Package ‘disto’

August 2, 2018

Type Package

Title Unified Interface to Distance, Dissimilarity, Similarity Matrices

Version 0.2.0

Description Provides a high level API to interface over sources storing distance, dissimilarity, similarity matrices with matrix style extraction, replacement and other utilities. Currently, in-memory dist object backend is supported.

URL https://github.com/talegari/disto

BugReports https://github.com/talegari/disto/issues

Imports proxy (>= 0.4.19), dplyr (>= 0.7.4), assertthat (>= 0.2.0), fastmatch (>= 1.1.0), tidyr (>= 0.8.0), factoextra (>= 1.0.5), ggplot2 (>= 2.2.1), broom (>= 0.4.4), fastcluster (>= 1.1.25), pbapply (>= 1.3.4),

Depends R (>= 3.4.0)

License GPL-3

Encoding UTF-8

RoxygenNote 6.0.1

Suggests knitr (>= 1.15.1), rmarkdown (>= 1.4),

VignetteBuilder knitr

NeedsCompilation no

Author KS Srikanth [aut, cre]

Maintainer KS Srikanth <sri.teach@gmail.com>

Repository CRAN

Date/Publication 2018-08-02 12:50:02 UTC

R topics documented:

as.data.frame.disto .......................................................... 2
dapply ................................................................. 3
as.data.frame.disto

Convert a disto object to dataframe

Description

Convert the underlying data of a disto object to a dataframe in long format (3 columns: item1, item2, distance). This might be a costly operation and should be used with caution.

Usage

## S3 method for class 'disto'
as.data.frame(xL ...)

Arguments

x object of class disto
...
arguments for tidy

Value

a dataframe in long format

Examples

temp <- stats::dist(iris[,1:4])
dio <- disto(objectname = "temp")
dio
head(as.data.frame(dio))
**dapply**

**Matrix like apply function for disto object**

**Description**

Apply function for data underlying disto object

**Usage**

dapply(x, margin = 1, fun, subset, nproc = 1)

**Arguments**

- **x**: disto object
- **margin**: (one among 1 or 2) dimension to apply function along
- **fun**: Function to apply over the margin
- **subset**: (integer vector) Row/Column numbers along the margin
- **nproc**: Number of parallel processes (unix only)

**Value**

Simplified output of 'sapply' like function
temp <- dist(iris[,1:4])
dio <- disto(objectname = "temp")

# function to pick indexes of 5 nearest neighbors
# an efficient alternative with Rcpp is required
udf <- function(x) dim(x) <- NULL order(x)[1:6]

hi <- dapply(dio, 1, udf)[-1,]
dim(hi)

**disto**

**Constructor for class 'disto'**

**Description**

Create mapping to data sources storing distances(symmetric), dissimilarities(non-symmetric), similarities and so on

Provides a high level API to interface over backends storing distance, dissimilarity, similarity matrices with matrix style extraction, replacement and other utilities. Currently, in-memory dist object backend is supported.

**Usage**

disto(..., backend = "dist")
Arguments

Arguments for a backend. See details

backend (string) Specify a backend. Currently supported: 'dist'

Details

This is a wrapper to create a 'disto' handle over different backends storing distances, dissimilarities, similarities etc with minimal data overhead like a database connection. The following named arguments are required to set-up the backend:

- dist:
  - objectname: Object of the class 'dist' or the name of the object as a 'string'.
  - env: Environment where the object exists. When this is missing, its assumed to be parent environment.

