappers for fitting a basic HMM are implemented. In Section 3.2, we reproduce the results from Section 3.1 but in a step-by-step fashion using only the matrices and vectors that were created in Section 2.1 (no classes). The step-by-step code in Section 2.1 is a useful guide for those wishing to alter the number of states in the HMM or develop other adaptations.

3.1 Using *Biobase*-derived classes

The objective of developing classes is to facilitate the process of fitting an HMM and to more effectively keep the assay- and meta-data bound in one object. The following code uses classes to arrive at the same HMM predictions as above. The starting point is the R object `locusset` created as indicated in Section 2.2. This object may be any one of the following classes: `SnpCallSet`, `SnpCopyNumberSet`, `oligoSnpSet`. Note that we could repeat the steps of parameterizing the HMM with transition- and emission-probabilities as described above, using the accessor methods `chromosome`, `position`, `copyNumber`, and `calls` to access the meta- or assay-data. We provide a convenience wrapper for fitting

the Vanilla HMM, `vanilla`, that computes the transition probabilities and emission probabilities. *For `oligoSnpSet` objects, the hidden states are assumed to be hemizygous deletion, normal, ROH, and amplification (in this order).*

```
> (brks <- hmm(object = locusset, states = c("hemDel", "normal",
+     "ROH", "amplification"), mu = c(1, 2, 2, 3), probs = c(0.999,
+     0.7, 0.999, 0.7), takeLog = TRUE))
```

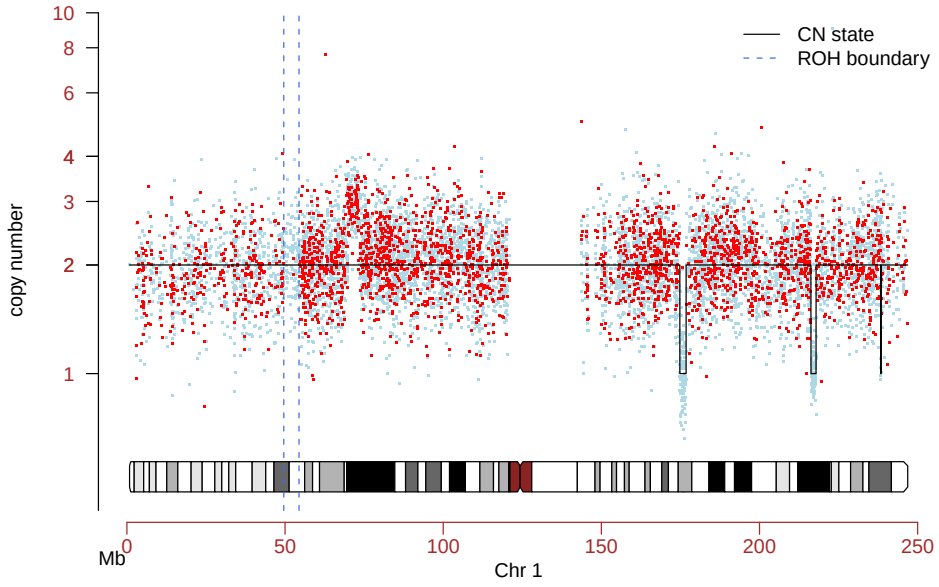
```
    sample chr     start       end   nbases nprobes         state
1  NA06993   1    836727  49545039 48708313     997        normal
2  NA06993   1  49597810  54409755  4811946     102           ROH
3  NA06993   1  54498950  69838068 15339119     902        normal
4  NA06993   1  69854466  73153900  3299435     202 amplification
5  NA06993   1  73174070 120928505 47754436    2502        normal
6  NA06993   1 143619946 174814292 31194347    1295        normal
7  NA06993   1 174815096 176800780  1985685     100        hemDel
8  NA06993   1 176800844 216239917 39439074    1900        normal
9  NA06993   1 216286002 217872810  1586809      99        hemDel
10 NA06993   1 217892177 238302668 20410492     901        normal
11 NA06993   1 238319943 238417864    97922       5        hemDel
12 NA06993   1 238429864 246860994  8431131     160        normal
13 NA06993   2    836727   7285897  6449171     100        normal
```

