Package 'tscR'

April 12, 2022

Type Package

Title A time series clustering package combining slope and Frechet distances

Version 1.6.1

Description Clustering for time series data using slope distance and/or shape distance.

License Artistic-2.0

Depends R (>= 4.1), dplyr

Imports gridExtra, methods, dtw, class, kmlShape, graphics, cluster, RColorBrewer, grDevices, knitr, rmarkdown, prettydoc, grid, ggplot2, latex2exp, stats, SummarizedExperiment, GenomicRanges, IRanges, S4Vectors

Encoding UTF-8

RoxygenNote 7.0.2

VignetteBuilder knitr

biocViews GeneExpression, Clustering, DNAMethylation, Microarray

Suggests testthat

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

git_branch RELEASE_3_14

git_last_commit f2624fa

git_last_commit_date 2022-01-27

Date/Publication 2022-04-12

Author Fernando Pérez-Sanz [aut, cre], Miriam Riquelme-Pérez [aut]

Maintainer Fernando Pérez-Sanz <fernando.perez8@um.es>

R topics documented:

combineCluster 2
frechetDist 3
frechetDistC 4

getClusters 5
importFromSE 6
imputeSenator-class 7
imputeSenators 8
imputeSenatorToData 9
plotCluster 10
plotClusterSenator 11
slopeDist 12
tscR 13

Index **15**

`combineCluster` *Combine clusters function*

Description

Combine clusters from two different clustering

Usage

```
combineCluster(x, y)
```

Arguments

`x`	Object of class pam. Output of getClusters.
`y`	Object of class pam. Output of getClusters.

Details

This function combines the clusters obtained in getClusters with 'slope' distance and the ones obtained with Frechet distance, resulting in a new clustering combining both distances.

Value

Object of class 'pam'. See `pam.object` for details

Author(s)

Fernando Pérez-Sanz (<fernando.perez8@um.es>)

Miriam Riquelme-Pérez (<miriam.riquelmep@gmail.com>)

See Also

`getClusters`,`plotCluster`

Examples

```
data(tscR)
data <- tscR
time <- c(1,2,3)
dist_slope <- slopeDist(data, time)
dist_frechet <- frechetDistC(data, time)
slope.cluster <- getClusters(dist_slope, 3)
frechet.cluster <- getClusters(dist_frechet, 4)
comb.cluster <- combineCluster(slope.cluster, frechet.cluster)
```

---

frechetDist *Pairwise Frechet distance*

Description

Compute pair-wise Frechet distance in a matrix of trajectories

Usage

```
frechetDist(x, time, ...)
```

Arguments

x	Numeric matrix or data.frame with trajectory values. Rows are trajectories, columns are time or similar. SummarizedExperiment object can be provided for compatibility with bioconductor container (for more information see vignette).
time	Numeric vector with time data (time intervals), with equal length to columns number in x.
...	Other arguments to pass to importFromSE if _x_ is SummarizedExperiment-class.

Details

This function is a wrapper of the distFrechet code from kmlShape package for use with a matrix or a data.frame so that the user can compute pairwise distances between all trajectories.

Value

A dist class object of size NxN, where N is rows number in the input data

Author(s)

Fernando Pérez-Sanz (<fernando.perez8@um.es>)

Miriam Riquelme-Pérez (<miriam.riquelmep@gmail.com>)

See Also

distFrechet (package kmlShape), slopeDist, frechetDistC (C and faster versión than frechetDist), importFro

Examples

```
data(tscR)
data <- tscR
time <- c(1,2,3)
dist_tscR <- frechetDist(data, time)
```

---

frechetDistC *Pairwise Frechet distance (C version)*

Description

Compute pairwise Frechet distance in a matrix of trajectories. This function is a C implementation and a lot faster version than frechetDist

Usage

```
frechetDistC(x, time, ...)
```

Arguments

x	Numeric matrix or data.frame with trajectory values. Rows are trajectories, columns are time or similar. SummarizedExperiment object can be provided for compatibility with bioconductor container (for more information see vignette).
time	Numeric vector with time data (time intervals), with equal length to columns number in x.
...	Other arguments to pass to importFromSE if _x_ is SummarizedExperiment-class.

Details

This function is a C adaptation of the distFrechet code from kmlShape package for use with a matrix or a dataframe so that the user can compute pairwise distances between all trajectories.

It is highly recommended to use this function over frechetDist because it is a lot faster.

