

Package ‘RcppArmadillo’

April 13, 2021

Type Package

Title 'Rcpp' Integration for the 'Armadillo' Templated Linear Algebra Library

Version 0.10.4.0.0

Date 2021-04-11

Author Dirk Eddelbuettel, Romain Francois, Doug Bates and Binxiang Ni

Maintainer Dirk Eddelbuettel <edd@debian.org>

Description 'Armadillo' is a templated C++ linear algebra library (by Conrad Sanderson) that aims towards a good balance between speed and ease of use. Integer, floating point and complex numbers are supported, as well as a subset of trigonometric and statistics functions. Various matrix decompositions are provided through optional integration with LAPACK and ATLAS libraries. The 'RcppArmadillo' package includes the header files from the templated 'Armadillo' library. Thus users do not need to install 'Armadillo' itself in order to use 'RcppArmadillo'. From release 7.800.0 on, 'Armadillo' is licensed under Apache License 2; previous releases were under licensed as MPL 2.0 from version 3.800.0 onwards and LGPL-3 prior to that; 'RcppArmadillo' (the 'Rcpp' bindings/bridge to Armadillo) is licensed under the GNU GPL version 2 or later, as is the rest of 'Rcpp'. Armadillo requires a C++11 compiler.

License GPL (>= 2)

LazyLoad yes

Depends R (>= 3.3.0)

LinkingTo Rcpp

Imports Rcpp (>= 0.11.0), stats, utils, methods

Suggests tinytest, Matrix (>= 1.3.0), pkgKitten, reticulate, slam

URL <https://github.com/RcppCore/RcppArmadillo>

BugReports <https://github.com/RcppCore/RcppArmadillo/issues>

NeedsCompilation yes

Repository CRAN

Date/Publication 2021-04-13 15:00:08 UTC

R topics documented:

RcppArmadillo-package	2
armadillo_set_seed	3
armadillo_set_seed_random	4
armadillo_version	5
fastLm	6
RcppArmadillo.package.skeleton	8

Index	10
--------------	-----------

RcppArmadillo-package *R and Armadillo Integration*

Description

The package brings the power of Armadillo to R.

Armadillo

Armadillo is a C++ linear algebra library, aiming towards a good balance between speed and ease of use.

It provides efficient classes for vectors, matrices and cubes, as well as many functions which operate on the classes (eg. contiguous and non-contiguous submatrix views).

Various matrix decompositions are provided, and an automatic expression evaluator (via template meta-programming) combines several operations to increase efficiency.

The syntax (API) is deliberately similar to Matlab. It is useful for algorithm development directly in C++, or quick conversion of research code into production environments.

Armadillo has been primarily developed at NICTA (Australia) by Conrad Sanderson, with contributions from around the world.

RcppArmadillo

RcppArmadillo acts as a bridge between Rcpp and Armadillo, allowing the programmer to write code using Armadillo classes that integrate seamlessly with R via Rcpp.

Using RcppArmadillo

The simplest way to get started is to create a skeleton of a package using RcppArmadillo. This can be done conveniently by the [RcppArmadillo.package.skeleton](#) function.

The important steps are

- Include the RcppArmadillo.h header file, which also includes armadillo.h.
- Import Rcpp, and LinkingTo Rcpp and RcppArmadillo by adding these lines to the DESCRIPTION file:

```
Imports: Rcpp (>= 0.11.0)
LinkingTo: Rcpp, RcppArmadillo
```

- Link against the BLAS and LAPACK libraries, by adding this line in the Makevars and Makevars.win files:

```
PKG_LIBS = $(LAPACK_LIBS) $(BLAS_LIBS) $(FLIBS)
```

Support

Please use the Rcpp-devel mailing list on r-forge for questions about RcppArmadillo (subscribe first). <http://lists.r-forge.r-project.org/cgi-bin/mailman/listinfo/rcpp-devel>

Author(s)

For RcppArmadillo: Dirk Eddelbuettel, Romain Francois, Doug Bates and Binxiang Ni

Maintainer: Dirk Eddelbuettel <edd@debian.org>

For Armadillo: Conrad Sanderson

References

Armadillo project: <http://arma.sourceforge.net/>

Conrad Sanderson and Ryan Curtin. [Armadillo: a template-based C++ library for linear algebra](#). Journal of Open Source Software, Vol. 1, pp. 26, 2016.