Value

Object of class 'disto' which is a thin wrapper on a list

Author(s)

Srikanth KS

See Also

Useful links:

- [https://github.com/talegari/disto](https://github.com/talegari/disto)
- Report bugs at [https://github.com/talegari/disto/issues](https://github.com/talegari/disto/issues)

Examples

```R
temp <- stats::dist(iris[,1:4])
dio <- disto(objectname = "temp")
dio
unclass(dio)
```

disto_dist

Constructor of disto with dist backend

Description

Constructor of disto with dist backend

Usage

disto_dist(arguments)
**dist_extract**

**Arguments**

- arguments to construct disto object

**Details**

to be used by disto constructor function

**Value**

returns a list

---

**Description**

Matrix style extraction supports 'inner' and 'outer'(default) products

**Usage**

```
dist_extract(object, i, j, k, product = "outer")
```

**Arguments**

- object dist object
- i (integer vector) row positions
- j (integer vector) column positions
- k (integer vector) positions
- product (string) One among: 'inner', 'outer'(default)

**Details**

In k-mode, both i and j should be missing and k should not be missing. In ij-mode, k should be missing and both i and j are optional. If i or j are missing, they are interpreted as all values of i or j (similar to matrix or dataframe subsetting). If i and j are of unequal length, the smaller one is recycled.

**Value**

A matrix or vector of distances when product is 'outer' and 'inner' respectively
Examples

# examples for dist_extract

# create a dist object
temp <- dist(iris[,1:4])
attr(temp, "Labels") <- outer(letters, letters, paste0)[1:150]
head(temp)
max(temp)
as.matrix(temp)[1:5, 1:5]

dist_extract(temp, 1, 1)
dist_extract(temp, 1, 2)
dist_extract(temp, 2, 1)
dist_extract(temp, "aa", "ba")

dist_extract(temp, 1:10, 11:20)
dim(dist_extract(temp, 1:10, ))
dim(dist_extract(temp, , 1:10))
dist_extract(temp, 1:10, 11:20, product = "inner")
length(dist_extract(temp, 1:10, , product = "inner"))
length(dist_extract(temp, , 1:10, product = "inner"))

dist_extract(temp, c("aa", "ba", "ca"), c("ca", "da", "fa"))
dist_extract(temp, c("aa", "ba", "ca"), c("ca", "da", "fa"), product = "inner")

dist_extract(temp, k = 1:3) # product is always inner when k is specified

---

dist_ij_k  Vectorized version of dist_ij_k_

Description

Convert ij indexes to k indexes for a dist object

Usage

dist_ij_k(i, j, size)

Arguments

i  row indexes
j  column indexes
size  value of size attribute of the dist object

Value

k indexes
**dist_ij_k_**  
*Convert ij index to k index*

**Description**  
Convert ij index to k index for a dist object

**Usage**  
\[
dist_ij_k_(i, j, size)
\]

**Arguments**  
- **i**: row index
- **j**: column index
- **size**: value of size attribute of the dist object

**Value**  
- k index

---

**dist_k_ij**  
*Vectorized version of dist_k_ij_

**Description**  
Convert kth indexes to ij indexes of a dist object

**Usage**  
\[
dist_k_ij(k, size)
\]

**Arguments**  
- **k**: kth indexes
- **size**: value of size attribute of the dist object

**Value**  
i j indexes as 2^*n* matrix where n is length of k vector
**dist_k_ij_**  
*Convert kth index to ij index*

**Description**
Convert kth index to ij index of a dist object

**Usage**
```
dist_k_ij_(k, size)
```

**Arguments**
- `k`: kth index
- `size`: value of size attribute of the dist object

**Value**
ij index as a length two integer vector

---

**dist_replace**  
*Replacement values in dist*

**Description**
Replacement values of a dist object with either ij or position indexing

**Usage**
```
dist_replace(object, i, j, value, k)
```

**Arguments**
- `object`: dist object
- `i`: (integer vector) row positions
- `j`: (integer vector) column positions
- `value`: (integer/numeric vector) Values to replace
- `k`: (integer vector) positions
dist_subset

Details

There are two modes to specify the positions:

- ij-mode where i and j are specified and k is missing. If i or j are missing, they are interpreted as all values of i or j (similar to matrix or dataframe subsetting). Lengths of i, j are required to be same. If 'value' is singleton, then it is extended to the length of i or j. Else, 'value' should have same length as i or j.