To obtain the matrix of predicted states for each locus, set `returnSegments` to `FALSE`:

```
> fit2 <- hmm(object = locusset, states = c("hemDel", "normal",
+     "ROH", "amplification"), mu = c(1, 2, 2, 3), probs = c(0.999,
+     0.7, 0.9999, 0.7), takeLog = TRUE, returnSegments = FALSE)
```

An example of how one may visualize the segmentation:

```
> show(plot(locusset[chromosome(locusset) == 1, ]))
> fit2[fit2 == 3] <- 2
> fit2[fit2 == 4] <- 3
> lines(position(locusset)[chromosome(locusset) == 1], fit2[chromosome(locusset) ==
+     1, 1])
> rohRegion <- as.integer(brks[brks$state == "ROH", ][c("start",
+     "end")])
> abline(v = rohRegion, col = "royalblue", lty = 2)
> legend("topright", lty = c(1, 2), col = c("black", "royalblue"),
+     legend = c("CN state", "ROH boundary"), bty = "n")
```



3.1.1 Extracting intermediate objects

To extract intermediate objects that were created by the `hmm` wrapper, one may pass an environment to store the intermediate objects that are created prior to fitting the HMM.

```
> env <- new.env()
> fit2 <- hmm(object = locusset, states = c("hemDel", "normal",
+     "ROH", "amplification"), mu = c(1, 2, 2, 3), probs = c(0.999,
+     0.7, 0.9999, 0.7), takeLog = TRUE, returnSegments = FALSE,
+     envir = env)
> ls(env)
```

```
 [1] "arm"            "emission"       "emission.cn"    "emission.gt"
 [5] "ice"            "initialP"       "locusset"       "mu"
 [9] "probs"          "returnSegments" "states"         "takeLog"
[13] "TAUP"           "transitionPr"   "verbose"
```

3.2 Step by step

We now go through the previous example step-by-step without using Biobase-derived classes or wrappers. While we reproduce the results in the previous section, one may modify any of these step (for instance, to expand the of hidden states). The basic elements of the HMM are

1 hidden states

2 initial state probabilities

3 transition probabilities

4 emission probabilities

Items (1) and (4) depend critically on the software that was used for obtaining the locus-level estimates and the scientific goals of the analysis.

3.2.1 Hidden states

In this section we discuss specification of the parameters for the hidden states. We assume that conditional on the underlying state the copy number estimates and genotypes are independent [3].

Copy number. There are two important considerations when specifying the hidden copy number states. First, the scale of the locus-level estimates are dependent on the preprocessing software. For instance, the BeadStudio software for the Illumina platform provides logR ratios, whereas *crlmm* provides an absolute estimate of copy number. Secondly, the data source is important. Integer copy number states are not appropriate when mixtures of different cell populations are present. Mixtures of cell populations can be diagnosed and the admixture estimated by visual inspection. For instance, a genomic region containing copy number estimates between 1 and 2 on the absolute scale with heterozygous genotype calls interspersed suggest a mixture of cell populations. Therefore, plot your data.

```
> chr1 <- annotation[, "chromosome"] == 1
> plot(annotation[chr1, "position"], copynumber[chr1, 1], pch = ".",
+     cex = 2, col = "grey60")
```

If genotypes are available, color-code by the diallelic genotype call.

```
> plot(annotation[chr1, "position"], copynumber[chr1, 1], pch = ".",
+     cex = 2, col = "grey60")
> het <- genotypes[, 1] == 2
> points(annotation[chr1 & !het, "position"], copynumber[chr1 &
+     !het, 1], pch = ".", cex = 2, col = "royalblue")
> points(annotation[chr1 & het, "position"], copynumber[chr1 &
+     het, 1], pch = ".", cex = 2, col = "red")
> abline(h = 1:3)
```

