Package ‘PIUMA’

May 30, 2024

Type Package
Title Phenotypes Identification Using Mapper from topological data Analysis
Version 1.0.0
Description The PIUMA package offers a tidy pipeline of Topological Data Analysis frameworks to identify and characterize communities in high and heterogeneous dimensional data.
License GPL-3 + file LICENSE
Encoding UTF-8
LazyData false
biocViews Clustering, GraphAndNetwork, DimensionReduction, Network, Classification
VignetteBuilder knitr
Imports cluster, umap, tsne, kernlab, vegan, dbscan, igraph, scales, Hmisc, patchwork, grDevices, stats, methods, SummarizedExperiment
Suggests BiocStyle, knitr, testthat, rmarkdown
Depends R (>= 4.3), ggplot2
RoxygenNote 7.2.3
URL https://github.com/BioinfoMonzino/PIUMA
BugReports https://github.com/BioinfoMonzino/PIUMA/issues
git_url https://git.bioconductor.org/packages/PIUMA
git_branch RELEASE_3_19
git_last_commit 0ae5da2
git_last_commit_date 2024-04-30
Repository Bioconductor 3.19
Date/Publication 2024-05-29
### Author
Mattia Chiesa [aut, cre](https://orcid.org/0000-0001-7427-9954),
Arianna Dagliati [aut](https://orcid.org/0000-0002-5041-0409),
Alessia Gerbasi [aut](https://orcid.org/0000-0003-4501-1777),
Giuseppe Albi [aut],
Laura Ballarini [aut],
Luca Piacentini [aut](https://orcid.org/0000-0003-1022-4481)

### Maintainer
Mattia Chiesa <mattia.chiesa@cardiologicomonzino.it>

### Contents

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>checkNetEntropy</td>
<td>3</td>
</tr>
<tr>
<td>checkScaleFreeModel</td>
<td>4</td>
</tr>
<tr>
<td>dfToDistance</td>
<td>5</td>
</tr>
<tr>
<td>dfToProjection</td>
<td>6</td>
</tr>
<tr>
<td>df_test_proj</td>
<td>7</td>
</tr>
<tr>
<td>getComp</td>
<td>8</td>
</tr>
<tr>
<td>getDfMapper</td>
<td>8</td>
</tr>
<tr>
<td>getDistMat</td>
<td>9</td>
</tr>
<tr>
<td>getJacc</td>
<td>10</td>
</tr>
<tr>
<td>getNodeDataMat</td>
<td>10</td>
</tr>
<tr>
<td>getOrigData</td>
<td>11</td>
</tr>
<tr>
<td>getOutcome</td>
<td>12</td>
</tr>
<tr>
<td>getOutcomeFact</td>
<td>12</td>
</tr>
<tr>
<td>getScaledData</td>
<td>13</td>
</tr>
<tr>
<td>jaccardMatrix</td>
<td>14</td>
</tr>
<tr>
<td>makeTDAobj</td>
<td>15</td>
</tr>
<tr>
<td>makeTDAobjFromSE</td>
<td>16</td>
</tr>
<tr>
<td>mapperCore</td>
<td>17</td>
</tr>
<tr>
<td>PIUMA</td>
<td>18</td>
</tr>
<tr>
<td>setComp</td>
<td>19</td>
</tr>
<tr>
<td>setDfMapper</td>
<td>19</td>
</tr>
<tr>
<td>setDistMat</td>
<td>20</td>
</tr>
<tr>
<td>setJacc</td>
<td>21</td>
</tr>
<tr>
<td>setNodeDataMat</td>
<td>21</td>
</tr>
<tr>
<td>setOrigData</td>
<td>22</td>
</tr>
<tr>
<td>setOutcome</td>
<td>23</td>
</tr>
<tr>
<td>setOutcomeFact</td>
<td>23</td>
</tr>
<tr>
<td>setScaledData</td>
<td>24</td>
</tr>
<tr>
<td>tdaDfEnrichment</td>
<td>25</td>
</tr>
<tr>
<td>TDAobj-class</td>
<td>26</td>
</tr>
<tr>
<td>tda_test_data</td>
<td>26</td>
</tr>
<tr>
<td>vascEC_meta</td>
<td>27</td>
</tr>
<tr>
<td>vascEC_norm</td>
<td>27</td>
</tr>
</tbody>
</table>

### Index
29
checkNetEntropy

Compute the Network Entropy

Description

This function computes the average of the entropies for each node of a network.

