Package 'PIUMA'

May 5, 2026

Type Package

Title Phenotypes Identification Using Mapper from topological data Analysis

Version 1.9.0

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 Hmisc, igraph, patchwork, scales, utils, cluster, umap, tsne, kernlab, vegan, dbscan, grDevices, stats, methods, SummarizedExperiment, ggplot2

Suggests BiocStyle, testthat, knitr, rmarkdown, Seurat, SingleCellExperiment, aricode, mclust, viridis, magick, ggrepel, dplyr

Depends R (>= 4.3)

RoxygenNote 7.3.1

URL https://github.com/BioinfoMonzino/PIUMA

BugReports https://github.com/BioinfoMonzino/PIUMA/issues

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

git_branch devel

git_last_commit c00b5d7

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-05-04

Author Mattia Chiesa [aut, cre] (ORCID: <https://orcid.org/0000-0001-7427-9954>), Arianna Dagliati [aut] (ORCID: <https://orcid.org/0000-0002-5041-0409>), Alessia Gerbasi [aut] (ORCID: <https://orcid.org/0000-0003-4501-1777>),

Giuseppe Albi [aut],
Laura Ballarini [aut],
Luca Piacentini [aut] (ORCID: <https://orcid.org/0000-0003-1022-4481>),
Carlo Leonardi [aut] (ORCID: <https://orcid.org/0000-0001-5348-8300>)

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

Contents

PIUMA-package *PIUMA: Phenotypes Identification Using Mapper from topological data Analysis*

Description

The PIUMA package offers a tidy pipeline of Topological Data Analysis frameworks to identify and characterize communities in high and heterogeneous dimensional data.

Author(s)

Maintainer: Mattia Chiesa <mattia.chiesa@cardiologicomonzino.it> (ORCID)

Authors:

- Arianna Dagliati <arianna.dagliati@unipv.it> (ORCID)
- Alessia Gerbasi <alessia.gerbasi01@universitadipavia.it> (ORCID)
- Giuseppe Albi <giuseppe.albi01@universitadipavia.it>
- Laura Ballarini <laura.ballarini01@universitadipavia.it>
- Luca Piacentini <luca.piacentini@cardiologicomonzino.it> (ORCID)
- Carlo Leonardi <cl35@sanger.ac.uk> (ORCID)

See Also

Useful links:

- https://github.com/BioinfoMonzino/PIUMA
- Report bugs at https://github.com/BioinfoMonzino/PIUMA/issues

---

autoClusterMapper *Automatic Clustering of a Mapper Graph by Predicted Geometry (with kNN tie-break)*

Description

Cluster Mapper nodes (x@graph$igraph) with a chosen community algorithm or an automatic selection based on predicted graph geometry (x@graph$predicted). Assign observations either by k-NN tie-breaking (default) or by pure topological label concatenation.

Usage

```
autoClusterMapper(
  x,
 method = c("automatic", "fast_greedy", "walktrap", "edge_betweenness", "optimal",
    "label_propagation"),
  k = 5L
)
```

Arguments

x	A TDAobj with x@graph$igraph and x@graph$predicted$class set.
method	One of "automatic","fast_greedy","walktrap","edge_betweenness","optimal","label_pr Default "automatic".
k	Integer >=1 or FALSE. Default 5L. If numeric, use k-NN mean distance to break ties; if FALSE, concatenate topological labels (e.g. "2_8", "1_2_3").

Details

In method = "automatic", the algorithm is chosen from the predicted geometry:

SF / CM Use *fast greedy* modularity optimization.

WS Use *Walktrap* (short random walks).

RGG Use *edge betweenness* (bridge detection).

SBM Prefer *optimal* (exact modularity) for small graphs; falls back for larger ones.

ER Use *label propagation* (fast, parameter-free).

Isolated nodes (degree = 0) become singletons with unique labels.

Value

The input TDAobj *invisibly*, with x@clustering updated:

nodes_cluster Data frame with columns node, obs, cluster.

obs_cluster Data frame with columns obs, cluster.