Value

A dist class object of size NxN, where N is rows number in the input data

Author(s)

Fernando Pérez-Sanz (<fernando.perez8@um.es>)

Miriam Riquelme-Pérez (<miriam.riquelmep@gmail.com>)

See Also

distFrechet (package kmlShape),slopeDist,frechetDist (R and slower versión than frechetDistC),importFro

Examples

```
data(tscR)
data <- tscR
time <- c(1,2,3)
dist_tscR <- frechetDistC(data, time)
```

---

getClusters *Generic clustering function*

Description

Compute clustering with pam function and a distance class object.

Usage

```
getClusters(x, k)
```

Arguments

x	Numeric distance object obtained with dimension n x n.
k	Numeric. Number of clusters

Details

This function is a wrapper of pam (cluster). x must be a dist object obtained from frechetdist, slopedist or any other distance metric on condition that it be an object of the dist class and has dimensions nxn, where n is equal to the number of trajectories.

Value

Object of class 'pam'. See pam.object for details

Author(s)

Fernando Pérez-Sanz (<fernando.perez8@um.es>)

Miriam Riquelme-Pérez (<miriam.riquelmep@gmail.com>)

See Also

pam,plotCluster.

Examples

```
data(tscR)
data <- tscR
time <- c(1,2,3)
dist_tscR <- slopeDist(data, time)
res.cluster <- getClusters(dist_tscR, 3)
```

`importFromSE` *Convenient import from a SummarizedExperiment object*

Description

Import a summarizedExperiment object containing expression data at different times.

Usage

```
importFromSE(se, sample, SE_byTime = FALSE)
```

Arguments

`se`	SummarizedExperiment. Where assays slot can be organized either by samples or by times.
`sample`	Numeric or character. Sample identifier. See details and vignette
`SE_byTime`	Logical. Default FALSE. Indicates whether the data is organized by sample or by time.

Details

This function enables the integration of an object of the summarizedExpmeriment class and is integrated into all other functions, so there is no need to run it in an isolated way. There are two possible organization options in the slot assays. A) each row is a gene each column, is a sample and each matrix is a time. In this case SE_byTime=FALSE. B) each row is a gene , each column is a time and each matrix is a sample. In this case SE_byTime= TRUE. This concept is illustrated in package vignette.

Value

Sample data frame where rows are genes and columns are times.,

Author(s)

Fernando Pérez-Sanz (<fernando.perez8@um.es>)

Miriam Riquelme-Pérez (<miriam.riquelmep@gmail.com>)

Examples

```
# Each matrix one time / each column one sample (SE_byTime=FALSE)
# Code to create dummy data

nrows <- 200
ncols <- 6
time1 <- matrix(runif(nrows * ncols, 1, 1e4), nrows)
time2 <- matrix(runif(nrows * ncols, 1, 1e4), nrows)
rowRanges <- GenomicRanges::GRanges(rep(c("chr1", "chr2"), c(50, 150)),
                    IRanges::IRanges(floor(runif(200, 1e5, 1e6)), width=100),
                                        strand=sample(c("+", "-"), 200, TRUE),
                                        feature_id=sprintf("ID%03d", 1:200))
colData <- S4Vectors::DataFrame(Treatment=rep(c("ChIP", "Input"), 3),
                     Samples = LETTERS[1:6],
                     row.names=LETTERS[1:6])
se <- SummarizedExperiment::SummarizedExperiment(assays=list(time1=time1, time2=time2),
                     rowRanges=rowRanges, colData=colData)

# Get sample "A" with all times

importFromSE(se, sample="A", SE_byTime = FALSE)

# or sample = 1 because is first columns in each matrix (each time)

# Each matrix one sample / each column one time (SE_byTime=TRUE)
# Code to create dummy data

nrows <- 200
ncols <- 6
sampleA <- matrix(runif(nrows * ncols, 1, 1e4), nrows)
sampleB <- matrix(runif(nrows * ncols, 1, 1e4), nrows)
rowRanges <- GenomicRanges::GRanges(rep(c("chr1", "chr2"), c(50, 150)),
                    IRanges::IRanges(floor(runif(200, 1e5, 1e6)), width=100),
                                        strand=sample(c("+", "-"), 200, TRUE),
                                        feature_id=sprintf("ID%03d", 1:200))
 colData <- S4Vectors::DataFrame(Time=paste("time",seq(1:6), sep=""),
                     sampleA = rep("A",6),
                     sampleB = rep("B", 6),
                     row.names = paste("time",seq(1:6), sep=""))
se <- SummarizedExperiment::SummarizedExperiment(assays=list(sampleA=sampleA, sampleB=sampleB),
                     rowRanges=rowRanges, colData=colData)
# Get sample "sampleA" with all times

importFromSE(se, sample=1, SE_byTime = TRUE)

# or sample = 1 because is the first matrix in assays structure.
```