Usage

checkNetEntropy(outcome_vect)

Arguments

outcome_vect  A vector containing the average outcome values for each node of a network.

Details

The average of the entropies is related to the amount of information stored in the network.

Value

The network entropy using each node of a network.

Author(s)

Mattia Chiesa, Laura Ballarini, Luca Piacentini

See Also

makeTDAobj, dfToDistance, dfToProjection, mapperCore, jaccardMatrix, tdaDfEnrichment

Examples

# use example data:
set.seed(1)
entropy <- checkNetEntropy(round(runif(10),0))
checkScaleFreeModel  
Assessment of Scale-Free model fitting

Description
This function assesses the fitting to a scale-free net model.

Usage
checkScaleFreeModel(x, showPlot = FALSE)

Arguments
x  
A TDAobj object, processed by the jaccardMatrix
showPlot  
Whether the plot has to be generated. Default: FALSE

Details
The scale-free networks show a high negative correlation between k and p(k).

Value
A list containing:
- the estimated gamma value
- The correlation between the k and the degree distribution p(k).
- The p-value of the correlation between the k and the degree distribution p(k).
- The correlation between the logarithm (base 10) of k and the logarithm (base 10) of the degree distribution p(k).
- The p-value of the correlation between the logarithm (base 10) of k and the logarithm (base 10) of the degree distribution p(k).

Author(s)
Mattia Chiesa, Laura Ballarini, Luca Piacentini

See Also
makeTDAobj, dfToDistance, dfToProjection, mapperCore, jaccardMatrix

Examples
## use example data:
data(tda_test_data)
#netModel <- checkScaleFreeModel(tda_test_data)
**dfToDistance**

*Compute the Distance Matrix from TDAobj*

**Description**

This function returns the distance matrix computed by using the Pearson’s, Euclidean or Gower distance methods. The distances are computed between the rows of a data.frame in the classical form $n \times m$, where $n$ (rows) are observations and $m$ (columns) are features.

**Usage**

```
dfToDistance(x, distMethod = c("euclidean", "gower", "pearson"))
```

**Arguments**

- **x**: A TDAobj object, generated by `makeTDAobj`. Rows ($n$) and columns ($m$) should be, respectively, observations and features.
- **distMethod**: The distance method to calculate the distance matrix. "euclidean", "gower" and "pearson" values are allowed. Default: "euclidean".

**Value**

The starting TDAobj object, in which the computed distance matrix has been added (slot: ‘dist_mat’)

**Author(s)**

Mattia Chiesa, Laura Ballarini, Luca Piacentini

**See Also**

- `makeTDAobj`

**Examples**

```r
## use example data:
data(tda_test_data)
dfDist <- dfToDistance(tda_test_data, "euclidean")
```
dfToProjection

Data projection using a Dimensionality Reduction Method

Description

This function performs the transformation of data from a high dimensional space into a low dimensional space, wrapping 6 well-known reduction methods; i.e., PCA, KPCA, t-SNE, UMAP, MDS, and Isomap. In the topological data analysis, the identified components are commonly used as lenses.

Usage

```
dfToProjection(
  x,
  method = c("PCA", "UMAP", "TSNE", "MDS", "KPCA", "ISOMAP"),
  nComp = 2,
  centerPCA = FALSE,
  scalePCA = FALSE,
  umapNNeigh = 15,
  umapMinDist = 0.1,
  tsnePerpl = 30,
  tsneMaxIter = 300,
  kpcaKernel = c("rbfdot", "laplacedot", "polydot", "tanhdot", "besseldot", "anovadot",
                 "vanilladot", "splinedot"),
  kpcaSigma = 0.1,
  kpcaDegree = 1,
  isomNNeigh = 5,
  showPlot = FALSE,
  vectColor = NULL
)
```