While the `locusLevelData` is based on HapMap samples, alterations were simulated to have integer copy number states. Therefore, in the code chunk below we specify an integer copy number for the hidden states hemizygous deletion (hemDel: copy number < 1), normal (copy number 2 and typical heterozygosity), region of homozygosity (ROH: copy number 2 and unusually low heterozygosity), and amplification (copy number $>= 3$).

```
> states <- c("hemDel", "normal", "ROH", "amplification")
> copynumberStates <- c(1, 2, 2, 3)
```

Genotypes In the absence of confidence scores for the genotype calls, the parameters needed for the hidden states for the genotypes is the probability that the true genotype is homozygous conditional on the underlying state.

```
> probs <- c(0.999, 0.7, 0.999, 0.7)
> names(probs) <- states
```

One may expand the number of hidden states to include > 3 copies and homozygous deletions, perform simple gain/loss analyses using copy number only, or assess regions of homozygosity with genotypes only.

3.2.2 Initial state probabilities.

The only requirement for specifying the initial state probabilities is that the probabilities sum to 1. Here, we assume *a priori* that the states are equally likely:

```
> initialP <- (rep(1, length(states)))/length(states)
```

3.2.3 Transition probabilities.

Our HMM uses locus-specific transition probabilities that are calculated as a function of the physical distance between loci. Specifically, the probability that the locus at position $t-1$ is *not* informative for the locus at position t is calculated as $1-e^{-d/C}$, where C is a constant specified by the user and d is the physical distance between locus t and locus $t-1$. The default for C is 1×10^8 and can be specified by the `TAUP` argument to the function `transitionProbability` to acheive a desired amount of sensitivity and specificity. Larger values of `TAUP` decreases the probability of transitioning to other states, and therefore provides a more smooth fit. In particular, `transitionProbability` (i) transforms the physical distance between adjacent loci to an estimate of the genomic distance and (ii) adds an 'arm' variable to the annotation matrix.

```
> tau <- transitionProbability(chromosome = chromosome, position = position,
+     TAUP = 1e+08)
> table(tau[, "arm"])

   0    1    2
4705 4460  100

> str(tau)

 num [1:9265, 1:4] 1 1 1 1 1 1 1 1 1 1 ...
 - attr(*, "dimnames")=List of 2
  ..$ : NULL
  ..$ : chr [1:4] "chromosome" "position" "arm" "transitionPr"
```

The values for `arm` indicate that the HMM will be fit independently to three separate segments of the supplied data, corresponding to arms 1p, 1q, and 2p. By default this function uses the chromosome annotation provided by the R package *SNPchip* for the centromere coordinates and will work only for chromosomes 1-22, X, and Y.

3.2.4 Emission probabilities

Copy number Skip to *Genotypes* if copy number estimates are not available.

One attractive feature of HMMs for smoothing copy number estimates as a function of the physical position is that uncertainty estimates can be incorporated into the emission probabilities. While uncertainty estimates for copy number and genotype calls can improve both the sensitivity and specificity for detecting chromosomal alterations [3], these estimates are not always available. However, we can develop ad-hoc estimates of the uncertainty provided a reasonable number of samples are available.

To obtain emission probabilities, we provide the matrix of copy number estimates and the location parameter for the hidden states (mu). The object `mu` should have length equal to the number of hidden states. If the hidden states are locus-specific, one may specify `mu` as a matrix with rows corresponding to loci and columns corresponding to states.