`imputeSenator-class` *Class 'inputeSenator' This class represents the result imputeSenators function*

Description

Class 'inputeSenator' This class represents the result imputeSenators function

Slots

`data` Dataframe with original data

`senators` Senators data value

`endCluster` Final cluster assignment

Author(s)

Fernando Pérez Sanz (<fernando.perez8@um.es>)

Miriam Riquelme Pérez (<miriam.riquelmep@gmail.com>)

See Also

`imputeSenators`

`imputeSenators` *Preclustering function for large data*

Description

Compute clustering with clara function to obtain a number of 'senators'

Usage

```
imputeSenators(x, k = 100, ...)
```

Arguments

`x`	Numeric matrix or data.frame with trajectory values. Rows are trajectories, columns are time or similar. SummarizedExperiment object can be provided for compatibility with bioconductor container (for more information see vignette).
`k`	Numeric. Number of senators
`...`	Other arguments to pass to importFromSE if _x_ is SummarizedExperiment-class.

Details

Calculates a series of senators representing a large set of trajectories that would otherwise be computationally very expensive. For it, by means of the clara function of the cluster package a clustering is made obtaining the centroids as senators. These centroids can then be clustered based on the slope distance or Frechet or both. Finally, the data set will be assigned to the same cluster your senator is assigned to.

Value

List with three slots:

data Dataframe with original data.

senatorData Matrix with senator trajectories.

senatorCluster Vector with senator clusters.

Author(s)

Fernando Pérez-Sanz (<fernando.perez8@um.es>)

Miriam Riquelme-Pérez (<miriam.riquelmep@gmail.com>)

See Also

plotClusterSenator, imputeSenatorToData, importFromSE.

Examples

```
data( tscR )
data <- tscR
time <- c( 1, 2, 3 )
senators <- imputeSenators( data, k = 100 )
senatorDist <- slopeDist( senators$senatorData, time )
sClust <- getClusters( senatorDist, k = 5 )
plotCluster( senators$senatorData, sClust, 2 )
```

---

imputeSenatorToData *Impute clusters from senators to data*

Description

Assign trajetories from the original data to senator clusters.

Usage

```
imputeSenatorToData(senators, clusters)
```

Arguments

senators	List object obtained from imputeSenator function
clusters	Pam object obtained from getClusters or combineCluster.

Details

When it's computed a clustering over senators, it's necessary to assign those cluster to original data. To do this, it's known which senator each original trajectory belong to, therefore the final cluster of each senator is identified and the trajectories of that senator are assigned to its definitive cluster.

Value

Object of imputeSenator-class.

Author(s)

Fernando Pérez-Sanz (<fernando.perez8@um.es>)

Miriam Riquelme-Pérez (<miriam.riquelmep@gmail.com>)

See Also

plotClusterSenator, imputeSenators, getClusters, imputeSenator-class.

Examples

```
data( tscR )
data <- tscR
time <- c( 1, 2, 3 )
senators <- imputeSenators( data, k = 100 )
senatorDist <- slopeDist( senators$senatorData, time )
sClust <- getClusters( senatorDist, k = 5 )
plotCluster( senators$senatorData, sClust, 2 )
endCluster <- imputeSenatorToData( senators, sClust )
```

---

plotCluster *Plot trajectories based on clustering*

Description

Draw trajectories and are colored based on their clusters

Usage

```
plotCluster(data, clust, ncluster, ...)
```

Arguments

data	Numeric data frame or matrix with de original data. SummarizedExperiment object can be provided for compatibility with bioconductor container (for more information see vignette).
clust	Object of class pam or partition obtained from getClusters output.
ncluster	When nclust = 'all', plots all trajectories and cluster together in a single plot. If it's an integer, it draws only trajectories that belong to that cluster. Finally, if it is a numeric vector, it draws trajectories corresponding to each cluster within a subplot.
...	Other arguments to pass to importFromSE if _x_ is SummarizedExperiment-class.

Details

It draws trajectories where x axis is time data and y axis trayectory values.

Value

Plot clustered trayectories

Author(s)

Fernando Pérez-Sanz (<fernando.perez8@um.es>)

Miriam Riquelme-Pérez (<miriam.riquelmep@gmail.com>)

See Also

matplot, plotClusterSenator, importFromSE.