Author(s)

Carlo Leonardi, Mattia Chiesa

See Also

mapperCore

Examples

```
data(vascEC_norm)
data(vascEC_meta)
#df_TDA <- cbind(vascEC_meta, vascEC_norm)
#df_TDA <- makeTDAobj(df_TDA,outcomes = c("stage","zone"))
#df_TDA <- dfToDistance(df_TDA,'euclidean')
#df_TDA <- dfToProjection(df_TDA, "UMAP", nComp = 2)
#df_TDA <- mapperCore(df_TDA,
 #         nBins = 20, overlap = 0.3,
 #         mClustNode = 2, clustMeth = "kmeans")
#df_TDA <- jaccardMatrix(df_TDA)
#df_TDA <- setGraph(df_TDA)
#df_TDA <- predict_mapper_class(df_TDA)
#df_TDA <- autoClusterMapper(df_TDA,method = 'walktrap')
```

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 beztween k and p(k).

Value

A list containing:

- connectivity of the resulting graph
- the estimated gamma value
- the correlation between degree \(k\) and its 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).
- A composite score reflecting how strongly power-law behavior coexists with graph cohesion, computed as the absolute product between cor(P(k)*k) and connectivity

Author(s)

Mattia Chiesa, Laura Ballarini, Luca Piacentini, Carlo Leonardi

See Also

makeTDAobj, dfToDistance, dfToProjection, mapperCore, jaccardMatrix

Examples

```
## use example data:
data(tda_test_data)
netModel <- checkScaleFreeModel(tda_test_data)
print(netModel)
```

---

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 x 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 matix. "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

```
data(vascEC_norm)
data(vascEC_meta)
df_TDA <- cbind(vascEC_meta, vascEC_norm)
df_TDA <- makeTDAobj(df_TDA,outcomes = c("stage","zone"))
df_TDA <- dfToDistance(df_TDA,'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-knwon 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".
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 tocolor 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

```
data(vascEC_norm)
data(vascEC_meta)
df_TDA <- cbind(vascEC_meta, vascEC_norm)
df_TDA <- makeTDAobj(df_TDA,outcomes = c("stage","zone"))
df_TDA <- dfToProjection(df_TDA,'PCA',nComp=2)
```

df_test_proj *A dataset to test the* dfToProjection *and* dfToDistance *funtions of* PIUMA *package.*

Description

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

Usage

```
data(df_test_proj)
```

Format

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

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

Description

The method to get clusters from the clustering slot

Usage

```
getClusters(x)

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

Arguments

x	a TDAobj object

Value

a data.frame

Author(s)

Carlo Leonardi

Examples

```
data(tda_test_data)
```

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

Description

The method to get data from the comp slot

Usage

```
getComp(x)

## 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

```
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

```
getDfMapper(x)

## 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)
```

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

Description

The method to get igraph object from the graph slot

Usage

```
getGraph(x)

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

Arguments

x	a TDAobj object

Value

an igraph object

Author(s)

Carlo Leonardi

Examples

```
data(tda_test_data)
```

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

Description

The method to get data from the jacc slot

Usage

```
getJacc(x)

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

Arguments

x	a TDAobj object

Value

a matrix with the jacc data

Author(s)

Mattia Chiesa

Examples

```
data(tda_test_data)
ex_out <- getJacc(tda_test_data)
```

---

getMetrics *Getter method for the 'metrics' slot under 'graph' of a TDAobj object.*

Description

The method to get metrics from the graph slot

Usage

```
getMetrics(x)

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

Arguments

x	a TDAobj object

Value

a vector

Author(s)

Carlo Leonardi, Mattia Chiesa

Examples

```
data(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 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

```
getOutcome(x)

## 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

```
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

```
getOutcomeFact(x)

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

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)
```

getScaledData *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'
getScaledData(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

```
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

```
## 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
- graph A list containing the igraph object derived from Jaccard matrix and intermediary objects
- clustering A list containing two data.frames indicating the clustering per node and per rows

Author(s)

Mattia Chiesa, Laura Ballarini, Luca Piacentini, Carlo Leonardi

Examples

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

`makeTDAobjFromSE` *Import SummarizedExperiment data and generate the TDAobj object*

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
- graph A list containing the igraph object derived from Jaccard matrix and intermediary objects
- clustering A list containing two data.frames indicating the clustering per node and per rows

Author(s)

Mattia Chiesa, Laura Ballarini, Luca Piacentini, Carlo Leonardi

Examples

```
## 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. There is no restriction to `nBins` and `mClustNode`, so the user can tune those for parameter search.

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, Carlo Leonardi

See Also

makeTDAobj, dfToDistance, dfToProjection

Examples

