Package: coop (via r-universe)

October 28, 2026

Type Package

Title Co-Operation: Fast Covariance, Correlation, and Cosine Similarity Operations

Version 0.6-3

Description Fast implementations of the co-operations: covariance, correlation, and cosine similarity. The implementations are fast and memory-efficient and their use is resolved automatically based on the input data, handled by R's S3 methods. Full descriptions of the algorithms and benchmarks are available in the package vignettes.

License BSD 2-clause License + file LICENSE

Depends R (>= 3.1.0)

Enhances slam (>= 0.1.32), Matrix

Suggests memuse

NeedsCompilation yes

ByteCompile yes

URL https://github.com/wrathematics/coop

BugReports https://github.com/wrathematics/coop/issues

Maintainer Drew Schmidt <wrathematics@gmail.com>

RoxygenNote 5.0.1

Author Drew Schmidt [aut, cre], Christian Heckendorf [ctb] (Caught some memory errors.)

Repository https://cran.r-universe.dev

Date/Publication 2021-09-19 12:40:02 UTC

RemoteUrl https://github.com/cran/coop

RemoteRef HEAD

RemoteSha aab4f5f491e0e18f34cf4c4b3c09937724858005

Contents

coop-package 2
cosine 3
covar 4
pcor 5
scaler 6
sparsity 7
weighted 8

Index **9**

`coop-package` *Cooperation: A Package of Co-Operations*

Description

Fast implementations of the co-operations: covariance, correlation, and cosine similarity. The implementations are fast and memory-efficient and their use is resolved automatically based on the input data, handled by R's S3 methods. Full descriptions of the algorithms and benchmarks are available in the package vignettes.

Covariance and correlation should largely need no introduction. Cosine similarity is commonly needed in, for example, natural language processing, where the cosine similarity coefficients of all columns of a term-document or document-term matrix is needed.

The `inplace` argument

When computing covariance and correlation with dense matrices, we must operate on the centered and/or scaled input data. When `inplace=FALSE`, a copy of the matrix is made. This allows for very wall-clock efficient processing at the cost of m*n additional double precision numbers allocated. On the other hand, if `inplace=TRUE`, then the wall-clock performance will drop considerably, but at the memory expense of only m+n additional doubles. For perspective, given a 30,000x30,000 matrix, a copy of the data requires an additional 6.7 GiB of data, while the inplace method requires only 469 KiB, a 15,000-fold difference.

Note that cosine is always computed in place.

The `t` functions

The package also includes "t" functions, like `tcosine()`. These behave analogously to `tcrossprod()` as `crossprod()` in base R. So of `cosine()` operates on the columns of the input matrix, then `tcosine()` operates on the rows. Another way to think of it is, `tcosine(x)` = `cosine(t(x))`.

Implementation Details

Multiple storage schemes for the input data are accepted. For dense matrices, an ordinary R matrix input is accepted. For sparse matrices, a matrix in COO format, namely `simple_triplet_matrix` from the slam package, is accepted.

The implementation for dense matrix inputs is dominated by a symmetric rank-k update via the BLAS subroutine `dsyrk`; see the package vignette for a discussion of the algorithm implementation and complexity.

The implementation for two dense vector inputs is dominated by the product `t(x) %*% y` performed by the BLAS subroutine `dgemm` and the normalizing products `t(y) %*% y`, each computed via the BLAS function `dsyrk`.

Author(s)

Drew Schmidt

`cosine` *Cosine Similarity*

Description

Compute the cosine similarity matrix efficiently. The function syntax and behavior is largely modeled after that of the `cosine()` function from the `lsa` package, although with a very different implementation.

Usage

```
cosine(x, y, use = "everything", inverse = FALSE)

tcosine(x, y, use = "everything", inverse = FALSE)
```

Arguments

`x`	A numeric dataframe/matrix or vector.
`y`	A vector (when `x` is a vector) or missing (blank) when `x` is a matrix.
`use`	The NA handler, as in R's `cov()` and `cor()` functions. Options are "everything", "all.obs", and "complete.obs".
`inverse`	Logical; should the inverse covariance matrix be returned?

Details

See `?coop-package` for implementation details.

Value

The $n \times n$ matrix of all pair-wise vector cosine similarities of the columns.