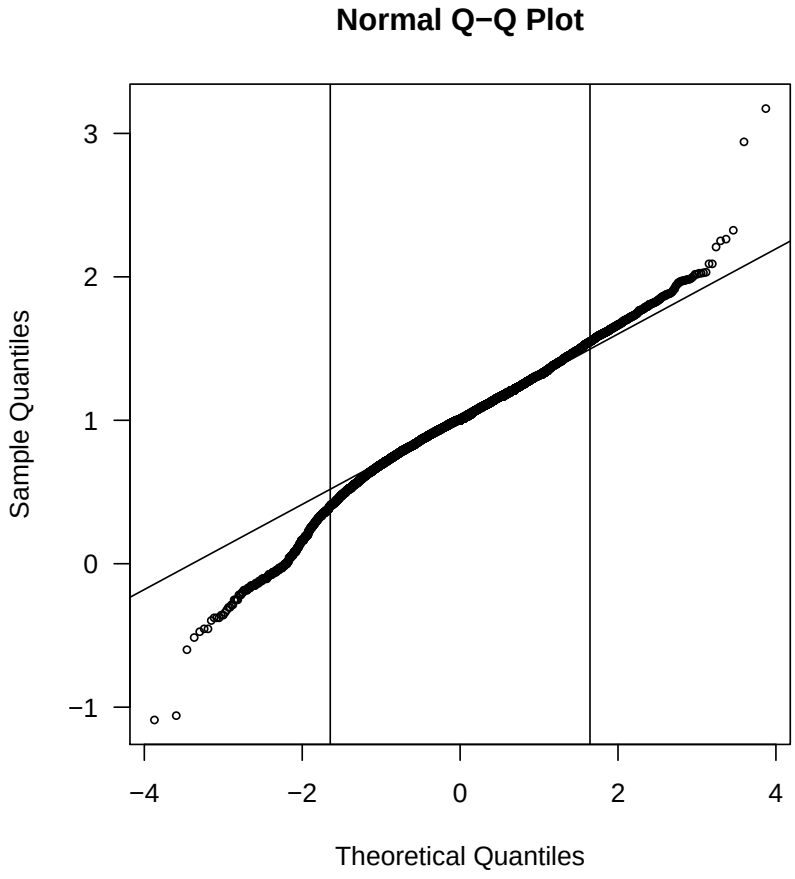
```
> copynumberStates <- c(1, 2, 2, 3)
> emission.cn <- copynumberEmission(copynumber = copynumber, states = states,
+     mu = copynumberStates, takeLog = TRUE, verbose = TRUE)
```

```
> copynumberStates <- matrix(c(1, 2, 2, 3), nrow(copynumber), length(states),
+     byrow = TRUE)
> emission.cn2 <- copynumberEmission(copynumber = copynumber, states = states,
+     mu = copynumberStates, takeLog = TRUE)
> stopifnot(identical(emission.cn, emission.cn2))
> dim(emission.cn)

[1] 9265    1    4
```

The third dimension of the array returned by `copynumberEmission` corresponds to the number of hidden states. The copy number and location parameter (mu) must be supplied on the same scale. For instance, in the above code chuck the copy number estimates and mu are on the scale of copy number. These estimates/parameters are both log-transformed within the body of the function by setting the argument `takeLog` to `TRUE`. Again, the emission probabilities are calculated under the assumption that the estimates, or a suitable transformation, are approximately Normal. Hence, in the simulated data we check that the middle 90% is approximately Gaussian.

```
> par(pty = "s", las = 1)
> qqnorm(log2(copynumber), cex = 0.6)
> qqline(log2(copynumber))
> abline(v = c(-1.645, 1.645))
```



When true uncertainty estimates for the copy number are not available, copy number outliers are more likely to result in extreme emission probabilities than can influence the HMM segmentation. There are several approaches to reducing the influence of outliers on the HMM segmentation: (i) improved uncertainty estimates (preferred), (ii) increase `TAUP` for the transition probabilities, (iii) threshold the emission probabilities, and (iv) threshold extreme values for the copy number estimates. For example,

```
> emission.cn[emission.cn[, , 2] < -10, , 2] <- -10
```

`copynumberEmission` estimates the scale parameter for the Normal distribution from the supplied data using the median absolute deviation (MAD). However, different standard deviations can be supplied by the user with the argument `sds`. The supplied `sds` must be of the same dimension as the copy number matrix. The following discussion elaborates briefly on the default procedure used to estimate the standard deviations.