Dirk Eddelbuettel and Conrad Sanderson, "RcppArmadillo: Accelerating R with high-performance C++ linear algebra", Computational Statistics and Data Analysis, 2014, 71, March, pages 1054-1063, doi: [10.1016/j.csda.2013.02.005](https://doi.org/10.1016/j.csda.2013.02.005).)

armadillo_set_seed	<i>Set the Armadillo Random Number Generator to the given value</i>
--------------------	---

Description

Set the Armadillo Random Number Generator to the given value

Usage

```
armadillo_set_seed(val)
```

Arguments

val	The seed used to initialize Armadillo's random number generator.
-----	--

Details

Depending on whether RcppArmadillo was compiled for the C++98 standard (currently the default) or for C++11 (optional), two different RNGs may be used. This function resets either. For C++98, the R programming language's RNG is used. For C++11, the RNG included in the `<random>` library is used only when `#define ARMA_USE_CXX11_RNG` is placed before `#include <RcppArmadillo.h>`. Otherwise, the R programming language's RNG will be used.

Value

The function is invoked for its side effect and has no return value.

Note

This has been found to not work as expected in **RStudio** as its code also uses the system RNG library. You may have to either not run within **RStudio** or change your code to use a different RNG such as the one from R.

See Also

The R documentation on its RNGs all of which are accessible via **Rcpp**.

armadillo_set_seed_random

Set the Armadillo Random Number Generator to a random value

Description

Set the Armadillo Random Number Generator to a random value

Usage

```
armadillo_set_seed_random()
```

Details

Depending on whether RcppArmadillo was compiled for the C++98 standard (currently the default) or for C++11 (optional), two different RNGs may be used. This function resets either. For C++98, the R programming language's RNG is used. For C++11, the RNG included in the `<random>` library is used only when `#define ARMA_USE_CXX11_RNG` is placed before `#include <RcppArmadillo.h>`. Otherwise, the R programming language's RNG will be used.

Value

The function is invoked for its side effect and has no return value.

Note

This has been found to not work as expected in **RStudio** as its code also uses the system RNG library. You may have to either not run within **RStudio** or change your code to use a different RNG such as the one from R.

See Also

The R documentation on its RNGs all of which are accessible via **Rcpp**.

armadillo_version	<i>Report the version of Armadillo</i>
-------------------	--

Description

Report the version of Armadillo

Usage

```
armadillo_version(single)
```

Arguments

`single` A logical vector indicating whether a single return values is requested, or a named vector with three elements `major`, `minor` and `patch`.

Details

The version is defined by Armadillo in the header `arma_version.hpp`.

Value

Depending on the value of `single`, either a single number describing the Armadillo version or a named vector with three elements `major`, `minor` and `patch`.

See Also

Armadillo header file `arma_version.hpp`.

fastLm	<i>Bare-bones linear model fitting function</i>
--------	---

Description

fastLm estimates the linear model using the solve function of Armadillo linear algebra library.

Usage

```
fastLmPure(X, y)

fastLm(X, ...)
## Default S3 method:
fastLm(X, y, ...)
## S3 method for class 'formula'
fastLm(formula, data = list(), ...)
```

Arguments

y	a vector containing the explained variable.
X	a model matrix.
formula	a symbolic description of the model to be fit.
data	an optional data frame containing the variables in the model.
...	not used

Details

Linear models should be estimated using the `lm` function. In some cases, `lm.fit` may be appropriate.

The `fastLmPure` function provides a reference use case of the Armadillo library via the wrapper functions in the **RcppArmadillo** package.

The `fastLm` function provides a more standard implementation of a linear model fit, offering both a default and a formula interface as well as `print`, `summary` and `predict` methods.

Lastly, one must be careful in timing comparisons of `lm` and friends versus this approach based on Armadillo. The reason that Armadillo can do something like `lm.fit` faster than the functions in the stats package is because Armadillo uses the Lapack version of the QR decomposition while the stats package uses a *modified* Linpack version. Hence Armadillo uses level-3 BLAS code whereas the stats package uses level-1 BLAS. However, Armadillo will either fail or, worse, produce completely incorrect answers on rank-deficient model matrices whereas the functions from the stats package will handle them properly due to the modified Linpack code.

An example of the type of situation requiring extra care in checking for rank deficiency is a two-way layout with missing cells (see the examples section). These cases require a special pivoting scheme of “pivot only on (apparent) rank deficiency” which is not part of conventional linear algebra software.