Examples

```
data(tscR)
data <- tscR
time <- c(1,2,3)
fdist <- frechetDistC(data, time)
fclust <- getClusters(fdist, 3)
plotCluster(data, fclust, 'all')
```

---

plotClusterSenator *Plot trajectories based on clustering*

Description

Draw trajectories and are colored based on their clusters with imputesenator object

Usage

```
plotClusterSenator(x, ncluster)
```

Arguments

x	imputesenator class object from imputeSenatorToData function.
ncluster	When nclust = 'all', plots all trajectories and cluster together in a single plot. If it's an integer, it draws only trajectories that belong to that cluster. Finally, if it is a numeric vector, it draws trajectories corresponding to each cluster within a subplot.

Details

It draws trajectories where x axis is time data and y axis trayectory values.

Value

Plot clustered trayectories

Examples

```
data( tscR )
data <- tscR
time <- c( 1, 2, 3 )
senators <- imputeSenators( data, k = 100 )
senatorDist <- slopeDist( senators$senatorData, time )
sClust <- getClusters( senatorDist, k = 5 )
endCluster <- imputeSenatorToData( senators, sClust )
plotClusterSenator( endCluster, 'all' )
```

---

slopeDist *Pairwise slope distance*

Description

Compute pairwise distance based on slopes in a matrix of trajectories

Usage

```
slopeDist(x, time, ...)
```

Arguments

x	Numeric matrix or data.frame with trajectory values. Rows are trajectories, columns are time or similar. SummarizedExperiment object can be provided for compatibility with bioconductor container (for more information see vignette).
time	Numeric vector with time data (time intervals), with equal length to columns number in x.
...	Other arguments to pass to importFromSE if _x_ is SummarizedExperiment-class.

Value

A dist class object of size NxN, where N is rows number in the input data

Author(s)

Fernando Pérez-Sanz (<fernando.perez8@um.es>)

Miriam Riquelme Pérez (<miriam.riquelmep@gmail.com>)

See Also

frechetDistC and frechetDist (R and slower versión than frechetDistC.),importFromSE.

Examples

```
data(tscR)
data <- tscR
time <- c(1,2,3)
dist_tscR <- slopeDist(data, time)
```

---

tscR *Dummy trajectories data*

Description

A dataset containing 300 tajectories and 3 time points

Usage

```
tscR
```

Format

A data frame 300 rows and 3 columns:

T1 time interval

T2 time interval

T3 time interval

Details

This dataset has been created specifically to be able to illustrate the operation of the package with different distance metrics. Thus, from 3-4 hand-created trajectories (ascending, descending, quasi-horizontal) we have generated 300 trajectories with random variations from the original ones. The code used was similar to the one attached here:

Source

Simulated data

Examples

```
df <- data.frame(T1 = c(4,3.9,4.1,4),
                 T2=c(5.5, 4.3, 3.7, 2.5),
                 T3 = c(7, 3.9,4.1, 1))
df1 <- matrix(NA, nrow=100, ncol=3)
df2 <- matrix(NA, nrow=100, ncol=3)
df3 <- matrix(NA, nrow=100, ncol=3)
df4 <- matrix(NA, nrow=100, ncol=3)
for(i in seq(1,75)){
 df1[i,] <- jitter(as.numeric(df[1,]), factor = 2.5)
 df2[i,] <- jitter(as.numeric(df[2,]), factor = 7.5)
 df3[i,] <- jitter(as.numeric(df[3,]), factor = 7.5)
 df4[i,] <- jitter(as.numeric(df[4,]), factor = 2.5)
}
df <- as.data.frame(rbind(df1,df2,df3, df4))
names(df) <- c("T1","T2","T3")
```

Index

∗ **classes**
imputeSenator-class, 8
∗ **datasets**
tscR, 13

clara, *8*
combineCluster, 2, *9*

distFrechet, *4, 5*

frechetDist, 3, *5*, *13*
frechetDistC, *4*, 4, *13*

getClusters, *2*, 5, *9, 10*

importFromSE, *4, 5*, 6, *9*, *11*, *13*
imputeSenator-class, 7, *10*
imputeSenators, *8*, 8, *10*
imputeSenatorToData, *9*, 9, *11*

matplot, *11*

pam, *5*
pam.object, *2*, *5*
plotCluster, *2*, *5*, 10
plotClusterSenator, *9–11*, 11

show,imputeSenator-method
*(*imputeSenator-class*)*, 8
slopeDist, *4, 5*, 12

tscR, 13