Arguments

- `x`: A TDAobj object, generated by `makeTDAobj`
- `method`: Name of the dimensionality reduction method to use. "PCA", "UMAP", "TSNE", "MDS", "KPCA" and "isomap" values are allowed. Default is: "PCA".
- `nComp`: The number of components to be computed. Default: 2
- `centerPCA`: Whether the data should be centered before PCA. Default: TRUE
- `scalePCA`: Whether the data should be scaled before PCA. Default: TRUE
- `umapNNeigh`: The number of neighbors for UMAP. Default: 15
- `umapMinDist`: The minimum distance between points for UMAP. Default: 0.1
- `tsnePerpl`: Perplexity argument of t-SNE. Default: 30
- `tsneMaxIter`: The maximum number of iterations for t-SNE. Default: 300
- `kpcaKernel`: The type of kernel for kPCA. "rbfdot", "laplacedot", "polydot", "tanhdot", "besseldot", "anovadot", "vanilladot" and "splinedot" are allowed. Default: "polydot".
df_test_proj

kpcaSigma  The 'sigma' argument for kPCA. Default: 0.1.
kpcaDegree  The 'degree' argument for kPCA. Default: 1.
isomNNeigh  The number of neighbors for Isomap. Default: 5.
showPlot  Whether the scatter plot of the first two principal components should be shown. Default: TRUE.
vectColor  Vector containing the variable to color the scatter plot. Default: NULL.

Value

The starting TDAobj object, in which the principal components of projected data have been added (slot:'comp')

Author(s)

Mattia Chiesa, Laura Ballarini, Luca Piacentini

See Also

makeTDAobj, dfToDistance

Examples

# use example data:
data(tda_test_data)
set.seed(1)
cmp <- dfToProjection(tda_test_data, "PCA", nComp=2)

---

df_test_proj

A dataset to test the dfToProjection and dfToDistance functions of PIUMA package.

Description

A dataset to test the dfToProjection and dfToDistance functions of PIUMA package.

Usage

df_test_proj

Format

A data.frame containing 15 rows (cells) and 15 columns (genes)

Value

An example dataset for PIUMA package
**getComp**  
*Getter method for the 'comp' slot of a TDAobj object.*

**Description**  
The method to get data from the comp slot

**Usage**  
```r  
getComp(x)  
```

```r  
## S4 method for signature 'TDAobj'  
getComp(x)  
```

**Arguments**  
- `x`  
a TDAobj object

**Value**  
a data.frame with the comp data

**Author(s)**  
Mattia Chiesa

**Examples**  
```r  
data(tda_test_data)  
```

---

**getDfMapper**  
*Getter method for the 'dfMapper' slot of a TDAobj object.*

**Description**  
The method to get data from the dfMapper slot

**Usage**  
```r  
getDfMapper(x)  
```

```r  
## S4 method for signature 'TDAobj'  
getDfMapper(x)  
```

**Arguments**  
- `x`  
a TDAobj object
Value

a data.frame with the dfMapper data

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)
ex_out <- getDfMapper(tda_test_data)

getDistMat

Getter method for the 'dist_mat' slot of a TDAobj object.

Description

The method to get data from the dist_mat slot

Usage

getDistMat(x)

## S4 method for signature 'TDAobj'
getDistMat(x)

Arguments

x a TDAobj object

Value

a data.frame with the dist_mat data

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)
ex_out <- getDistMat(tda_test_data)
**getNodeDataMat**

*Getter method for the 'node_data_mat' slot of a TDAobj object.*

**Description**

The method to get data from the node_data_mat slot

**Usage**

```
getNodeDataMat(x)
```

```
## S4 method for signature 'TDAobj'
getNodeDataMat(x)
```

**Arguments**

- `x` a TDAobj object

**Value**

a matrix with the node_data_mat data

**Author(s)**

Mattia Chiesa

**Examples**

```
data(tda_test_data)
ex_out <- getNodeDataMat(tda_test_data)
```
getOrigData

Arguments

x a TDAobj object

Value

a data.frame with the node_data_mat data

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)
ex_out <- getNodeDataMat(tda_test_data)

getOrigData

Getter method for the 'orig_data' slot of a TDAobj object.

