Package ‘ropenblas’

October 14, 2022

Type Package

Title Download, Compile and Link 'OpenBLAS' Library with R

Version 0.3.0

Maintainer Pedro Rafael D. Marinho <pedro.rafael.marinho@gmail.com>

Description The 'ropenblas' package (<https://prdm0.github.io/ropenblas/>) is useful for users of any 'GNU/Linux' distribution. It will be possible to download, compile and link the 'OpenBLAS' library (<https://www.openblas.net/>) with the R language, always by the same procedure, regardless of the 'GNU/Linux' distribution used. With the 'ropenblas' package it is possible to download, compile and link the latest version of the 'OpenBLAS' library even the repositories of the 'GNU/Linux' distribution used do not include the latest versions of 'OpenBLAS'. If of interest, older versions of the 'OpenBLAS' library may be considered. Linking R with an optimized version of 'BLAS' (<https://netlib.org/blas/>) may improve the computational performance of R code. The 'OpenBLAS' library is an optimized implementation of 'BLAS' that can be easily linked to R with the 'ropenblas' package.

Depends R (>= 3.1.0)

License GPL-3

URL https://prdm0.github.io/ropenblas/,
https://github.com/prdm0/ropenblas

BugReports https://github.com/prdm0/ropenblas/issues

SystemRequirements GNU Make, GCC Compiler Suite (C and Fortran)

Encoding UTF-8

Imports glue, magrittr, getPass, rstudioapi, stringr, git2r, RCurl, XML, cli, pingr, withr, rlang, fs, rvest

RoxygenNote 7.2.1

NeedsCompilation no

Author Pedro Rafael D. Marinho [aut, cre] (<https://orcid.org/0000-0003-1591-8300>), Enes Ahmeti [ctb]

Repository CRAN

Date/Publication 2022-08-29 17:50:02 UTC

R topics documented:

`last_version_openblas` *OpenBLAS library versions*

Description

OpenBLAS library versions

Usage

```
last_version_openblas()
```

Details

This function automatically searches **OpenBLAS** library versions in the official **GitHub** project.

1. `last_version`: Returns the latest stable version of the **OpenBLAS** library.
2. `versions`: All stable versions of the **OpenBLAS** library.
3. `n`: Total number of versions.

See Also

`last_version_r`, `ropenblas`, `rcompiler`

Examples

```
# last_version_openblas()
```

`last_version_r` R *language versions*

Description

R language versions

Usage

```
last_version_r(major = NULL)
```

Arguments

`major`	Major release number of R language (e.g. `1L`, `2L`, `3L`, ...). If `major = NULL`, the function will consider the major release number.

Details

This function automatically searches R language versions in the official language repositories. That way, doing `last_version_r(major = NULL)` you will always be well informed about which latest stable version the R language is in. You can also set the higher version and do a search on the versions of the R language whose major version was `1L` or `2L`, for example.

Value

A list of two named elements will be returned. Are they:

1. `last_version`: Returns the latest stable version of the language given a major version (major version). If `major = NULL`, the latest stable version of the language will be returned based on the set of all language versions.
2. `versions`: Character vector with all language versions based on a major version (higher version). If `major = NULL`, `versions` will be a vector with the latest language versions.
3. `n`: Total number of versions of R based on major version. If `major = NULL`, `versions` will be a vector with the latest language versions.

See Also

`ropenblas`, `rcompiler`

Examples

```
# last_version_r(major = NULL)
```

link_again *Linking the OpenBLAS library with R again*

Description

The link_again function links again the **OpenBLAS** library with the R language, being useful to correct problems of untying the **OpenBLAS** library that is common when the operating system is updated.

Usage

```
link_again(restart_r = TRUE)
```

Arguments

restart_r	If TRUE (default), a new section of R is started after linking the **OpenBLAS** library.

