Package ‘BallMapper’

October 12, 2022

Type   Package
Title   The Ball Mapper Algorithm
Version 0.2.0

Description The core algorithm is described in ”Ball mapper: a shape summary for topological data analysis” by Pawel Dlotko, (2019) <arXiv:1901.07410>. Please consult the following YouTube video <https://www.youtube.com/watch?v=M9Dm1nl_zSQ> for the idea of functionality. Ball Mapper provide a topologically accurate summary of a data in a form of an abstract graph. To create it, please provide the coordinates of points (in the points array), values of a function of interest at those points (can be initialized randomly if you do not have it) and the value epsilon which is the radius of the ball in the Ball Mapper construction. It can be understood as the minimal resolution on which we use to create the model of the data.

Maintainer Pawel Dlotko <pdlotko@gmail.com>

License MIT + file LICENCE

Encoding UTF-8

LazyData true

Imports igraph, scales, networkD3, testthat, fields, methods, stringr

R topics documented:

BallMapper .......................................................... 2
colorByAllVariables ............................................ 3
colorByAverageValueOfOtherVariable ....................... 4
colorByStDevValueOfOtherVariable ......................... 5
coloredDynamicNetwork ...................................... 5
ColorIgraphPlot .................................................. 6
BallMapper

Create vertices and edges (with additional properties) of a Ball Mapper graph representation of the input data. Please be aware that the program will not perform any normalization on the data. As with cluster analysis we recommend that you consider whether to normalize the data prior to running the function.

Usage

BallMapper(points, values, epsilon)

Arguments

points, a collection of input points in a form of a data frame. These are typically points in Euclidean space. By default the Euclidean distance is used to construct the Ball Mapper.

values, a collection of outcome values which apply to the data points. Mean values of this variable within any given ball will be used to color the Ball Mapper graph. If it is not available, please set it to a constant array with the same length as the number of observations in the dataset.

epsilon, the value of radius of balls used in the Ball Mapper construction.
Value

The function returns a long list of outputs which are explained below: vertices, comprises two binded lists: First one which contains an increasing sequence of numbers starting from 1 to the number of vertices. Each of them corresponds to a landmark point. The second one contains the number of points covered by a ball of radius epsilon centered by the following landmark points. edges, a collection of not directed edges composed of the first and the second vertex. Ordering of vertices do not have meaning. edges_strength, For every edge [a,b] we define its strength as the number of points that are covered by both landmarks a and b. This array contains the strength of every edge in the Ball Mapper graph. points_covered_by_landmarks, is a list of vectors. I-th vector contains the positions of points covered by i-th landmark. landmarks, contains a list of positions of the landmark points used to construct the balls. coloring, is a vector having as many positions as the number of landmarks. It contains the averaged outcome values of the coloring variable corresponding to the points covered by each landmark.

Examples

```r
var <- seq(from=0,to=6.3,by=0.1)
points <- as.data.frame( cbind( sin(var),cos(var) ) )
values <- as.data.frame( sin(var) )
epsilon <- 0.25
l <- BallMapper(points,values,epsilon)

l <- BallMapper(points,values,epsilon)
```

Description

Produce a collection of png files with mapper graphs colored by following coordinates (so that the number of files is the same as the number of coordinates).

Usage

```r
colorByAllVariables(outputFromBallMapper, points, 
    fileNamePrefix = "output_", defaultXResolution = 512, 
    defaultYResolution = 512)
```

Arguments

- `outputFromBallMapper`: an output from the BallMapper function
- `points`: a collection of input points in a form of a data frame used to create Ball Mapper graph.
- `fileNamePrefix`: a prefix of a file name. A plot that uses i-th variable as a coloring will contain this string as a prefix followed by the number i. Set to "output_" by default.
defaultXResolution
store a default resolution of image in x direction. Set to 512 by default.

defaultYResolution
store a default resolution of image in y direction. Set to 512 by default.

Value

none.

---

colorByAverageValueOfOtherVariable

*Produce a new coloring vector being an average of values of given function at points covered by each vertex of Ball Mapper graph.*

Description

Produce a new coloring vector being an average of values of given function at points covered by each vertex of Ball Mapper graph.

