Package 'dissimilarities'

April 26, 2025

Type Package

Title Creating, Manipulating, and Subsetting 'dist' Objects

Version 0.1.0

Author Minh Long, Nguyen

Maintainer Minh Long, Nguyen <edelweiss611428@gmail.com>

Description This package provides efficient functions for creating, manipulating, and subsetting ``dist'' objects in R—commonly used in cluster analysis. It enables users to work seamlessly with ``dist'' objects while minimising unnecessary conversions and overhead.

Encoding UTF-8

LazyData true

Imports Rcpp,
microbenchmark,
proxy,
stats

LinkingTo Rcpp,
RcppArmadillo

RoxygenNote 7.3.2

Contents

Dist2Mat *Dist2Mat conversion*

Description

This function efficiently converts a "dist" object to a distance "matrix".

Usage

```
Dist2Mat(dist)
```

Arguments

dist	A "dist" object, which can be computed via the stats::dist() function.

Details

This function efficiently converts a "dist" object to a distance "matrix". It is faster than base::as.matrix() and proxy::as.matrix().

Value

A distance "matrix".

Author(s)

Minh Long Nguyen <edelweiss611428@gmail.com>

Examples

```
library("microbenchmark")
x = matrix(rnorm(200), nrow = 50)
dx = dist(x)
#Dist2Mat conversion
microbenchmark(base::as.matrix(dx),
               proxy::as.matrix(dx),
               Dist2Mat(dx))
#Check if equal
all.equal(as.vector(base::as.matrix(dx)), as.vector(Dist2Mat(dx)))
```

fastDist *"dist" object computation*

Description

This function efficiently computes a "dist" object from a numeric matrix.

Usage

```
fastDist(X, method = "euclidean", diag = F, upper = F, p = 2L)
```

Arguments

X	A numeric matrix.
method	A string specifying a distance method. Supported methods include "euclidean", "manhattan", "maximum", "minkowski", "cosine", and "canberra".
diag	A boolean value, indicating whether to display the diagonal entries.
upper	A boolean value, indicating whether to display the upper triangular entries.
p	A positive integer, required for computing Minkowski distance; by default p = 2 (i.e., Euclidean).

Details

This function computes a distance matrix of class "dist", consisting of pair-wise distances between rows in the input data matrix. This object is internally an 1d array.

Value

A distance matrix of class "dist".

Author(s)

Minh Long Nguyen <edelweiss611428@gmail.com>

Examples

```
library("microbenchmark")
x = matrix(rnorm(200), nrow = 50)
microbenchmark(stats::dist(x, "minkowski", p = 5),
               fastDist(x, "minkowski", p = 5))
all.equal(as.vector(stats::dist(x, "minkowski", p = 5)), as.vector(fastDist(x, "minkowski", p = 5)))
```

---

fastDistAB *Pairwise distances between rows of two matrices*

Description

This function efficiently computes pairwise distances between rows of two numeric matrices.

Usage

```
fastDistAB(A, B, method, p = 2)
```

Arguments

A	A numeric matrix.
B	A numeric matrix.
method	A string specifying a distance method. Supported methods include "euclidean", "manhattan", "maximum", "minkowski", "cosine", and "canberra".
p	A positive integer, required for computing Minkowski distance; by default p = 2 (i.e., Euclidean).

Details

This function computes a distance "matrix" storing pairwise distances between rows of two numeric matrices A and B.

Value

A matrix storing pairwise distances between rows of A and B.

Author(s)

Minh Long Nguyen <edelweiss611428@gmail.com>

Examples

```
library("microbenchmark")
X = matrix(rnorm(200), nrow = 50)
A = X[1:25,]
B = X[26:50,]
microbenchmark(proxy::dist(A,B, "minkowski", p = 5),
               fastDistAB(A,B, "minkowski", p = 5L))
all.equal(as.vector(proxy::dist(A,B, "minkowski", p = 5)), as.vector(fastDistAB(A,B, "minkowski", p = 5L)))
```

---

subCols *Subsetting by column from a "dist" object*

Description

This function efficiently subsets a "dist" object by column to a smaller distance "matrix".

