Package ‘GeneralizedUmatrix’

Type Package

Title Credible Visualization for Two-Dimensional Projections of Data

Version 1.1.8

Date 2020-02-16

Maintainer Michael Thrun <m.thrun@gmx.net>

Description Projections are common dimensionality reduction methods, which represent high-dimensionality data in a two-dimensional space. However, when restricting the output space to two dimensions, which results in a two dimensional scatter plot (projection) of the data, low dimensional similarities do not represent high dimensional distances coercively [Thrun, 2018]. This could lead to a misleading interpretation of the underlying structures [Thrun, 2018]. By means of the 3D topographic map the generalized Umatrix is able to depict errors of these two-dimensional scatter plots. The package is based on the book of Thrun, M.C.: "Projection Based Clustering through Self-Organization and Swarm Intelligence" (2018) <DOI:10.1007/978-3-658-20540-9>.

License GPL-3

Imports Rcpp, ggplot2

Suggests DataVisualizations, DatabionicSwarm, matrixStats, rgl, grid, mgcv, png, ProjectionBasedClustering, reshape2, fields, ABCAnalysis, plotly, deldir, shiny, knitr (>= 1.12), rmarkdown (>= 0.9)

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 3.0)

NeedsCompilation yes

SystemRequirements C++11

LazyLoad yes

LazyData TRUE

URL http://www.deepbionics.org

Encoding UTF-8

VignetteBuilder knitr

BugReports https://github.com/Mthrun/GeneralizedUmatrix/issues
Description

Projections are common dimensionality reduction methods, which represent high-dimensional data in a two-dimensional space. However, when restricting the output space to two dimensions, which results in a two dimensional scatter plot (projection) of the data, low dimensional similarities do not represent high dimensional distances coercively [Thrun, 2018]. This could lead to a misleading interpretation of the underlying structures [Thrun, 2018]. By means of the 3D topographic map the generalized Umatrix is able to depict errors of these two-dimensional scatter plots. The package is based on the book of Thrun, M.C.: "Projection Based Clustering through Self-Organization and Swarm Intelligence" (2018) <DOI:10.1007/978-3-658-20540-9>.
Details

For a brief introduction to **GeneralizedUmatrix** please see the vignette **Introduction of the Generalized Umatrix Package**.

For further details regarding the generalized Umatrix see [Thrun, 2018], chapter 4-5.

If you want to verify your clustering result externally, you can use **Heatmap** or **SilhouettePlot** of the CRAN package **DataVisualizations**.

Index of help topics:

- **CalcUstarmatrix**: Calculate the U*matrix for a given Umatrix and Pmatrix.
- **Chainlink**: Chainlink is part of the Fundamental Clustering Problem Suit (FCPS) [Ultsch, Chainlink005].
- **DefaultColorSequence**: Default color sequence for plots
- **Delta3DWeightsC**: intern function
- **GeneralizedUmatrix**: Generalized U-Matrix for Projection Methods
- **GeneralizedUmatrix-package**: Credible Visualization for Two-Dimensional Projections of Data
- **GeneratePmatrix**: Generates the P-matrix
- **NormalizeUmatrix**: Normalize Umatrix
- **TopviewTopographicMap**: Topview of Topographic Map ind 2D
- **UmatrixColorMap**: U-Matrix colors
- **XYcoords2LinesColumns**: XYcoords2LinesColumns(X,Y) Converts points given as x(i),y(i) coordinates to integer coordinates Columns(i),Lines(i)
- **plotTopographicMap**: Visualizes th Generalized U-matrix in 3D
- **sESOM4BMUs**: simplified ESOM
- **trainstepC**: internal function for s-esom
- **upscaleUmatrix**: Upscale a Umatrix grid

Author(s)

Michal Thrun

Maintainer: Michael Thrun <mthrun@informatik.uni-marburg.de>

References


Examples

data("Chainlink")
Data=Chainlink$Data
Cls=Chainlink$Cls
InputDistances=as.matrix(dist(Data))
res=cmdscale(d=InputDistances, k = 2, eig = TRUE, add = FALSE, x.ret = FALSE)
ProjectedPoints=as.matrix(res$points)
#see also ProjectionBasedClustering package for other common projection methods
#see DatabionicSwarm for projection method without parameters or objective function
 ProjectedPoints=DatabionicSwarm::Pswarm(Data)$ProjectedPoints
resUmatrix=GeneralizedUmatrix(Data,ProjectedPoints)
plotTopographicMap(resUmatrix$Umatrix,resUmatrix$Bestmatches,Cls)