Usage

colorByAverageValueOfOtherVariable(outputFromBallMapper, 
newFunctionOnPoints)

Arguments

outputFromBallMapper
an output from the BallMapper function

ewFunctionOnPoints
values of function on points.

Value

Vector of function values on vertices on Ball Mapper graph. 

```
var <- seq(from=0,to=6.3,by=0.1)
points <- as.data.frame( cbind( sin(var),cos(var) ) )
values <- as.data.frame(sin(var))
l <- BallMapper(points, values, 0.25)
ColorIgraphPlot(l) 
new_coloring <- colorByAverageValueOfOtherVariable(l,cos(var))
l$coloring <- new_coloring
ColorIgraphPlot(l)
```
colorByStDevValueOfOtherVariable

Description

Produce a new coloring vector being a standard deviation of values of given function at points covered by each vertex of Ball Mapper graph.

Usage

colorByStDevValueOfOtherVariable(outputFromBallMapper, newFunctionOnPoints)

Arguments

outputFromBallMapper
  an output from the BallMapper function
newFunctionOnPoints
  values of function on points.

Value

Vector of function values on vertices on Ball Mapper graph.

```r
var <- seq(from=0,to=6.3,by=0.1)
points <- as.data.frame( cbind( sin(var),cos(var) ) )
values <- as.data.frame(sin(var))
l <- BallMapper(points, values, 0.25)
ColorIgraphPlot(l)
new_coloring <- colorByStDevValueOfOtherVariable(l,sin(var))
l$coloring <- new_coloring
ColorIgraphPlot(l)
```

coloredDynamicNetwork

Description

This procedure produces a dynamic graph with colors. It allows zoom-in operation and displays information about vertices when they are clicked upon.

Usage

coloredDynamicNetwork(outputOfBallMapper, showLegend = FALSE)
Arguments

- `outputOfBallMapper`, an output from the BallMapper function.
- `showLegend`, if set to TRUE a legend will be displayed indicating the coloring of the values of vertices.

Value

None

Examples

```r
var <- seq(from=0,to=6.3,by=0.1)
points <- as.data.frame( cbind( sin(var),cos(var) ) )
values <- as.data.frame( sin(var) )
epsilon <- 0.25
l <- BallMapper(points,values,epsilon)
coloredDynamicNetwork(l)
```

Description

Produce a static color visualization of the Ball Mapper graph. It is based on the output from BallMapper function.

Usage

```r
ColorIgraphPlot(outputFromBallMapper, showVertexLabels = TRUE,
                  showLegend = FALSE, minimal_ball_radius = 7,
                  maximal_ball_scale = 20, maximal_color_scale = 10,
                  seed_for_plotting = -1, store_in_file = "",
                  default_x_image_resolution = 512, default_y_image_resolution = 512,
                  number_of_colors = 100)
```

Arguments

- `outputFromBallMapper`, an output from the BallMapper function.
- `showVertexLabels`, a boolean value determining if the vertex labels are to be shown (TRUE by default).
- `showLegend`, a boolean value determining if the legend is to be shown (FALSE by default).
- `minimal_ball_radius`, provide a minimal value of the radius of balls used in visualization (7 by default).
maximal_ball_scale,
provide a maximal value of the radius of balls used in visualization (20 by default).
maximal_color_scale,
Provide a maximal value (starting from 0) of the color of a ball (10 by default).
seed_for_plotting,
if set to the same number will suspend the fandom argument in the plotting routine and produce plots with the same layout everytime.
store_in_file
if set to a string, will open a png file, and store the plot therein. By default it is set to an empty string.
default_x_image_resolution
store a default resolution of image in x direction. Set to 512 by default.
default_y_image_resolution
store a default resolution of image in y direction. Set to 512 by default.
number_of_colors
store a number of colors used in the plot.

Value
None.

Examples

```r
var <- seq(from=0, to=6.3, by=0.1)
points <- as.data.frame( cbind( sin(var),cos(var) ) )
values <- as.data.frame( sin(var) )
epsilon <- 0.25
l <- BallMapper(points,values,epsilon)
ColorIgraphPlot(l)
```

---

color_by_distance_to_reference_points

This function will provide a new coloring which is the minimal and average distance of points in the point cloud to the reference points. The output from this procedure can be used as an alternative coloring in BallMapper.