Details

The function link_again be able to link again the R language with the **OpenBLAS** library. Thus, link_again will only make the relinkagem when in some previous section of R the ropenblas function has been used for the initial binding of the R language with the **OpenBLAS** library.

Relinking is useful in situations of updating the operating system. In some update, it is possible that the **OpenBLAS** library compiled in the /opt directory is unlinked. In this scenario, when the **OpenBLAS** library has already been compiled using the ropenblas function, the link_again function performs a new link without the need to recompile, thus making the process less time consuming.

Note

In situations where there was a disconnection due to an update of the operating system, the ropenblas function can be used to re-link the OpenBLAS library with the R language, however, it will be necessary to compile the **OpenBLAS** library again. If you are interested in recompiling the **OpenBLAS** library and linking with R, use the ropenblas function. If the interest is to take advantage of a previous compilation of the **OpenBLAS** library, the function link_again may be useful.

See Also

ropenblas

Examples

```
# link_again()
```

rcompiler *Compile a version of* R *on GNU/Linux systems*

Description

This function is responsible for compiling a version of the R language.

Usage

```
rcompiler(x = NULL, with_blas = NULL, complementary_flags = NULL)
```

Arguments

`x` Version of R you want to compile. By default (`x = NULL`) will be compiled the latest stable version of the R language. For example, `x = "3.6.2"` will compile and link **R-3.6.2** version as the major version on your system.

`with_blas` String, `--with-blas = NULL` by default, with flags for `--with-blas` used in the R compilation process. Details on the use of this flag can be found **here**.

`complementary_flags` String, `complementary_flags = NULL` by default, for adding complementary flags in the **R** language compilation process.

Details

This function is responsible for compiling a version of the **R** language. The `x` argument is the version of **R** that you want to compile. For example, `x = "4.0.0"` will compile and link **R-4.0.0** version as the major version on your system. By default (`x = NULL`) will be compiled the latest stable version of the **R**.

For example, to compile the latest stable version of the **R** language, do:

```
rcompiler()
```

Regardless of your GNU/Linux distribution and what version of **R** is in your repositories, you can have the latest stable version of the **R** language compiled into your computer architecture.

You can use the `rcompiler()` function to compile different versions of **R**. For example, running `rcompiler(x = "3.6.3")` and `rcompiler()` will install versions 3.6.3 and 4.0.0 on its GNU/Linux distribution, respectively. If you are in version 4.0.0 of **R** and run the code `rcompiler(x = "3.6.3")` again, the function will identify the existence of version 3.6.3 in the system and give you the option to use the binaries that were built in a previous compilation. This avoids unnecessarys compilations.

In addition to the `x` argument, the `rcompiler()` function has two other arguments that will allow you to change and pass new compilation flags. Are they:

1. `with_blas`: This argument sets the `--with-blas` flag in the R language compilation process and must be passed as a string. Details on the use of this flag can be found **here**. If `with_blas = NULL` (default), then it will be considered:

```
./configure --prefix=/opt/R/version_r --enable-memory-profiling --enable-R-shlib
 --enable-threads=posix --with-blas="-L/opt/OpenBLAS/lib -I/opt/OpenBLAS/include
 -lpthread -lm"
```

   Most likely, you will have little reason to change this aprgument. Unless you know what you're doing, consider `with_blas = NULL`. Do not change the installation directory, that is, always consider `--prefix = /opt/R/version_r`, where `version_r` is a valid version of **R**. For a list of valid versions of **R**, run the `last_version_r()`. Installing **R** in the `/opt/R/version_r` directory is important because some functions in the package require this. Both the **R** language and the **OpenBLAS** library will be installed in the `/opt` directory. If this directory does not exist in your GNU/Linux distribution, it will be created;

2. `complementary_flags`: String (`complementary_flags = NULL` by default) for adding complementary flags in the **R** language compilation process. Passing a string to `complementary_flags` will compile it in the form:

```
./configure --with-blas="..." complementary_flags
```

Value

Returns a warning message informing you if the procedure occurred correctly. You will also be able to receive information about missing dependencies.