Description

The method to get data from the orig_data slot

Usage

getOrigData(x)

## S4 method for signature 'TDAobj'
getOrigData(x)

Arguments

x a TDAobj object

Value

a data.frame with the original data

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)
ex_out <- getOrigData(tda_test_data)
getOutcome

*Getter method for the 'outcome' slot of a TDAobj object.*

**Description**

The method to get data from the outcome slot

**Usage**

```r
getOutcome(x)
```

```r
## S4 method for signature 'TDAobj'
getOutcome(x)
```

**Arguments**

- `x`: a TDAobj object

**Value**

a data.frame with the outcome data

**Author(s)**

Mattia Chiesa

**Examples**

```r
data(tda_test_data)
ex_out <- getOutcome(tda_test_data)
```

---

getOutcomeFact

*Getter method for the 'outcomeFact' slot of a TDAobj object.*

**Description**

The method to get data from the outcomeFact slot

**Usage**

```r
getOutcomeFact(x)
```

```r
## S4 method for signature 'TDAobj'
getOutcomeFact(x)
```

**Arguments**

- `x`: a TDAobj object

**Examples**

```r
data(tda_test_data)
ex_out <- getOutcomeFact(tda_test_data)
```
getScaledData

Arguments

x a TDAobj object

Value

a data.frame with the outcomeFact data

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)
ex_out <- getOutcomeFact(tda_test_data)

getSocketAddress

Getter method for the 'scaled_data' slot of a TDAobj object.

description

The method to get data from the scaled_data slot

Usage

getScaledData(x)

## S4 method for signature 'TDAobj'
getSocketAddress(x)

Arguments

x a TDAobj object

Value

a data.frame with the scaled data

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)
ex_out <- getScaledData(tda_test_data)
jaccardMatrix

**Compute the Matrix of Jaccard Indexes**

**Description**

This function computes the Jaccard index for each pair of nodes contained in TDAobj, generated by the `mapperCore` function. The resulting data.frame can be used to represent data as a network, for instance, in Cytoscape.

**Usage**

```r
jaccardMatrix(x)
```

**Arguments**

- `x` A TDAobj object, processed by the `mapperCore` function.

**Details**

The Jaccard index measures the similarity of two nodes A and B. It ranges from 0 to 1. If A and B share no members, their Jaccard index would be 0 (= NA). If A and B share all members, their Jaccard index would be 1. Hence, the higher the index, the more similar the two nodes. If the Jaccard index between A and B is different from NA, it means that an edge exists between A and B. The output matrix of Jaccard indexes can be used as an adjacency matrix. The resulting data.frame can be used to represent data as a network, for instance, in Cytoscape.

**Value**

The starting TDAobj object, in which the matrix of Jaccard indexes, calculated comparing each node of the ‘dfMapper’ slot, has been added (slot: ‘jacc’)

**Author(s)**

Mattia Chiesa, Laura Ballarini, Luca Piacentini

**See Also**

- `makeTDAobj`
- `dfToDistance`
- `dfToProjection`
- `mapperCore`

**Examples**

```r
## use example data:
data(tda_test_data)
jacc_mat <- jaccardMatrix(tda_test_data)
```
makeTDAobj

Import data and generate the TDAobj object

Description
This function import a data.frame and create the object to store all data needed for TDA analysis. In addition, some preliminary preprocess steps are performed; specifically, outcomes variables data will be separated the rest of dataset. The remaining dataset will be also re-scaled (0-1)

Usage
makeTDAobj(df, outcomes)

Arguments
- **df** A data.frame representing a dataset in the classical n x m form. Rows (n) and columns (m) should be, respectively, observations and features.
- **outcomes** A string or vector of string containing the name of variables that have to be considered 'outcomes'

Value
A TDA object containing:
- `orig_data` A data.frame of original data (without outcomes)
- `scaled_data` A data.frame of re-scaled data (without outcomes)
- `outcomeFact` A data.frame of original outcomes
- `outcome` A data.frame of original outcomes converted as numeric
- `comp` A data.frame containing the components of projected data
- `dist_mat` A data.frame containing the computed distance matrix
- `dfMapper` A data.frame containing the nodes, with their elements, identified by TDA
- `jacc` A matrix of Jaccard indexes between each pair of dfMapper nodes
- `node_data_mat` A data.frame with the node size and the average value