Description

This function will provide a new coloring which is the minimal and average distance of points in the point cloud to the reference points. The output from this procedure can be used as an alternative coloring in BallMapper.

Usage

`color_by_distance_to_reference_points(allPoints, refPoints)`
coordinates_of_points_in_subcollection

Arguments

allPoints is a collection of all points in the dataset.
refPoints is a subset of all points. The function will compute the distance of each point from allPoints to referencePoints

Value

a pair of minimal and average distances. They can be used to color the BallMapper graph.

```r
define_function <- function(points, values, l)
  l <- BallMapper(points, values, l)
  pts <- as.data.frame(points_covered_by_landmarks(l, 1))
  new_coloring_function <- color_by_distance_to_reference_points(points, pts, l)
  l$coloring <- new_coloring_function[, 1]
  ColorIgraphPlot(l)
  l$coloring <- new_coloring_function[, 2]
  ColorIgraphPlot(l)
```

coordinates_of_points_in_subcollection

This is an auxiliary function. It takes the coordinates of points, ids of subset of points, and number of coordinate, and return a sorted vector of the given coordinate in the considered points. For instance, given the collection of points: 1 2 3 4 5 6 7 8 9 and which_subset = 2,3 and number_of_coordinate = 2 the procedure below will return the vector [2,5,8].

Description

This is an auxiliary function. It takes the coordinates of points, ids of subset of points, and number of coordinate, and return a sorted vector of the given coordinate in the considered points. For instance, given the collection of points: 1 2 3 4 5 6 7 8 9 and which_subset = 2,3 and number_of_coordinate = 2 the procedure below will return the vector [2,5,8].

Usage

coordinates_of_points_in_subcollection(points, which_subset, number_of_coordinate)

Arguments

points is a collection of input points in a form of a data frame. The same one as on the input of the Ball Mapper.
which_subset Indices of points in the given subset.
number_of_coordinate which coordinate of the considered points to export.

Value

the sorted vector of values of a given variable at the collection of points.

```r
var <- seq(from=0, to=6.3, by=0.1)
points <- as.data.frame(cbind(sin(var), cos(var)))
values <- as.data.frame(sin(var))
l <- BallMapper(points, values, 0.25)
coordinates_of_points_in_subcollection(points, c(6, 7, 8), 1)
```
find_dominant_difference_using_averages

This procedure takes two subsets of points (that come from the vertices of Ball Mapper) and returns the coordinates on which the averages of those two collections differ most. To balance the effect of potentially different orders of magnitude of data in column, we divide the difference in means by the mean of the whole column.

Description

This procedure takes two subsets of points (that come from the vertices of Ball Mapper) and returns the coordinates on which the averages of those two collections differ most. To balance the effect of potentially different orders of magnitude of data in column, we divide the difference in means by the mean of the whole column.

Usage

find_dominant_difference_using_averages(points, subset1, subset2)

Arguments

points a collection of input points in a form of a data frame. The same one as on the input of the Ball Mapper.
subset1 First subset of ids of points.
subset2 Second subset of ids of points.

Value

Vector of coordinate ids with the absolute value of difference between averages, ordered according to the second variable.

```r
var <- seq(from=0, to=6.3, by=0.1) points <- as.data.frame(cbind(sin(var), cos(var))) values <- as.data.frame(sin(var)) l <- BallMapper(points, values, 0.25) g1 <- c(1, 21) g2 <- c(11, 12) find_dominant_difference_using_averages(points, g1, g2)
```

find_dominant_difference_using_averages_normalized_by_sd

This procedure takes two subsets of points (that come from the vertices of Ball Mapper) and returns the coordinates on which the averages of those two collections differ most. To balance the effect of potentially different orders of magnitude of data in column, we divide the difference in means by the standard deviation of the whole column.

```r
```
**Description**

This procedure takes two subsets of points (that come from the vertices of Ball Mapper) and returns the coordinates on which the averages of those two collections differ most. To balance the effect of potentially different orders of magnitude of data in column, we divide the difference in means by the standard deviation of the whole column.

**Usage**

```r
find_dominant_difference_using_averages_normalized_by_sd(points, subset1, subset2)
```

**Arguments**

- **points**: a collection of input points in the form of a data frame. The same one as on the input of the Ball Mapper.
- **subset1**: First subset of ids of points.
- **subset2**: Second subset of ids of points.