In the example dataset, we have only one sample and no estimates of the copy number uncertainty. Therfore, we obtain a robust estimate of the copy number standard deviation across SNPs and use this as a rough estimate of the uncertainty. As the log-transformed copy number estimates are more nearly Gaussian, we calculate a robust estimate of the standard deviation using the median absolute deviation (MAD) on the log scale:

```
> cn.sds <- VanillaICE:::robustSds(copynumber, takeLog = TRUE)
> dim(cn.sds)

[1] 9265    1
```

When multiple samples are available (e.g., 10 or more), SNP-specific estimates of the copy number uncertainty can be obtained by scaling an estimate of the variability across samples by a sample-specific estimate of noise. In the following code chunk, we simulate a matrix of copy number for 200 samples and then compute a robust SNP-specific estimate of the standard deviation.

```
> CT <- matrix(sample(copynumber, 100 * 200, replace = TRUE), 100,
+     200)
> sds <- VanillaICE:::robustSds(CT, takeLog = TRUE)
```

The `robustSds` function returns a SNP-specific matrix of standard deviations provided that the copy number matrix has at least 3 samples. The *preferred* approach when the sample size is small (say, less than 10), is to develop SNP-specific estimates of the variance on a larger reference set, such as HapMap, using the same software, and then scale these estimates by a measure of the sample variance.

Genotypes. Proceed to *Fitting the hidden Markov model* if genotypes estimates are not available. If genotypes were estimated using the *CRLMM* or *oligo* and confidence estimates are available, see Section 4.

As copy number estimates are typically very noisy, genotype calls can potentially provide increased power to identify small regions of hemizygous deletions. In addition, the genotypes can be used to identify unusually long regions of homozygosity (ROH). In our experience, sequences of 70 - 100 homozygous genotypes are not uncommon and likely represent normal regions of the genome with fairly uniform haplotype structure. Emission probabilities for the Vanilla HMM are estimated from a Bernoulli with state-specific probabilities of a homozygous genotype call as specified in Section 3.2.1.

```
> emission.gt <- genotypeEmission(genotypes = genotypes, states = states,
+     probHomCall = probs)
```

If any of the genotype calls are missing and missingness is not independent of the underlying hidden state, one may specify the probability of a missing genotype calls for each hidden state (`probMissing`). By default, the HMM will assume that missing genotype calls are independent of the underlying hidden state. In particular, this assumption may not be reasonable for homozygous deletions.

Conditional on the hidden state, we assume that the copy number and genotype are independent. Therefore, the emission probabilities for an HMM that models the copy number and genotypes jointly are computed by adding the emission probabilities (log-scale) for copy number and genotype:

```
> emission <- emission.gt + emission.cn
```

If assay data for only genotypes or only copy number is available, the emission probability is simply the genotype or copy number emission probabilities, respectfully.

3.2.5 Viterbi algorithm.

The sequence of states that maximizes the likelihood is obtained by the Viterbi algorithm. Note that the argument `arm` to this function is a factor indicating the chromosomal arms – the Viterbi algorithm is computed for independently for each chromosomal arm.

```
> fit <- viterbi(initialStateProbs = log(initialP), emission = emission,
+     tau = tau[, "transitionPr"], arm = tau[, "arm"])
> table(fit)

fit
   1    2    3    4
 204 8757  102  202

> breaks(x = fit, states = states, position = tau[, "position"],
+     chromosome = tau[, "chromosome"], sampleNames = colnames(copynumber))

    sample chr     start       end   nbases nprobes         state
1  NA06993   1    836727  49545039 48708313     997        normal
2  NA06993   1  49597810  54409755  4811946     102           ROH
3  NA06993   1  54498950  69838068 15339119     902        normal
4  NA06993   1  69854466  73153900  3299435     202 amplification
5  NA06993   1  73174070 120928505 47754436    2502        normal
6  NA06993   1 143619946 174814292 31194347    1295        normal
7  NA06993   1 174815096 176800780  1985685     100        hemDel
8  NA06993   1 176800844 216239917 39439074    1900        normal
9  NA06993   1 216286002 217872810  1586809      99        hemDel
10 NA06993   1 217892177 238302668 20410492     901        normal
11 NA06993   1 238319943 238417864    97922       5        hemDel
12 NA06993   1 238429864 246860994  8431131     160        normal
13 NA06993   2    836727   7285897  6449171     100        normal
```