Author(s)
Mattia Chiesa, Laura Ballarini, Luca Piacentini

Examples
```r
## use example data:
data("vascEC_meta")
data("vascEC_norm")
df <- cbind(vascEC_meta,vascEC_norm)
res <- makeTDAobj(df, "zone")
```
Description

This function import a SummarizedExperiment object and create the object to store all data needed for TDA analysis. In addition, some preliminary preprocess steps are performed; specifically, outcomes variables data will be separated the rest of dataset. The remaining dataset will be also re-scaled (0-1).

Usage

makeTDAobjFromSE(SE, outcomes)

Arguments

SE A SummarizedExperiment object
outcomes A string or vector of string containing the name of variables that have to be considered 'outcomes'

Value

A TDA object containing:

- orig_data A data.frame of original data (without outcomes)
- scaled_data A data.frame of re-scaled data (without outcomes)
- outcomeFact A data.frame of original outcomes
- outcome A data.frame of original outcomes converted as numeric
- comp A data.frame containing the components of projected data
- dist_mat A data.frame containing the computed distance matrix
- dfMapper A data.frame containing the nodes, with their elements, identified by TDA
- jacc A matrix of Jaccard indexes between each pair of dfMapper nodes
- node_data_mat A data.frame with the node size and the average value

Author(s)

Mattia Chiesa, Laura Ballarini, Luca Piacentini

Examples

```r
## use example data:
data("vascEC_meta")
data("vascEC_norm")
suppressMessages(library(SummarizedExperiment))
dataSE <- SummarizedExperiment(assays=as.matrix(t(vascEC_norm)),
colData=as.data.frame(vascEC_meta))
res <- makeTDAobjFromSE(dataSE, "zone")
```
mapperCore

Implement the TDA Mapper algorithm on TDAobj

Description

This is a comprehensive function permitting to perform the core TDA Mapper algorithm with 2D lenses. It allow setting several types of clustering methods.

Usage

mapperCore(
  x,
  nBins = 15,
  overlap = 0.4,
  mClustNode = 2,
  remEmptyNode = TRUE,
  clustMeth = c("kmeans", "HR", "DBSCAN", "OPTICS"),
  HRMethod = c("average", "complete")
)

Arguments

x
A TDAobj object, processed by the dfToDistance and dfToProjection functions.

nBins
The number of bins (i.e. the resolution of the cover). Default: 15.

overlap
The overlap between bins (i.e. the gain of the cover). Default: 0.4.

mClustNode
The number of clusters in each overlapping bin. Default: 2

remEmptyNode
A logical value to remove or not the empty nodes from the resulting data.frame. Default: TRUE.

clustMeth
The clustering algorithm. "HR", "kmeans", "DBSCAN", and "OPTICS" are allowed. Default: "kmeans".

HRMethod
The name of the linkage criterion (when clustMeth="HR"). "average" and "complete" values are allowed. Default: "average".

Value

The starting TDAobj object, in which the result of mapper algorithm (inferred nodes with their elements) has been added (slot: 'dfMapper')

A data.frame containing the clusters, with their elements, identified by TDA .

Author(s)

Mattia Chiesa, Laura Ballarini, Luca Piacentini
PIUMA: Phenotypes Identification Using Mapper from topological data Analysis

Description

The application of unsupervised learning methodologies could help the identification of specific phenotypes in huge heterogeneous cohorts, such as clinical or -omics data. Among them, the Topological Data Analysis (TDA) is a rapidly growing field that combines concepts from algebraic topology and computational geometry to analyze and extract meaningful information from complex and high-dimensional data sets. Moreover, TDA is a robust and effective methodology, able to preserve the intrinsic characteristics of data and the mutual relations among observations, depicting complex data in a graph-based representation. Indeed, building topological models as networks, TDA allows complex diseases to be inspected in a continuous space, where subjects can fluctuate over the graph, sharing, at the same time, more than one adjacent node of the network. Overall, TDA offers a powerful set of tools to capture the underlying topological features of data, revealing essential patterns and relationships that might be hidden from traditional statistical techniques. The PIUMA package (Phenotypes Identification Using Mapper from topological data Analysis) allows implementing all the main steps of a Topological Data Analysis. PIUMA is the italian word meaning 'feather'.