See Also

`ropenblas`, `last_version_r`

Examples

```
# rcompiler()
```

rnews *R News file*

Description

Returns the contents of the NEWS.html file in the standard browser installed on the operating system.

Usage

```
rnews(pdf = FALSE, dev = FALSE)
```

Arguments

`pdf` If `FALSE` (default), the NEWS.html file will open in the browser, otherwise NEWS.pdf will be opened.

`dev` If `FALSE` (default), it will not show changes made to the language development version. To see changes in the development version, do `dev = TRUE`.

Details

The NEWS.html file contains the main changes from the recently released versions of the R language. The goal is to facilitate the query by invoking it directly from the R command prompt. The rnews function is analogous to the news function of the **utils** package. However, using the news command in a terminal style bash shell is possible to receive a message like:

```
news()
starting httpd help server ... done
Error in browseURL(url) : 'browser' must be a non-empty character string
```

This is an error that may occur depending on the installation of R. Always prefer the use of the news function but if you need to, use the rnews function.

`ropenblas` *Download, Compile and Link OpenBLAS Library with* R

Description

Link R with an optimized version of the **BLAS** library (**OpenBLAS**).

Usage

```
ropenblas(x = NULL, restart_r = TRUE)
```

Arguments

`x` **OpenBLAS** library version to be considered. By default, `x = NULL`.

`restart_r` If `TRUE`, a new section of R is started after compiling and linking the **OpenBLAS** library.

Details

The `ropenblas()` function will only work on Linux systems. When calling the `ropenblas()` function on Windows, no settings will be made. Only a warning message will be issued informing you that the configuration can only be performed on Linux systems.

The function will automatically download the latest version of the **OpenBLAS** library. However, it is possible to inform olds versions to the single argument of `ropenblas()`. The `ropenblas()` function downloads, compiles and link R to use of the **OpenBLAS** library. Everything is done very simply, just loading the library and invok the function `ropenblas()`.

Considering using the **OpenBLAS** library rather than the **BLAS** may bring extra optimizations for your code and improved computational performance for your simulations, since **OpenBLAS** is an optimized implementation of the library **BLAS**.

You must install the following dependencies on your operating system (Linux):

1. **GNU Make**;

2. **GNU GCC Compiler (C and Fortran)**.

Your linux operating system may already be configured to use the **OpenBLAS** library. Therefore, most likely R will already be linked to this library. To find out if the R language is using the **OpenBLAS** library, at R, do:

```
extSoftVersion()["BLAS"]
```

If R is using the **OpenBLAS** library, something like `/any_directory/libopenblas.so` should be returned. Therefore, there should be the name openblas in the **s**hared **o**bject returned (file extension **.so**).

If the `ropenblas()` function can identify that the R language is using the version of **OpenBLAS** you wish to configure, a warning message will be returned asking if you really would like to proceed with the configuration again.

The `ropenblas()` function will download the desired version of the library **OpenBLAS**, compile and install the library in the `/opt` directory of your operational system. If the directory does not exist, it will be created so that the installation can be completed. Subsequently, files from the version of **BLAS** used in R will be symbolically linked to the shared object files of the library version **OpenBLAS** compiled and installed in `/opt`.

You must be the operating system administrator to use this library. Therefore, do not attempt to use it without telling your system administrator. If you have the ROOT password, you will be responsible for everything you do on your operating system. Other details you may also find **here**.

Value

Returns a warning message informing you if the procedure occurred correctly. You will also be able to receive information about missing dependencies.

Note

You do not have to in every section of R make use of the `ropenblas()` function. Once the function is used, R will always consider using the **OpenBLAS** library in future sections.

Author(s)

Pedro Rafael D. Marinho (e-mail: <pedro.rafael.marinho@gmail.com>)

See Also

`rcompiler`, `last_version_r`

Examples

```
# ropenblas()
```

Index