**Value**

A vector of coordinate ids with the absolute value of difference between averages normalized by the standard deviation of the considered column, ordered according to the second variable.

```r
var <- seq(from=0, to=6.3, by=0.1) points <- as.data.frame(cbind(sin(var), cos(var))) values <- as.data.frame(sin(var)) l <- BallMapper(points, values, 0.25) g1 <- c(1, 21) g2 <- c(11, 12) find_dominant_difference_using_averages(points, g1, g2)
```

---

**GrayscaleIgraphPlot**

*Produce a static grayscale visualization of the Ball Mapper graph. It is based on the output from the BallMapper function.*

**Description**

Produce a static grayscale visualization of the Ball Mapper graph. It is based on the output from the BallMapper function.

**Usage**

```r
GrayscaleIgraphPlot(outputFromBallMapper, showVertexLabels = TRUE, minimal_ball_radius = 7, maximal_ball_scale = 20, seed_for_plotting = -1, store_in_file = "", default_x_image_resolution = 512, default_y_image_resolution = 512)
```
Arguments

outputFromBallMapper,  
an output from the BallMapper function
showVertexLabels,  
a boolean value determining if vertex labels are to be shown (TRUE by default).
minimal_ball_radius,  
provide a minimal value of the radius of balls used in visualization (7 by default).
maximal_ball_scale,  
provides a maximal value of the radius of the balls used in visualization (20 by default).
seed_for_plotting,  
if set to the same number will suspend the fandom argument in the plotting routine and produce plots with the same layout everytime.
store_in_file  
if set to a string, will open a png file, and store the plot therein. By default it is set to an empty string.
default_x_image_resolution  
store a default resolution of image in x direction. Set to 512 by default.
default_y_image_resolution  
store a default resolution of image in y direction. Set to 512 by default.

Value

None.

Examples

```
var <- seq(from=0,to=6.3,by=0.1)
points <- as.data.frame( cbind( sin(var),cos(var) ) )
values <- as.data.frame( sin(var) )
epsilon <- 0.25
l <- BallMapper(points,values,epsilon)
GrayscaleIgraphPlot(l)
```

normalize_to_average_0_stdev_1

This function normalize each column (variable) of the input dataset so that the the average of the normalized column is 0 and its standard deviation is 1.

Description

This function normalize each column (variable) of the input dataset so that the the average of the normalized column is 0 and its standard deviation is 1.

Usage

normalize_to_average_0_stdev_1(points)
normalize_to_min_0_max_1

Arguments

points, a collection of input points in a form of a data frame.

Value

Normalized collection of points.

Examples

```r
var <- seq(from=0,to=6.3,by=0.1)
points <- as.data.frame( cbind( sin(var),cos(var) ) )
normalized_points <- normalize_to_average_0_stdev_1 (points)
```

Description

This function normalize each column (variable) of the input dataset so that the maximum is mapped to one, minimum to zero, and the intermediate values linearly to the appropriate points in the interval (0,1).

Usage

`normalize_to_min_0_max_1(points)`

Arguments

points, a collection of input points in a form of a data frame.

Value

Normalized collection of points.

Examples

```r
var <- seq(from=0,to=6.3,by=0.1)
points <- as.data.frame( cbind( sin(var),cos(var) ) )
normalized_points <- normalize_to_min_0_max_1 (points)
```
points_covered_by_landmarks

This function returns a list of points covered by the given collection of landmarks.

**Description**

This function returns a list of points covered by the given collection of landmarks.

**Usage**

```r
points_covered_by_landmarks(outputFromBallMapper, numbers_of_landmarks)
```

**Arguments**

- `outputFromBallMapper`:
  - an output from the BallMapper function
- `numbers_of_landmarks`:
  - a vector containing the numbers of landmarks under consideration.

**Value**

A vector of points covered by the landmarks given in `numbers_of_landmarks`.

---

pointToBallList

Produce a two column list. The first column contain the number of point (possibly with repetitions), the second one contains the number of landmark points that cover it. For example, let us assume that point 1 is covered by landmark 1 and 2, and point 2 is covered by the landmark 2. In this case the obtained list is of a form: 1 1 1 2 2 2 This list can be used for a further analysis of various parts of Ball Mapper graph.