- k-mode where k is present and both i and k are missing. k is the positions in the dist object. If 'value' is singleton, then it is extended to the length of k. Else, 'value' should have same length as k.

Value

dist object

Examples

# create a dist object
d <- dist(iris[,1:4])
attr(d, "Labels") <- outer(letters, letters, paste0)[1:150]
head(d)
max(d)
as.matrix(d)[1:5, 1:5]

# replacement in ij-mode

d <- dist_replace(d, 1, 2, 100)
dist_extract(d, 1, 2, product = "inner")
d <- dist_replace(d, "ca", "ba", 102)
dist_extract(d, "ca", "ba", product = "inner")

d <- dist_replace(d, 1:5, 6:10, 11:15)
dist_extract(d, 1:5, 6:10, product = "inner")
d <- dist_replace(d, c("ca", "da"), c("aa", "ba"), 102)
dist_extract(d, c("ca", "da"), c("aa", "ba"), product = "inner")

# replacement in k-mode

d <- dist_replace(d, k = 2, value = 101)
dist_extract(d, k = 2)
dist_extract(d, 3, 1, product = "inner")  # extracting k=2 in ij-mode

Description

Compute subset faster than regular ‘[‘ on a dist object. This is from proxy package (not exported by proxy).
Usage

\[
\text{dist_subset}(x, \text{subset}, \ldots)
\]

Arguments

\begin{itemize}
\item \texttt{x} \hspace{1cm} \text{dist object}
\item \texttt{subset} \hspace{1cm} \text{index of the subset. This has to be unique.}
\item \texttt{\ldots} \hspace{1cm} \text{additional arguments}
\end{itemize}

Value

returns a dist subset

---

\begin{itemize}
\item \texttt{names.disto} \hspace{1cm} \textbf{Get names/labels}
\end{itemize}

Description

Get names/labels of the underlying distance storing backend

Usage

\[
\texttt{## S3 method for class 'disto'}
\]
\[
\texttt{\textbf{names}(x)}
\]

Arguments

\begin{itemize}
\item \texttt{x} \hspace{1cm} \text{disto object}
\end{itemize}

Value

A character vector

Examples

\[
\text{temp} \leftarrow \text{stats::dist}(\text{iris[,1:4]})
\]
\[
\text{dio} \leftarrow \text{disto}(\text{objectname = "temp"})
\]
\[
\text{dio}
\]
\[
\text{names(dio)} \leftarrow \text{paste0("a", 1:150)}
\]
plot.disto

Plot a disto object

Description

Various plotting options for subsets of disto objects

Usage

## S3 method for class 'disto'
plot(x, ...)

Arguments

x

object of class disto

...

Additional arguments. See details.

Details

Among the additional arguments,

- 'type' is mandatory. Currently, these options are supported: heatmap, dendrogram.
- sampleSize: A random sample of indexes is drawn from the distance object underlying the disto mapping. Default value of sampleSize is set to 100.
- seed seed for random sample. Default is 100.

Value

ggplot object

Examples

temp <- stats::dist(iris[,1:4])
dio <- disto(objectname = "temp")
plot(dio, type = "heatmap")
plot(dio, type = "dendrogram")
print.disto

*Print method for dist class*

**Description**
Print method for dist class

**Usage**
```
## S3 method for class 'disto'
print(x, ...)
```

**Arguments**
- `x` object of class disto
- `...` currently not in use

**Value**
invisible NULL. Function writes backend type and size to terminal as a message.

**Examples**
```
temp <- stats::dist(iris[,1:4])
dio <- disto(objectname = "temp")
print(dio)
```

---

**size**

*Obtain size of the disto object*

**Description**
Obtain size of the disto object

**Usage**
```
size(disto, ...)
```

**Arguments**
- `disto` object of class disto
- `...` currently not in use

**Value**
Integer vector of length 1
**Examples**

```r
temp <- stats::dist(iris[,1:4])
dio <- disto(objectname = "temp")
size(dio)
```

**Description**

Summary method for dist class

**Usage**

```r
## S3 method for class 'disto'
summary(object, ...)
```

**Arguments**

- `object`: object of class disto
- `...`: currently not in use

**Value**

invisibly returns the tidy output of summary as a dataframe.