##Interactive Island Generation
## from a tiled Umatrix (toroidal assumption)
## Not run:
Imx = ProjectionBasedClustering::interactiveGeneralizedUmatrixIsland(resUmatrix$Umatrix, resUmatrix$Bestmatches)
plotTopographicMap(resUmatrix$Umatrix,
resUmatrix$Bestmatches, Imx = Imx)

## End(Not run)

##External Verification
## Not run:

DataVisualizations::Heatmap(Data,Cls)
# if spherical cluster structure
DataVisualizations::SilhouettePlot(Data,Cls)

## End(Not run)

---

**CalcUstarmatrix**  
*Calculate the U*matrix for a given Umatrix and Pmatrix.*

**Description**
Calculate the U*matrix for a given Umatrix and Pmatrix.

**Arguments**
- **Umatrix[1:Lines,1:Column]**
  - Local averages of distances at each point of the trainedGridWts[1:Lines,1:Column,1:variables] of ESOM or other SOM of same format
- **Pmatrix[1:Lines,1:Column]**
  - Local densities at each point of the trainedGridWts[1:Lines,1:Column,1:variables] of ESOM or other SOM of same format

**Value**
- **UStarMatrix[1:Lines,1:Column]**
**Chainlink**

**Author(s)**

Michael Thrun

**References**


---

**Chainlink** is part of the Fundamental Clustering Problem Suit (FCPS) [Ultsch, Chainlink005].

**Description**

linear not separable dataset of two interwined chains.

**Usage**

data("Chainlink")

**Details**

Size 1000, Dimensions 3, stored in Chainlink$Data
Teo clusters, stored in Chainlink$Cls
Published in [Ultsch et al.1994] in German and [Ultsch 1995] in English.

**References**


**Examples**

data(Chainlink)
str(Chainlink)

library(DataVisualizations)
DataVisualizations::plot3D(Chainlink$Data,Chainlink$Cls)
**DefaultColorSequence**  
*Default color sequence for plots*

**Description**

Defines the default color sequence for plots made within the Projections package.

**Usage**

```r
data("DefaultColorSequence")
```

**Format**

A vector with 562 different strings describing colors for plots.

---

**Delta3DWeightsC**  
*intern function*

**Description**

Thr implementation of the main formula of SOM, ESOM, sESOM algorithms.

**Usage**

```r
Delta3DWeightsC(vx,Datasample)
```

**Arguments**

- **vx**: array of weights [1:Lines,1:Columns,1:Weights]
- **Datasample**: NumericVector of one Datapoint[1:n]

**Details**

Intern function in case of ComputeInR==FALSE in GeneralizedUmatrix

**Value**

Modified array of weights [1:Lines,1:Columns,1:Weights]

**Author(s)**

Michael Thrun

**References**

Generalized U-matrix

Description

Generalized U-Matrix visualizes high-dimensional distance and density based structures in two-dimensional scatter plots of projection methods like CCA, MDS, PCA or NeRV with the help of a topographic map with hypsometric tints [Thrun et al. 2016] based on the Umatrix method for emergent SOMs [Ultsch 2003], for further explanation see [Thrun, 2018].

Usage

GeneralizedUmatrix(Data, ProjectedPoints,
PlotIt=FALSE,Cls=NULL,Toroid=TRUE,Tiled=FALSE,ComputeInR=FALSE)

Arguments

Data
[1:n,1:d] array of data: n cases in rows, d variables in columns

ProjectedPoints
[1:n,2] matrix containing coordinates of the Projection: A matrix of the fitted configuration.

PlotIt
Optional, bool, default=FALSE, if =TRUE: U-Matrix of every current Position of Databots will be shown However, the amount of details shown will be less than in plotTopographicMap.

Cls
Optional, For plotting, see plotUmatrix in package Umatrix

Toroid
Optional, Default=FALSE,
==FALSE planar computation
==TRUE: toroid borderless computation, set so only if projection method is also toroidal

Tiled
Optional,For plotting see plotUmatrix in package Umatrix

ComputeInR
Optional, =T: Rcode, =F Cpp Code

Details

Introduced first in [Thrun, 2018, p.46], additionaly reviewed in [Ultsch/Thrun, 2017].

Value

List with

Umatrix
[1:Lines,1:Columns] [1:Lines,1:Columns] Umatrix to be plotted, numerical matrix storing the U-heights, see [Thrun, 2018] for definition.

