Package 'ResidualMatrix'

April 18, 2024

Version 1.13.0

Date 2022-08-15

Title Creating a DelayedMatrix of Regression Residuals

Imports methods, Matrix, S4Vectors, DelayedArray

Suggests testthat, BiocStyle, knitr, rmarkdown, BiocSingular

biocViews Software, DataRepresentation, Regression, BatchEffect, ExperimentalDesign

Description Provides delayed computation of a matrix of residuals after fitting a linear model to each column of an input matrix. Also supports partial computation of residuals where selected factors are to be preserved in the output matrix. Implements a number of efficient methods for operating on the delayed matrix of residuals, most notably matrix multiplication and calculation of row/column sums or means.

License GPL-3

VignetteBuilder knitr

RoxygenNote 7.2.1

BugReports https://github.com/LTLA/ResidualMatrix/issues

URL https://github.com/LTLA/ResidualMatrix

git_url https://git.bioconductor.org/packages/ResidualMatrix

git_branch devel

git_last_commit c1ed3d7

git_last_commit_date 2023-10-24

Repository Bioconductor 3.19

Date/Publication 2024-04-17

Author Aaron Lun [aut, cre, cph]

Maintainer Aaron Lun <infinite.monkeys.with.keyboards@gmail.com>

Contents

ResidualMatrix-package 2
ResidualMatrix-class 2
ResidualMatrixSeed-class 4

Index **5**

`ResidualMatrix-package`
The **ResidualMatrix** *package*

Description

Originally implemented in the **BiocSingular** package, the `ResidualMatrix` class has been placed into its own package to enable greater re-use. This class provides delayed computation of residuals from a linear model fit, allowing us to represent large matrices of residuals without actually calculating them in memory. The idea is to allow us to easily regress out uninteresting factors for big datasets, much like a delayed, scalable version of **limma**'s venerable `removeBatchEffect` function.

`ResidualMatrix-class` *The ResidualMatrix class*

Description

The ResidualMatrix class supports delayed calculation of the residuals from a linear model fit. This serves as a light-weight representation of what would otherwise be a large dense matrix in memory. It also enables efficient matrix multiplication based on features of the the original matrix (e.g., sparsity).

Construction

`ResidualMatrix(x, design=NULL, keep=NULL)` returns a ResidualMatrix object, given:

- `x`, a matrix-like object. This can alternatively be a ResidualMatrixSeed, in which case `design` and `keep` are ignored.
- `design`, a numeric matrix containing the experimental design, to be used for linear model fitting on each *column* of `x`. This defaults to an intercept-only matrix.
- `keep`, an integer vector specifying the columns of `design` to *not* regress out. By default, all columns of `design` are regressed out.
- `restrict`, an integer or logical vector specifying the rows of `x` to use for model fitting. If `NULL`, all rows of `x` are used.

When `keep=NULL`, the ResidualMatrix contains values equivalent to `lm.fit(x=design, y=x)$residuals`.

Methods

In the following code chunks, x is a ResidualMatrix object:

- x[i, j, .., drop=FALSE] will return a ResidualMatrix object for the specified row and column subsets, or a numeric vector if either i or j are of length 1.
- t(x) will return a ResidualMatrix object with transposed contents.
- dimnames(x) <- value will return a ResidualMatrix object where the rows and columns are renamed by value, a list of two character vectors (or NULL).

colSums(x), colMeans(x), rowSums(x) and rowMeans(x) will return the relevant statistics for a ResidualMatrix x.

%*%, crossprod and tcrossprod can also be applied where one or both of the arguments are ResidualMatrix objects.

ResidualMatrix objects are derived from DelayedMatrix objects and support all of valid operations on the latter. All operations not listed here will use the underlying **DelayedArray** machinery. Unary or binary operations will generally create a new DelayedMatrix instance containing a ResidualMatrixSeed.

Author(s)

Aaron Lun

Examples

```
design <- model.matrix(~gl(5, 50))

library(Matrix)
y0 <- rsparsematrix(nrow(design), 200, 0.1)
y <- ResidualMatrix(y0, design)
y

# For comparison:
fit <- lm.fit(x=design, y=as.matrix(y0))
DelayedArray(fit$residuals)

# Keeping some of the factors:
y2 <- ResidualMatrix(y0, design, keep=1:2)
y2
DelayedArray(fit$residuals + design[,1:2] %*% fit$coefficients[1:2,])

# Matrix multiplication:
crossprod(y)
tcrossprod(y)
y %*% rnorm(200)
```

`ResidualMatrixSeed-class` *The ResidualMatrixSeed class*

Description

This is a seed class that powers the **DelayedArray** machinery underlying the ResidualMatrix.