Details

See the package vignette, by typing vignette("PIUMA") to discover all the functions.

Author(s)

Mattia Chiesa, Laura Ballarini, Luca Piacentini
**setComp**

*Setter method for the 'comp' slot of a TDAobj object.*

**Description**

The method to set the comp slot.

**Usage**

```r
setComp(x, y)
```

## S4 method for signature 'TDAobj'

```r
setComp(x, y)
```

**Arguments**

- **x**
  - a TDAobj object
- **y**
  - a data.frame with the comp data

**Value**

a TDAobj object

**Author(s)**

Mattia Chiesa

**Examples**

```r
data(tda_test_data)
```

---

**setDfMapper**

*Setter method for the 'dfMapper' slot of a TDAobj object.*

**Description**

The method to set the dfMapper slot.

**Usage**

```r
setDfMapper(x, y)
```

## S4 method for signature 'TDAobj'

```r
setDfMapper(x, y)
```

**Examples**

```r
```
Arguments

x  a TDAobj object
y  a data.frame with the dfMapper data

Value

a TDAobj object

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)

---

setDistMat  

 Setter method for the 'dist_mat' slot of a TDAobj object.

---

Description

The method to set the dist_mat slot

Usage

setDistMat(x, y)

## S4 method for signature 'TDAobj'
setDistMat(x, y)

Arguments

x  a TDAobj object
y  a data.frame with the dist_mat data

Value

a TDAobj object

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)
**setJacc**

*Setter method for the 'jacc' slot of a TDAobj object.*

**Description**

The method to set the jacc slot

**Usage**

```
setJacc(x, y)
```

```
## S4 method for signature 'TDAobj'
setJacc(x, y)
```

**Arguments**

- `x`: a TDAobj object
- `y`: a matrix with the jacc data

**Value**

a TDAobj object

**Author(s)**

Mattia Chiesa

**Examples**

```
data(tda_test_data)
```

---

**setNodeDataMat**

*Setter method for the 'node_data_mat' slot of a TDAobj object.*

**Description**

The method to set the node_data_mat slot

**Usage**

```
setNodeDataMat(x, y)
```

```
## S4 method for signature 'TDAobj'
setNodeDataMat(x, y)
```

**Examples**

```
data(tda_test_data)
```
setOrigData

Arguments

x a TDAobj object
y a data.frame with the node_data_mat data

Value

a TDAobj object

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)

---

setOrigData Setter method for the 'orig_data' slot of a TDAobj object.

Description

The method to set the orig_data slot

Usage

setOrigData(x, y)

## S4 method for signature 'TDAobj'
setOrigData(x, y)

Arguments

x a TDAobj object
y a data.frame with the original data

Value

a TDAobj object

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)
**setOutcome**  
*Setter method for the 'outcome' slot of a TDAobj object.*

**Description**  
The method to set the outcome slot

**Usage**  
```r
setOutcome(x, y)
```

## S4 method for signature 'TDAobj'
```r
setOutcome(x, y)
```

**Arguments**  
- `x`: a TDAobj object
- `y`: a data.frame with the outcome data

**Value**  
a TDAobj object

**Author(s)**  
Mattia Chiesa

**Examples**  
```r
data(tda_test_data)
```

---

**setOutcomeFact**  
*Setter method for the 'outcomeFact' slot of a TDAobj object.*

**Description**  
The method to set the outcomeFact slot

**Usage**  
```r
setOutcomeFact(x, y)
```

## S4 method for signature 'TDAobj'
```r
setOutcomeFact(x, y)
```
Arguments

- `x`: a `TDAobj` object
- `y`: a `data.frame` with the outcomeFact data

Value

- a `TDAobj` object

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)

---

**setScaledData**  
*Setter method for the 'scaled_data' slot of a TDAobj object.*

Description

The method to set the scaled_data slot

Usage

```r
setScaledData(x, y)
```

## S4 method for signature 'TDAobj'
setScaledData(x, y)

Arguments

- `x`: a `TDAobj` object
- `y`: a `data.frame` with the scaled data

Value

- a `TDAobj` object

Author(s)

Mattia Chiesa

Examples

data(tda_test_data)
tdaDfEnrichment

---

**Add information to TDAobj**

**Description**

This function computes the average value of additional features provided by the user and calculate the size for each node of 'dfMapper' slot.