4 Integretating Confidence Estimates (ICE): Extending *CRLMM*

In this section, we illustrate how one may fit an HMM that incorporates confidence esimates of the SNP-level summaries for the genotype calls. To reduce the amount of repeated code, we will use the *Biobase*-derived classes to illustrate this approach. We will instantiate an object of class `oligoSnpSet` as described in Section 2.2, however we will also store the CRLMM confidence scores (represented as an integer).

```
> copynumber <- locusLevelData[["copynumber"]]/100
> crlmmConfidence <- locusLevelData[["crlmmConfidence"]]
> genotypes <- locusLevelData[["genotypes"]]
> fd <- locusLevelData[["annotation"]]
> featuredata <- new("AnnotatedDataFrame", data = data.frame(fd),
+     varMetadata = data.frame(labelDescription = colnames(fd)))
> (locusset2 <- new("oligoSnpSet", copyNumber = copynumber, calls = genotypes,
+     callsConfidence = crlmmConfidence, phenoData = annotatedDataFrameFrom(copynumber,
+         byrow = FALSE), featureData = featuredata, annotation = "genomewidesnp6"))

oligoSnpSet (storageMode: lockedEnvironment)
assayData: 9265 features, 1 samples
  element names: calls, callsConfidence, cnConfidence, copyNumber
experimentData: use 'experimentData(object)'
Annotation: genomewidesnp6
phenoData
An object of class "AnnotatedDataFrame"
  sampleNames: NA06993
  varLabels and varMetadata description: none
featureData
An object of class "AnnotatedDataFrame"
  featureNames: SNP_A-1677174, SNP_A-1718890, ..., SNP_A-100  (9265 total)
  varLabels and varMetadata description:
    chromosome: chromosome
    position: position
Annotation [1] "genomewidesnp6"

> locusset2 <- locusset2[order(chromosome(locusset2), position(locusset2)),
+     ]
```