Construction

`ResidualMatrixSeed(x, design=NULL, keep=NULL)` returns a ResidualMatrixSeed object, given:

- `x`, a matrix-like object. This can alternatively be a ResidualMatrixSeed, in which case `design` is ignored.
- `design`, a numeric matrix containing the experimental design, to be used for linear model fitting on each *column* of `x`. This defaults to an intercept-only matrix.
- `keep`, an integer vector specifying the columns of `design` to *not* regress out. By default, all columns of `design` are regressed out.
- `restrict`, an integer or logical vector specifying the rows of `x` to use for model fitting. If `NULL`, all rows of `x` are used.

Methods

ResidualMatrixSeed objects are implemented as DelayedMatrix backends. They support standard operations like `dim`, `dimnames` and `extract_array`.

Passing a ResidualMatrixSeed object to the `DelayedArray` or `ResidualMatrix` constructors will create a ResidualMatrix (which is what most users should be working with, anyway).

Author(s)

Aaron Lun

Examples

```
design <- model.matrix(~gl(5, 50))

library(Matrix)
y0 <- rsparsematrix(nrow(design), 200, 0.1)
s <- ResidualMatrixSeed(y0, design)
s

ResidualMatrix(s)

DelayedArray(s)
```

Index

[,ResidualMatrix,ANY,ANY,ANY-method
(ResidualMatrix-class), 2
%*%,ANY,ResidualMatrix-method
(ResidualMatrix-class), 2
%*%,ResidualMatrix,ANY-method
(ResidualMatrix-class), 2
%*%,ResidualMatrix,ResidualMatrix-method
(ResidualMatrix-class), 2

colMeans, *3*
colMeans,ResidualMatrix-method
(ResidualMatrix-class), 2
colSums, *3*
colSums,ResidualMatrix-method
(ResidualMatrix-class), 2
crossprod, *3*
crossprod,ANY,ResidualMatrix-method
(ResidualMatrix-class), 2
crossprod,ResidualMatrix,ANY-method
(ResidualMatrix-class), 2
crossprod,ResidualMatrix,missing-method
(ResidualMatrix-class), 2
crossprod,ResidualMatrix,ResidualMatrix-method
(ResidualMatrix-class), 2

DelayedArray, *4*
DelayedArray,ResidualMatrixSeed-method
(ResidualMatrixSeed-class), 4
DelayedMatrix, *3*, *4*
dim,ResidualMatrixSeed-method
(ResidualMatrixSeed-class), 4
dimnames,ResidualMatrixSeed-method
(ResidualMatrixSeed-class), 4
dimnames<-,ResidualMatrix,ANY-method
(ResidualMatrix-class), 2

extract_array,ResidualMatrixSeed-method
(ResidualMatrixSeed-class), 4

ResidualMatrix, *4*
ResidualMatrix *(*ResidualMatrix-class*)*, 2
ResidualMatrix-class, 2
ResidualMatrix-package, 2
ResidualMatrixSeed, *3*
ResidualMatrixSeed
(ResidualMatrixSeed-class), 4
ResidualMatrixSeed-class, 4
rowMeans, *3*
rowMeans,ResidualMatrix-method
(ResidualMatrix-class), 2
rowSums, *3*
rowSums,ResidualMatrix-method
(ResidualMatrix-class), 2

show,ResidualMatrixSeed-method
(ResidualMatrixSeed-class), 4

t,ResidualMatrix-method
(ResidualMatrix-class), 2
tcrossprod, *3*
tcrossprod,ANY,ResidualMatrix-method
(ResidualMatrix-class), 2
tcrossprod,ResidualMatrix,ANY-method
(ResidualMatrix-class), 2
tcrossprod,ResidualMatrix,missing-method
(ResidualMatrix-class), 2
tcrossprod,ResidualMatrix,ResidualMatrix-method
(ResidualMatrix-class), 2