Package 'RcppEigen'

May 1, 2017

Type Package

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

Version 0.3.3.3.0

Date 2017-04-29

Author Douglas Bates, Dirk Eddelbuettel, Romain Francois, and Yixuan Qiu; the authors of Eigen for the included version of Eigen

Maintainer Dirk Eddelbuettel <edd@debian.org>

Description R and 'Eigen' integration using 'Rcpp'.
'Eigen' is a C++ template library for linear algebra: matrices, vectors, numerical solvers and related algorithms. It supports dense and sparse matrices on integer, floating point and complex numbers, decompositions of such matrices, and solutions of linear systems. Its performance on many algorithms is comparable with some of the best implementations based on 'Lapack' and level-3 'BLAS'. The 'RcppEigen' package includes the header files from the 'Eigen' C++ template library (currently version 3.3.3). Thus users do not need to install 'Eigen' itself in order to use 'RcppEigen'.
Since version 3.1.1, 'Eigen' is licensed under the Mozilla Public License (version 2); earlier version were licensed under the GNU LGPL version 3 or later. 'RcppEigen' (the 'Rcpp' bindings/bridge to 'Eigen') is licensed under the GNU GPL version 2 or later, as is the rest of 'Rcpp'.

License GPL (>= 2) | file LICENSE

Depends R (>= 2.15.1)

LazyLoad yes

LinkingTo Rcpp

Imports Matrix (>= 1.1-0), Rcpp (>= 0.11.0), stats, utils

Suggests inline, RUnit, pkgKitten

URL http://dirk.eddelbuettel.com/code/rcpp.eigen.html

BugReports https://github.com/RcppCore/RcppEigen/issues

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-05-01 15:10:35 UTC

R topics documented:

`RcppEigen-package` *Rcpp/Eigen bridge*

Description

The package eases the use of the Eigen C++ template library for linear algebra with Rcpp

Details

This package contains the header files for the Eigen C++ template library. The typical usage is to install this package and list it in the `LinkingTo:` line in the '`DESCRIPTION`' file of other packages. The C++ source code and the R source code in this package are for illustration only.

As described at the Eigen project's home page, `http://eigen.tuxfamily.org`, Eigen is a versatile, fast, reliable and elegant collection of C++ classes for linear algebra.

References

Douglas Bates and Dirk Eddelbuettel (2013). Fast and Elegant Numerical Linear Algebra Using the **RcppEigen** Package. *Journal of Statistical Software*, **52(5)**, 1-24. URL http://www.jstatsoft.org/v52/i05/.

`fastLm` *Bare-bones linear model fitting function*

Description

`fastLm` estimates the linear model using one of several methods implemented using the `Eigen` linear algebra library.

Usage

```
fastLmPure(X, y, method = 0L)
fastLm(X, ...)
## Default S3 method:
fastLm(X, y, method = 0L, ...)
## S3 method for class 'formula'
fastLm(formula, data = list(), method = 0L, ...)
```

Arguments

y	the response vector
X	a model matrix
formula	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given in the 'Details' section of the documentation for lm.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which lm is called.
method	an integer scalar with value 0 for the column-pivoted QR decomposition, 1 for the unpivoted QR decomposition, 2 for the LLT Cholesky, 3 for the LDLT Cholesky, 4 for the Jacobi singular value decomposition (SVD) and 5 for a method based on the eigenvalue-eigenvector decomposition of $\mathbf{X}'\mathbf{X}$. Default is zero.
...	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 Eigen C++ template library via the wrapper functions in the **RcppEigen** 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.

Internally the fastLm function, by default, uses a QR decomposition with column pivots, which is a rank-revealing decomposition, so that it can handle rank-deficient cases effectively. Other methods for determining least squares solutions are available according to the value of the method argument.

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 several components:

coefficients	a vector of coefficients
se	a vector of the standard errors of the coefficient estimates
rank	a scalar denoting the computed rank of the model matrix
df.residual	a scalar denoting the degrees of freedom in the model
residuals	the vector of residuals
s	a numeric scalar - the root mean square for residuals
fitted.values	the vector of fitted value

fastLm returns a richer object which also includes the call argument similar to the lm or rlm functions..

Author(s)

Eigen is described at http://eigen.tuxfamily.org. RcppEigen is written by Douglas Bates, Dirk Eddelbuettel and Romain Francois.

References

Douglas Bates and Dirk Eddelbuettel (2013). Fast and Elegant Numerical Linear Algebra Using the **RcppEigen** Package. *Journal of Statistical Software*, **52(5)**, 1-24. URL http://www.jstatsoft.org/v52/i05/.

See Also

lm, lm.fit

Examples

```
data(trees, package="datasets")
mm <- cbind(1, log(trees$Girth))   # model matrix
y  <- log(trees$Volume)            # response

## bare-bones direct interface
flm <- fastLmPure(mm, y)
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 non-rank-revealing methods break 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
try(summary(fastLm(y ~ f1 * f2, dd))) # also detects rank deficiency
```

RcppEigen.package.skeleton *Create a skeleton for a new package that intends to use RcppEigen*

Description

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

It is based on the package.skeleton function which it executes first.

Usage

```
RcppEigen.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 RcppEigen 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 RcppEigen and a LinkingTo line so that the package finds Rcpp and RcppEigen header files.

The 'NAMESPACE' gains a useDynLib directive.

The 'src' directory is created if it does not exists and a 'Makevars' file is added setting the environment variable 'PKG_LIBS' to accomodate the necessary flags to link with the Rcpp library.

If the example_code argument is set to TRUE, example files 'rcppeigen_hello_world.h' and 'rcppeigen_hello_world.cpp' are also created in the 'src'. An R file 'rcppeigen_hello_world.R' is expanded in the 'R' directory, the rcppeigen_hello_world function defined in this files makes use of the C++ function 'rcppeigen_hello_world' defined in the C++ file. These files are given as an example and should eventually by 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:
  RcppEigen.package.skeleton("foobar")

## End(Not run)
```

Index