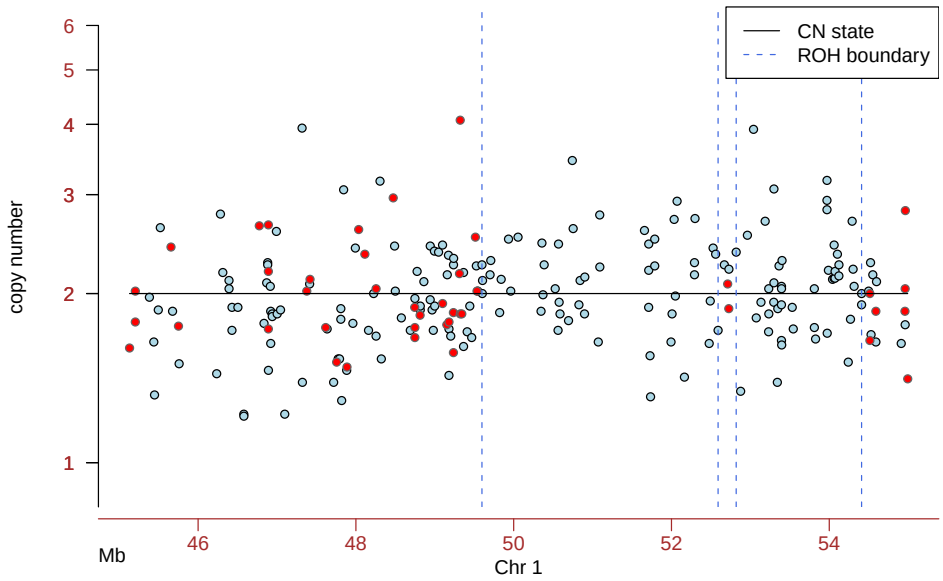
The `annotation` slot must be specified so that the appropriate reference distribution will be loaded. Supported Affymetrix platforms include genomewidesnp6, pd.mapping250k.nsp, and pd.mapping250k.sty. When fitting the ICE HMM using the `hmm` wrapper, we again assume that the hidden states are hemizygous deletion, normal, copy-neutral region of homozygosity, and amplification (in this order). Fitting the ICE HMM using other states is possible, and one could follow the step-by-step approach outlined in Section 3.2.

```
> fit3 <- hmm(object = locusset2, states = c("hemDel", "normal",
+     "ROH", "amplification"), mu = c(1, 2, 2, 3), takeLog = TRUE,
+     returnSegments = FALSE, ice = TRUE)
> brks <- hmm(object = locusset2, states = c("hemDel", "normal",
+     "ROH", "amplification"), probs = c(0.05, 0.99, 0.7, 0.999),
+     mu = c(1, 2, 2, 3), takeLog = TRUE, ice = TRUE)
```

Here we plot one of the regions that has different breakpoints in the ICE and Vanilla HMMs.

```
> i <- which(position(locusset2) >= 45 * 1e+06 & position(locusset2) <=
+     55 * 1e+06 & chromosome(locusset2) == 1)
> gp <- plot(locusset2[i, ])
> gp$pch <- 21
> gp$col <- "black"
> gp$cex <- 0.9
```

```
> gp$ylim <- c(0.9, 6)
> gp$add.cytoband <- FALSE
> show(gp)
> fit3[fit3 == 3] <- 2
> fit3[fit3 == 4] <- 3
> lines(position(locusset2)[i], fit3[i, 1])
> rohRegion <- as.numeric(as.matrix(brks[brks$state == "ROH", ][c("start",
+     "end")]))
> abline(v = rohRegion, col = "royalblue", lty = 2)
> legend("topright", lty = c(1, 2), col = c("black", "royalblue"),
+     legend = c("CN state", "ROH boundary"), bty = "o", bg = "white")
```



5 Appendix

5.1 Confidence scores for genotype calls.

We suggest using the `CRLMM` algorithm [1] for genotype calls. `CRLMM` (in the R package *oligo*) provides confidence scores ($\mathrm{S}_{\widehat{\mathrm{GT}}}$) of the genotype estimates ($\widehat{\mathrm{GT}}$). From 269 HapMap samples assayed on the Affymetrix 50k Xba and Hind chips, we have a gold standard of the true genotype defined by the consensus of the HapMap centers. We use kernal-based density estimates to obtain

$$f\left\{\mathrm{S}_{\widehat{\mathrm{HOM}}} \mid \widehat{\mathrm{HOM}}, \mathrm{HOM}\right\},\ f\left\{\mathrm{S}_{\widehat{\mathrm{HOM}}} \mid \widehat{\mathrm{HOM}}, \mathrm{HET}\right\},\ f\left\{\mathrm{S}_{\widehat{\mathrm{HET}}} \mid \widehat{\mathrm{HET}}, \mathrm{HOM}\right\},\quad \text{and } f\left\{\mathrm{S}_{\widehat{\mathrm{HET}}} \mid \widehat{\mathrm{HET}}, \mathrm{HET}\right\} \tag{1}$$

separately for the Xba and Hind 50k chips. The first term in (1), for example, denotes the density of the scores when the genotype is correctly called homozygous ($\widehat{\mathrm{HOM}}$) and the true genotype is homozygous (HOM). See [3] for a more complete description of the methods.

```
> gp <- plot(snpset[chromosome(snpset) == 1, ])
> lines(position(snpset)[chromosome(snpset) == 1], viterbiResults[chromosome(snpset) ==
+     1, ])
> gp$abline.v <- TRUE
> allParameters <- unlist(snpPar(gp))
> gp$col.predict[3] <- "white"
> gp$hmm.ycoords <- c(0.7, 0.9)
> show(gp)
```