Author(s)

Drew Schmidt

See Also

sparsity

Examples

```
x <- matrix(rnorm(10*3), 10, 3)

coop::cosine(x)
coop::cosine(x[, 1], x[, 2])
```

---

covar *Covariance*

Description

An optimized, efficient implemntation for computing covariance.

Usage

```
covar(x, y, use = "everything", inplace = FALSE, inverse = FALSE)

tcovar(x, y, use = "everything", inplace = FALSE, inverse = FALSE)
```

Arguments

x	A numeric dataframe/matrix or vector.
y	A vector (when x is a vector) or missing (blank) when x is a matrix.
use	The NA handler, as in R's cov() and cor() functions. Options are "everything", "all.obs", and "complete.obs".
inplace	Logical; if TRUE then the method used is slower but uses less memory than if FALSE. See ?coop-package for details.
inverse	Logical; should the inverse covariance matrix be returned?

Details

See ?coop-package for implementation details.

Value

The covariance matrix.

Author(s)

Drew Schmidt

See Also

cosine

Examples

```
x <- matrix(rnorm(10*3), 10, 3)

coop::pcor(x)
coop::pcor(x[, 1], x[, 2])
```

---

pcor *Pearson Correlation*

Description

An optimized, efficient implemntation for computing the pearson correlation.

Usage

```
pcor(x, y, use = "everything", inplace = FALSE, inverse = FALSE)

tpcor(x, y, use = "everything", inplace = FALSE, inverse = FALSE)
```

Arguments

x	A numeric dataframe/matrix or vector.
y	A vector (when x is a vector) or missing (blank) when x is a matrix.
use	The NA handler, as in R's cov() and cor() functions. Options are "everything", "all.obs", and "complete.obs".
inplace	Logical; if TRUE then the method used is slower but uses less memory than if FALSE. See ?coop-package for details.
inverse	Logical; should the inverse covariance matrix be returned?

Details

See ?coop for implementation details.

Value

The pearson correlation matrix.

Author(s)

Drew Schmidt

See Also

cosine

Examples

```
x <- matrix(rnorm(10*3), 10, 3)

coop::pcor(x)
coop::pcor(x[, 1], x[, 2])
```

---

scaler *scaler*

Description

A function to center (subtract mean) and/or scale (divide by standard deviation) data column-wise in a computationally efficient way.

Usage

```
scaler(x, center = TRUE, scale = TRUE)
```

Arguments

x	The input matrix.
center, scale	Logical; determine if the data should be centered and/or scaled.

Details

Unlike its R counterpart `scale()`, the arguments `center` and `scale` can only be logical values (and not vectors).

Value

The centered/scaled data, with attributes as in R's `scale()`.

`sparsity` *Sparsity*

Description

Show the sparsity (as a count or proportion) of a matrix. For example, .99 sparsity means 99% of the values are zero. Similarly, a sparsity of 0 means the matrix is fully dense.

Usage

```
sparsity(x, proportion = TRUE)
```

Arguments

`x`	The matrix, stored as an ordinary R matrix or as a "simple triplet matrix" (from the slam package).
`proportion`	Logical; should a proportion or a count be returned?

Details

The implementation is very efficient for dense matrices. For sparse triplet matrices, the count is trivial.

Value

The sparsity of the input matrix, as a proportion or a count.

Author(s)

Drew Schmidt

Examples

```
## Completely sparse matrix
x <- matrix(0, 10, 10)
coop::sparsity(x)

## 15\% density / 85\% sparsity
x[sample(length(x), size=15)] <- 1
coop::sparsity(x)
```

`weighted` *Weighted Co-Operation*

Description

An optimized, efficient implemntation for computing weighted covariance, correlation, and cosine similarity. Similar to R's `cov.wt()`.

Arguments

`x`	A matrix or data.frame.
`wt`	A vector of weights or scalar weight.
`method`	Either "unbiased" or "ml". Unlike R, case is ignored.

Details

See `?coop-package` for implementation details.

Author(s)

Drew Schmidt

See Also

`cosine`, `pcor`, and `covar`

Examples

```
x <- matrix(rnorm(10*3), 10, 3)
cov.wt(x)
```

Index

∗ **package**
coop-package, 2

coop-package, 2
cosine, 3, *5*, *6*, *8*
covar, 4, *8*

pcor, 5, *8*

scaler, 6
sparsity, *4*, 7

tcosine *(*cosine*)*, 3
tcovar *(*covar*)*, 4
tpcor *(*pcor*)*, 5

weighted, 8