```
# use example data:
data(vascEC_norm)
data(vascEC_meta)
df_TDA <- cbind(vascEC_meta, vascEC_norm)
df_TDA <- makeTDAobj(df_TDA,outcomes = c("stage","zone"))
df_TDA <- dfToDistance(df_TDA,'euclidean')
df_TDA <- dfToProjection(df_TDA, "PCA", nComp = 2)
df_TDA <- mapperCore(df_TDA,
          nBins = 5, overlap = 0.5,
          mClustNode = 2, clustMeth = "kmeans")
```

PIUMA *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

See Also

Useful links:

- https://github.com/BioinfoMonzino/PIUMA
- Report bugs at https://github.com/BioinfoMonzino/PIUMA/issues

---

predict_mapper_class *Predict Mapper Graph Geometry (lightweight)*

Description

Infer a geometry label for `x@graph$igraph` using fast heuristics. Writes only `x@graph$predicted$class` (one of `c("SF","RGG","WS","ER","SBM","CM")`).

#' @details Heuristics (hierarchical decision):

- **SF (relaxed)**: rely on `checkScaleFreeModel(x)`. Declare scale-free if at least one of the following holds: `|corlogklogpk|` >= `0.55`, `|corkpk|` >= `0.70`, `1.6` <= `gamma` <= `3.6`, or `Connectivity` >= `0.40`; alternatively accept SF if the product score `|corlogklogpk| * Connectivity` >= `0.2`.
- **WS**: small-world index `sigma` > `1.2` with `C/C_ER` >= `3` and `L/L_ER` <= `1.2`.
- **RGG**: very high clustering vs ER (`C/C_ER` >= `5`), longer paths (`L/L_ER` >= `1.3`), and positive degree assortativity (`r` >= `0.10`).
- **ER**: Poisson-like degree dispersion `VMR` = `var(k)/mean(k)` ~ `1` (within 30%), `|C - p|` <= `0.05`, `|r|` <= `0.05`, and `0.8` <= `sigma` <= `1.2`.
- **SBM**: strong modular structure, `Q` >= `0.40` with >= `3` communities.
- **CM**: heterogeneous degrees (`var(k)/mean(k)` > `2`) with clustering close to ER (`|C - C_ER|` <= `0.05`); otherwise use a sigma-based fallback (WS if `sigma` > `1.2`, else ER).

The function sets only `x@graph$predicted`. It is intentionally lightweight for fast computation.

Usage

```
predict_mapper_class(x, verbose = FALSE)
```

Arguments

`x`	A `TDAobj` with `x@graph$igraph` set.
`verbose`	Logical; print the chosen label. Default `FALSE`.

Value

The input `TDAobj` with `x@graph$predicted` set.

Author(s)

Carlo Leonardi, Mattia Chiesa

Examples

```
data(tda_test_data)
#tda_test_data <- predict_mapper_class(tda_test_data)
```

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

Description

The method to set the comp slot

Usage

```
setComp(x, y)

## S4 method for signature 'TDAobj'
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

```
data(tda_test_data)
```

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

Description

The method to set the dfMapper slot

Usage

```
setDfMapper(x, y)

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

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)
```

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

Description

The method to set igraph object to the graph slot

Usage

```
setGraph(x)

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

Arguments

x	a TDAobj object

Value

a TDAobj object

Author(s)

Carlo Leonardi

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)
```

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

```
setOutcome(x, y)

## S4 method for signature 'TDAobj'
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

```
data(tda_test_data)
```

---

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

Description

The method to set the outcomeFact slot

Usage

```
setOutcomeFact(x, y)

## S4 method for signature 'TDAobj'
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

```
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

```
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

```
## 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

`graph` A list containing the igraph object of your Jaccard matrix, metrics and intermediary objects

`clustering` A data.frame containing clusters from TDA on nodes and cells

tda_test_data *A TDAobj to test the* PIUMA *package.*

Description

A TDAobj with data in all slots for testing.

Usage

```
data(tda_test_data)
```

Format

A TDAobj.

vascEC_meta *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

```
data(vascEC_meta)
```

Format

A data frame with 1180 rows (cells) and 2 columns (outcomes).

vascEC_norm	*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

```
data(vascEC_norm)
```

Format

A matrix with 1180 rows (cells) and 838 columns (genes).

Index