Value

fastLmPure returns a list with three components:

coefficients a vector of coefficients
 stderr a vector of the (estimated) standard errors of the coefficient estimates
 df.residual a scalar denoting the degrees of freedom in the model

fastLm returns a richer object which also includes the residuals, fitted values and call argument similar to the `lm` or `rlm` functions..

Author(s)

Armadillo is written by Conrad Sanderson. RcppArmadillo is written by Romain Francois, Dirk Eddelbuettel, Douglas Bates and Binxiang Ni.

References

Armadillo project: <http://arma.sourceforge.net/>

See Also

[lm](#), [lm.fit](#)

Examples

```
data(trees, package="datasets")

## bare-bones direct interface
flm <- fastLmPure( cbind(1, log(trees$Girth)), log(trees$Volume) )
print(flm)

## standard R interface for formula or data returning object of class fastLm
flmmod <- fastLm( log(Volume) ~ log(Girth), data=trees)
summary(flmmod)

## case where fastLm breaks down
dd <- data.frame(f1 = gl(4, 6, labels = LETTERS[1:4]),
                 f2 = gl(3, 2, labels = letters[1:3]))[-(7:8), ]
xtabs(~ f2 + f1, dd)    # one missing cell
mm <- model.matrix(~ f1 * f2, dd)
kappa(mm)             # large, indicating rank deficiency
set.seed(1)
dd$y <- mm %*% seq_len(ncol(mm)) + rnorm(nrow(mm), sd = 0.1)
summary(lm(y ~ f1 * f2, dd))    # detects rank deficiency
summary(fastLm(y ~ f1 * f2, dd)) # some huge coefficients
```

RcppArmadillo.package.skeleton

Create a skeleton for a new package that intends to use RcppArmadillo

Description

RcppArmadillo.package.skeleton automates the creation of a new source package that intends to use features of RcppArmadillo.

It is based on the [package.skeleton](#) function which it executes first.

Usage

```
RcppArmadillo.package.skeleton(name = "anRpackage", list = character(),
environment = .GlobalEnv, path = ".", force = FALSE,
code_files = character(), example_code = TRUE)
```

Arguments

name	See package.skeleton
list	See package.skeleton
environment	See package.skeleton
path	See package.skeleton
force	See package.skeleton
code_files	See package.skeleton
example_code	If TRUE, example c++ code using RcppArmadillo is added to the package

Details

In addition to [package.skeleton](#) :

The ‘DESCRIPTION’ file gains a Depends line requesting that the package depends on Rcpp and RcppArmadillo and a LinkingTo line so that the package finds Rcpp and RcppArmadillo header files.

The ‘NAMESPACE’, if any, gains a useDynLib directive.

The ‘src’ directory is created if it does not exist and a ‘Makevars’ file is added setting the environment variable ‘PKG_LIBS’ to accommodate the necessary flags to link with the Rcpp library.

If the example_code argument is set to TRUE, example files ‘rcpparma_hello_world.h’ and ‘rcpparma_hello_world.cpp’ are also created in the ‘src’. An R file ‘rcpparma_hello_world.R’ is expanded in the ‘R’ directory, the rcpparma_hello_world function defined in this file makes use of the C++ function ‘rcpparma_hello_world’ defined in the C++ file. These files are given as an example and should eventually be removed from the generated package.

Value

Nothing, used for its side effects

References

Read the *Writing R Extensions* manual for more details.

Once you have created a *source* package you need to install it: see the *R Installation and Administration* manual, [INSTALL](#) and [install.packages](#).

See Also

[package.skeleton](#)

Examples

```
## Not run:  
RcppArmadillo.package.skeleton( "foobar" )  
  
## End(Not run)
```

Index

- * **interface**
 - RcppArmadillo-package, [2](#)
- * **package**
 - RcppArmadillo-package, [2](#)
- * **programming**
 - RcppArmadillo-package, [2](#)
 - RcppArmadillo.package.skeleton, [8](#)
- * **regression**
 - fastLm, [6](#)

armadillo_set_seed, [3](#)
armadillo_set_seed_random, [4](#)
armadillo_version, [5](#)

fastLm, [6](#)
fastLmPure (fastLm), [6](#)

INSTALL, [9](#)
install.packages, [9](#)

lm, [6](#), [7](#)
lm.fit, [6](#), [7](#)

package.skeleton, [8](#), [9](#)

RcppArmadillo (RcppArmadillo-package), [2](#)
RcppArmadillo-package, [2](#)
RcppArmadillo.package.skeleton, [2](#), [8](#)
RcppArmadilloExample
 (RcppArmadillo-package), [2](#)
r1m, [7](#)