**Description**

Produce a two column list. The first column contain the number of point (possibly with repetitions), the second one contains the number of landmark points that cover it. For example, let us assume that point 1 is covered by landmark 1 and 2, and point 2 is covered by the landmark 2. In this case the obtained list is of a form: 1 1 1 2 2 2 This list can be used for a further analysis of various parts of Ball Mapper graph.

**Usage**

```r
pointToBallList(coverageFromBallMapper)
```
**Arguments**

coverageFromBallMapper,

a coverage parameter of an output from BallMapper function

**Value**

List of landmarks covering each point, as described above.

**Examples**

```r
dat <- seq(from=0, to=6.3, by=0.1)
points <- as.data.frame(cbind(sin(var), cos(var)))
values <- as.data.frame(sin(var))
epsilon <- 0.25
l <- BallMapper(points, values, epsilon)
list <- pointToBallList(l coverage)
```

---

**readBallMapperGraphFromFile**

This procedure reads the BallMapper object from file. The parameter of the file is filename. We assume that files: filename_vertices filename_edges filename_edges_strength filename_points_covered_by_landmarks filename_landmarks filename_coloring

**Description**

This procedure reads the BallMapper object from file. The parameter of the file is filename. We assume that files: filename_vertices filename_edges filename_edges_strength filename_points_covered_by_landmarks filename_landmarks filename_coloring

**Usage**

```r
readBallMapperGraphFromFile(filename)
```

**Arguments**

filename

prefix of the name of the file containing elements of Ball Mapper graph.

**Value**

BallMapper object var <- seq(from=0, to=6.3, by=0.1) points <- as.data.frame(cbind(sin(var), cos(var))) values <- as.data.frame(sin(var)) l <- BallMapper(points, values, 0.25) storeBallMapperGraphInFile(l, "my_favorite_BM_graph") l_prime <- readBallMapperGraphFromFile("my_favorite_BM_graph")
simpleDynamicNetwork

This is a simple example of dynamic visualization using networkD3 library. This version do not implement coloring of vertices, just give a general overview of the edges.

Description

This is a simple example of dynamic visualization using networkD3 library. This version do not implement coloring of vertices, just give a general overview of the edges.

Usage

simpleDynamicNetwork(outputFromBallMapper, storeAsHtml = FALSE)

Arguments

outputFromBallMapper,
    an output from BallMapper function.
storeAsHtml,    if set true, it will store the graph in HTML file.

Value

None

Examples

var <- seq(from=0,to=6.3,by=0.1)
points <- as.data.frame( cbind( sin(var),cos(var) ) )
values <- as.data.frame( sin(var) )
epsilon <- 0.25
l <- BallMapper(points,values,epsilon)
simpleDynamicNetwork(l)

storeBallMapperGraphInFile

This procedure store the Ball Mapper graph in a file in the following format:

Description

This procedure store the Ball Mapper graph in a file in the following format:

Usage

storeBallMapperGraphInFile(outputFromBallMapper, filename = "BM_graph")
Arguments

outputFromBallMapper
output from the BallMapper procedure.

filename the name of the file to store the data.

Value

None

```r
var <- seq(from=0, to=6.3, by=0.1)
points <- as.data.frame(cbind(sin(var), cos(var)))
values <- as.data.frame(sin(var))
l <- BallMapper(points, values, 0.25)
storeBallMapperGraphInFile(l, "my_favorite_BM_graph")
```
Index

BallMapper, 2

color_by_distance_to_reference_points, 7

colorByAllVariables, 3
colorByAverageValueOfOtherVariable, 4
colorByStDevValueOfOtherVariable, 5
coloredDynamicNetwork, 5
ColorIgraphPlot, 6
coordinates_of_points_in_subcollection, 8

find_dominant_difference_using_averages, 9
find_dominant_difference_using_averages_normalized_by_sd, 9

GrayscaleIgraphPlot, 10

normalize_to_average_0_stdev_1, 11
normalize_to_min_0_max_1, 12

points_covered_by_landmarks, 13
pointToBallList, 13

readBallMapperGraphFromFile, 14

simpleDynamicNetwork, 15
storeBallMapperGraphInFile, 15