**Usage**

```r
tdaDfEnrichment(x, df)
```

**Arguments**

- `x`: A TDAobj object, processed by the `mapperCore` function.
- `df`: A data.frame with scaled values in the classical n x m form: rows (n) and columns (m) must be observations and features, respectively.

**Value**

The starting TDAobj object, in which the a data.frame with additional information for each node has been added (slot: 'node_data_mat')

**Author(s)**

Mattia Chiesa, Laura Ballarini, Luca Piacentini

**See Also**

`makeTDAobj`, `dfToDistance`, `dfToProjection`, `mapperCore`, `jaccardMatrix`

**Examples**

```r
## use example data:
data(tda_test_data)
data(df_test_proj)
enrich_mat_tda <- tdaDfEnrichment(tda_test_data, df_test_proj)
```
TDAobj-class

The object ‘TDAobj’

Description

The TDA object for storing TDA data

Value

TDAobj class showClass("TDAobj")

Slots

orig_data A data.frame of original data (without outcomes)
scaled_data A data.frame of re-scaled data (without outcomes)
outcomeFact A data.frame of original outcomes
outcome A data.frame of original outcomes converted as numeric
comp A data.frame containing the components of projected data
dist_mat A data.frame containing the computed distance matrix
dfMapper A data.frame containing the nodes, with their elements, identified by TDA
jacc A matrix of Jaccard indexes between each pair of dfMapper nodes
node_data_mat A data.frame with the node size and the average value of each feature

tda_test_data

A TDAobj to test the PIUMA package.

Description

A TDAobj to test the PIUMA package.

Usage

tda_test_data

Format

A TDAobj with data in all slots

Value

An example dataset for PIUMA package
Example datasets for PIUMA package

Description

We tested PIUMA on a subset of the single-cell RNA Sequencing dataset (GSE:GSE193346 generated and published by Feng et al. (2022) on Nature Communication to demonstrate that distinct transcriptional profiles are present in specific cell types of each heart chambers, which were attributed to have roles in cardiac development. In this tutorial, our aim will be to exploit PIUMA for identifying sub-population of vascular endothelial cells, which can be associated with specific heart developmental stages. The original dataset consisted of three layers of heterogeneity: cell type, stage and zone (i.e., heart chamber). Our testing dataset was obtained by subsetting vascular endothelial cells (cell type) by Seurat object, extracting raw counts and metadata. Thus, we filtered low expressed genes and normalized data by DaMiRseq.

Usage

vascEC_meta

Format

A dataframe containing 1180 rows (cells) and 2 columns (outcomes)

Value

An example dataset for PIUMA package

We tested PIUMA on a subset of the single-cell RNA Sequencing dataset (GSE:GSE193346 generated and published by Feng et al. (2022) on Nature Communication to demonstrate that distinct transcriptional profiles are present in specific cell types of each heart chambers, which were attributed to have roles in cardiac development. In this tutorial, our aim will be to exploit PIUMA for identifying sub-population of vascular endothelial cells, which can be associated with specific heart developmental stages. The original dataset consisted of three layers of heterogeneity: cell type, stage and zone (i.e., heart chamber). Our testing dataset was obtained by subsetting vascular endothelial cells (cell type) by Seurat object, extracting raw counts and metadata. Thus, we filtered low expressed genes and normalized data by DaMiRseq.
Description

We tested PIUMA on a subset of the single-cell RNA Sequencing dataset (GSE:GSE193346 generated and published by Feng et al. (2022) on Nature Communication to demonstrate that distinct transcriptional profiles are present in specific cell types of each heart chambers, which were attributed to have roles in cardiac development. In this tutorial, our aim will be to exploit PIUMA for identifying sub-population of vascular endothelial cells, which can be associated with specific heart developmental stages. The original dataset consisted of three layers of heterogeneity: cell type, stage and zone (i.e., heart chamber). Our testing dataset was obtained by subsetting vascular endothelial cells (cell type) by Seurat object, extracting raw counts and metadata. Thus, we filtered low expressed genes and normalized data by DaMiRseq.