Usage

```
subCols(dist, idx)
```

Arguments

dist	A "dist" object, which can be computed via the stats::dist() function, representing the full pairwise distance matrix between observations.
idx	An integer vector, specifying the column indices of the subsetted matrix.

Details

This function efficiently extracts columns from a "dist" object without involving unnecessary conversion (to a full data matrix), thereby improving efficiency and memory usage.

Value

A matrix storing pairwise distances between pairs in idx1 x idx2.

Author(s)

Minh Long Nguyen <edelweiss611428@gmail.com>

Examples

```
library("microbenchmark")
x = matrix(rnorm(200), nrow = 50)
dx = dist(x)
#Randomly subsetting a 50x10 matrix
idx = sample(1:50, 10)
microbenchmark(base::as.matrix(dx)[1:50,idx],
               proxy::as.matrix(dx)[1:50,idx],
               subCols(dx, idx))
#Check if equal
all.equal(as.vector(base::as.matrix(dx)[1:50,idx]), as.vector(subCols(dx, idx)))
```

`subDist2Dist` *Dist2Dist subsetting*

Description

This function efficiently subsets a "dist" object to a smaller "dist" object.

Usage

```
subDist2Dist(dist, idx, diag = F, upper = F)
```

Arguments

`dist`	A "dist" object, which can be computed via the stats::dist() function, representing the full pairwise distance matrix between observations.
`idx`	An integer vector, specifying the indices of the observations to retain.
`diag`	A boolean value, indicating whether to display the diagonal entries.
`upper`	A boolean value, indicating whether to display the upper triangular entries.

Details

This function efficiently subsets a "dist" object to a smaller "dist" object, which is useful for constructing subsetting-based clustering algorithms. It does not involve conversion of the original "dist" object to a numeric matrix, thereby reducing memory usage and improving computational efficiency.

Value

A subsetted "dist" object.

Author(s)

Minh Long Nguyen <edelweiss611428@gmail.com>

Examples

```
library("microbenchmark")
x = matrix(rnorm(200), nrow = 50)
dx = dist(x)
#Subsetting the first 10 units
microbenchmark(as.dist(base::as.matrix(dx)[1:10,1:10]),
               as.dist(proxy::as.matrix(dx)[1:10,1:10]),
               subDist2Dist(dx, 1:10))
#Check if equal
all.equal(as.vector(as.dist(base::as.matrix(dx)[1:10,1:10])), as.vector(subDist2Dist(dx, 1:10)))
```

`subDist2Mat` *Dist2Mat subsetting*

Description

This function efficiently subsets a "dist" object to a smaller distance "matrix".

Usage

```
subDist2Mat(dist, idx1, idx2)
```

Arguments

`dist`	A "dist" object, which can be computed via the stats::dist() function, representing the full pairwise distance matrix between observations.
`idx1`	An integer vector, specifying the row indices of the subsetted matrix.
`idx2`	An integer vector, specifying the column indices of the subsetted matrix.

Details

This function efficiently subsets a "dist" object to a smaller "matrix" based on row and column indices of the distance matrix. It does not involve conversion of the original "dist" object to a numeric matrix, thereby reducing memory usage and improving computational efficiency.

Value

A matrix storing pairwise distances between pairs in idx1 x idx2.

Author(s)

Minh Long Nguyen <edelweiss611428@gmail.com>

Examples

```
library("microbenchmark")
x = matrix(rnorm(200), nrow = 50)
dx = dist(x)
#Randomly subsetting a 10x10 matrix
idx1 = sample(1:50, 10)
idx2 = sample(1:50, 10)
```

```
microbenchmark(base::as.matrix(dx)[idx1,idx2],
               proxy::as.matrix(dx)[idx1,idx2],
               subDist2Mat(dx, idx1, idx2))
#Check if equal
all.equal(as.vector(base::as.matrix(dx)[idx1,idx2]), as.vector(subDist2Mat(dx, idx1, idx2)))
```

Index