5.2 More examples

5.2.1 Copy number only: assessing gain/loss

The method `hmm` has a different set of underlying hidden states depending on whether copy number estimates, genotype calls, or both are available. When only copy number estimates are available, the hidden states (for autosomes) are hemizygous or homozygous deletion (one or fewer copies), normal (two copies), and amplification (three or more copies). The corresponding Biobase-derived class is SnpCopyNumberSet.

5.2.2 Genotypes only: assessing ROH

When only genotype calls are available, the hidden states are loss and retention (ret) of heterozygosity. We define *loss* to be a sequence of homozygous SNPs longer than what we would expect to observe by chance. Note that many long stretches of homozygosity may occur as a result of a population sharing a common underlying haplotype structure; loss predictions from an HMM fit to an indvidual do not necessarily reflect the 'loss' of an allele in that individual. The corresponding Biobase-derived class is SnpCallSet.

5.3 Expanding the number of hidden states

Section is incomplete. For homozygous deletion, we specify a small value (less than 1) but greater than zero to account for background fluorescence.

```
> states <- c("homozygousDeletion", "hemizygousDeletion", "normal",
+     "LOH", "3copyAmp", "4copyAmp")
> mu <- c(0.05, 1, 2, 2, 3, 4)
> probs <- c(0.99, 0.999, 0.99, 0.999, 0.99, 0.99)
> probMissing <- c(0.95, rep(0.5, 5))
```

5.4 Classes for segmented data

TODO

5.5 Plotting methods for segmented data classes

TODO

6 Session Information

The version number of R and packages loaded for generating the vignette were:

- R version 2.9.0 (2009-04-17), `i386-pc-mingw32`

- Locale: `LC_COLLATE=English_United States.1252;LC_CTYPE=English_United States.1252;LC_MONETARY=Engli`
- Base packages: base, datasets, graphics, grDevices, methods, stats, tools, utils
- Other packages: Biobase 2.4.0, oligoClasses 1.6.0, SNPchip 1.8.0, VanillaICE 1.6.0
- Loaded via a namespace (and not attached): Biostrings 2.12.0, IRanges 1.2.0

References

[1] Benilton Carvalho, Henrik Bengtsson, Terence P Speed, and Rafael A Irizarry. Exploration, normalization, and genotype calls of high-density oligonucleotide SNP array data. *Biostatistics*, 8(2):485–499, Apr 2007.

[2] Stefano Colella, Christopher Yau, Jennifer M Taylor, Ghazala Mirza, Helen Butler, Penny Clouston, Anne S Bassett, Anneke Seller, Christopher C Holmes, and Jiannis Ragoussis. QuantiSNP: an Objective Bayes Hidden-Markov Model to detect and accurately map copy number variation using SNP genotyping data. *Nucleic Acids Res*, 35(6):2013–2025, 2007.

[3] Robert B Scharpf, Giovanni Parmigiani, Jonathan Pevsner, and Ingo Ruczinski. Hidden Markov models for the assessment of chromosomal alterations using high-throughput SNP arrays. *Annals of Applied Statistics*, 2(2):687–713, 2008.

[4] Kai Wang, Mingyao Li, Dexter Hadley, Rui Liu, Joseph Glessner, Struan F A Grant, Hakon Hakonarson, and Maja Bucan. Penncnv: an integrated hidden markov model designed for high-resolution copy number variation detection in whole-genome snp genotyping data. *Genome Res*, 17(11):1665–1674, Nov 2007.