**Examples**

```r
temp <- stats::dist(iris[,1:4])
dio <- disto(objectname = "temp")
dio
summary(dio)
```

**'names<-.disto'**

Set names/labels

**Description**

Set names/labels of the underlying distance storing backend

**Usage**

```r
## S3 replacement method for class 'disto'
names(x) <- value
```
### Arguments

- **x**: disto object
- **value**: A character vector

### Value

invisible disto object

### Examples

```r
temp <- stats::dist(iris[,1:4])
dio <- disto(objectname = "temp")
dio
names(dio) <- paste0("a", 1:150)
```

---

### ` [.disto`

*Extract from a disto object in matrix style extraction*

**Description**

Extract a disto object in matrix style extraction and via direct indexing. `product` specification allows both outer (matrix output, default option) and inner (vector) product type extraction. For dist backend see: dist_extract.

**Usage**

```r
## S3 method for class 'disto'
x[i, j, k, product = "outer"]
```

**Arguments**

- **x**: object of class ’disto’
- **i**: (integer vector) row indexes
- **j**: (integer vector) column indexes
- **k**: (integer vector) direct indexes
- **product**: (string) One among: "inner", "outer"

**Value**

When product is ’outer’, returns a matrix. Else, a vector.
Examples

temp <- stats::dist(iris[,1:4])
dio <- disto(objectname = "temp")
dio
names(dio) <- paste0("a", 1:150)

dio[1,2]
dio[2,1]
dio[c("a1", "a10"), c("a5", "a72")]
dio[c("a1", "a10"), c("a5", "a72"), product = "inner"]
dio[k = c(1,3,5)]

'[<-.disto'  In-place replacement of values

Description

For dist backend see: dist_replace.

Usage

## S3 replacement method for class 'disto'
x[i, j, k] <- value

Arguments

x  object of class 'disto'
i  (integer vector) row index
j  (integer vector) column index
k  (integer vector) direct index
value  (integer/numeric vector) Values to replace

Value

Invisible disto object. Note that this function is called for its side effect.

Examples

temp <- stats::dist(iris[,1:4])
dio <- disto(objectname = "temp")
names(dio) <- paste0("a", 1:150)
dio

dio[1,2] <- 10
dio[1:10, 2:11] <- 100
Description

Extract a single value from disto object in matrix style extraction and via direct indexing. This does not support using names. This is faster than link{extract}. For dist backend see: dist_extract.

Usage

## S3 method for class 'disto'

x[[i, j, k]]

Arguments

x object of class 'disto'
i (integer vector) row index
j (integer vector) column index
k (integer vector) direct index

Value

(A real number) Distance value

Examples

temp <- stats::dist(iris[,1:4])
dio <- disto(objectname = "temp")
dio
dio[[1, 2]]
dio[[2, 1]]
dio[[k = 3]]
Index

[.disto('[.disto'), 14
[<-.disto('[<-.disto'), 15
[[.disto('[[.disto'), 16
' [.disto', 14
' [<-.disto', 15
' [[.disto', 16
' names<-.disto''', 13

as.data.frame.disto, 2
dapply, 3
dist_extract, 5, 14, 16
dist_ij_k, 6
dist_ij_k_, 7
dist_k_ij, 7
dist_k_ij_, 8
dist_replace, 8, 15
dist_subset, 9
disto, 3
disto-package (disto), 3
disto_dist, 4

names.disto, 10
names<-.disto('names<-.disto'''), 13

plot.disto, 11
print.disto, 12

size, 12
summary.disto, 13

tidy, 2