Usage

vascEC_norm

Format

A matrix containing 1180 rows (cells) and 838 columns (genes)

Value

An example dataset for PIUMA package
Index

* datasets
  df_test_proj, 7
  tda_test_data, 26
  vascEC_meta, 27
  vascEC_norm, 27

* package
  PIUMA, 18

checkNetEntropy, 3
checkScaleFreeModel, 4

df_test_proj, 7
dfToDistance, 3, 4, 5, 7, 14, 17, 18, 25
dfToProjection, 3, 4, 6, 7, 14, 17, 18, 25

getComp, 8
getComp, PIUMA-getComp (getComp), 8
getComp, TDAobj-method (getComp), 8
getDfMapper, 8
getDfMapper, PIUMA-getDfMapper (getDfMapper), 8
getDfMapper, TDAobj-method (getDfMapper), 8
getDistMat, 9
getDistMat, PIUMA-getDistMat (getDistMat), 9
getDistMat, TDAobj-method (getDistMat), 9
getJacc, 10
getJacc, PIUMA-getJacc (getJacc), 10
getJacc, TDAobj-method (getJacc), 10
getNodeDataMat, 10
getNodeDataMat, PIUMA-getNodeDataMat (getNodeDataMat), 10
getNodeDataMat, TDAobj-method (getNodeDataMat), 10
getOrigData, 11
getOrigData, PIUMA-getOrigData (getOrigData), 11
getOrigData, TDAobj-method (getOrigData), 11

getOutcome, 12
getOutcome, PIUMA-getOutcome (getOutcome), 12
getOutcome, TDAobj-method (getOutcome), 12

getOutcomeFact, 12
getOutcomeFact, PIUMA-getOutcomeFact (getOutcomeFact), 12
getOutcomeFact, TDAobj-method (getOutcomeFact), 12

getScaledData, 13
getScaledData, PIUMA-getScaledData (getScaledData), 13
getScaledData, TDAobj-method (getScaledData), 13

jaccardMatrix, 3, 4, 14, 25

makeTDAobj, 3–7, 14, 15, 18, 25
makeTDAobjFromSE, 16
mapperCore, 3, 4, 14, 17, 25

PIUMA, 18

setComp, 19
setComp, PIUMA-setComp (setComp), 19
setComp, TDAobj-method (setComp), 19
setDfMapper, 19
setDfMapper, PIUMA-setDfMapper (setDfMapper), 19
setDfMapper, TDAobj-method (setDfMapper), 19

setDistMat, 20
setDistMat, PIUMA-setDistMat (setDistMat), 20
setDistMat, TDAobj-method (setDistMat), 20

setJacc, 21
setJacc, PIUMA-setJacc (setJacc), 21
setJacc, TDAobj-method (setJacc), 21
setNodeDataMat, 21
setNodeDataMat, PIUMA-setNodeDataMat (setNodeDataMat), 21
setNodeDataMat, TDAobj-method (setNodeDataMat), 21
setOrigData, 22
setOrigData, PIUMA-setOrigData (setOrigData), 22
setOrigData, TDAobj-method (setOrigData), 22
setOutcome, 23
setOutcome, PIUMA-setOutcome (setOutcome), 23
setOutcome, TDAobj-method (setOutcome), 23
setOutcomeFact, 23
setOutcomeFact, PIUMA-setOutcomeFact (setOutcomeFact), 23
setOutcomeFact, TDAobj-method (setOutcomeFact), 23
setScaledData, 24
setScaledData, PIUMA-setScaledData (setScaledData), 24
setScaledData, TDAobj-method (setScaledData), 24

tda_test_data, 26
tdaDfEnrichment, 3, 25
TDAobj-class, 26

vascEC_meta, 27
vascEC_norm, 27