EsomNeurons
[1:Lines,1:Columns,1:weights] 3-dimensional numeric array (wide format), not wts (long format)
GeneralizedUmatrix

Bestmatches [1:n,OutputDimension] Grid Converted Projected Points information converted by convertProjectionProjectedPoints() to predefined Grid by Lines and Columns

gplotres Ausgabe von ggplot

Author(s)

Michael Thrun

References


Examples

data("Chainlink")
Data=Chainlink$Data
Cls=Chainlink$Cls
InputDistances=as.matrix(dist(Data))
res=cmdscale(d=InputDistances, k = 2, eig = TRUE, add = FALSE, x.ret = FALSE)
ProjectedPoints=as.matrix(res$points)
## Not run:
Stress = ProjectionBasedClustering::KruskalStress(InputDistances,
as.matrix(dist(ProjectedPoints)))
## End(Not run)

resUmatrix=GeneralizedUmatrix(Data,ProjectedPoints)
plotTopographicMap(resUmatrix$Umatrix,resUmatrix$Bestmatches,Cls)
GeneratePmatrix  Generates the P-matrix

Description

Generates a P-matrix too visualize only density based structures of high-dimensional data.

Arguments

- **Data**  
  [1:n,1:d], A [n, d] matrix containing the data
- **EsomNeurons**  
  [1:Lines,Columns,1:Weights] 3D array of weights given by ESOM or sESOM algorithm.
- **Radius**  
  The radius for measuring the density within the hypersphere.
- **PlotIt**  
  If set the Pmatrix will also be plotted
  ...

Details

To set the Radius the ABCanalysis of high-dimensional distances can be used [Ultsch/Lötsch, 2015]. For a detailed definition and equation of automated density estimation (Radius) see Thrun et al. 2016.

Value

PMatrix[1:Lines,1:Columns]

Author(s)

Michael Thrun

References


NormalizeUmatrix

Description
Normalizing the U-matrix using the abstact U-Matrix concept [Loetsch/Ultsch, 2014].

Usage
NormalizeUmatrix(Data, Umatrix, BestMatches)

Arguments
- Data [1:n,1:d] numerical matrix of data with n cases and d variables
- Umatrix [1:lines,1:Columns] matrix of U-heights

Details
see publication [Loetsch/Ultsch, 2014].

Value

Note
Development Ongoing.

Author(s)
Felix Pape, Michael Thrun

References

Examples
#ToDo
plotTopographicMap  Visualizes the Generalized U-matrix in 3D

Description

Visualizes high-dimensional distance and density based structures of the combination two-dimensional scatter plots (projections) with high-dimensional data as the topographic map with hypsometric tints which is a 3D landscape.

Usage

plotTopographicMap(GeneralizedUmatrix, BestMatchingUnits, 
Cls=NULL,ClsColors=NULL,Imx=NULL,Names=NULL, BmSize=0.5,...)

Arguments

GeneralizedUmatrix
(1:Lines,1:Columns), [1:Lines,1:Columns] Umatrix to be plotted, numerical matrix storing the U-heights, see [Thrun, 2018] for definition.

BestMatchingUnits
(1:n,1:2), Positions of bestmatches to be plotted onto the Umatrix

Cls
(1:n), numerical vector of classification of k classes for the bestmatch at the given point

ClsColors
Vector of colors that will be used to colorize the different classes

Imx
a mask (Imx) that will be used to cut out the umatrix

Names
If set: [1:k] character vector naming the k classes for the legend. . In this case, further parameters with the possibility to adjust are: NamesCex: (size); NamesPosition: Legend position; NamesTitle: title of legend; NamesColors: colors if ClsColors are not default (NULL).

BmSize
size(diameter) of the points in the visualizations. The points represent the Best-MatchingUnits

... Besides the legend/names parameter the list of further parameters, use only of you know what you are doing:

Tiled Should the Umatrix be drawn 4times?
ShowAxis shall the axis be shown?
NoLevels number of contour lines
Colormap in the case of density p matrix...

title same as main
main same as title
sub same as in plot
xlab same as in plot
ylab same as in plot
zlab same as in plot
Details

The visualization and result of this function is a topographic map with hypsometric tints (Thrun, Lerch, L?tsch, & Ultsch, 2016). Hypsometric tints are surface colors that represent ranges of elevation (see (Thrun et al., 2016)). Here, contour lines are combined with a specific color scale. The color scale is chosen to display various valleys, ridges, and basins: blue colors indicate small distances (sea level), green and brown colors indicate middle distances (low hills), and shades of white colors indicate vast distances (high mountains covered with snow and ice). Valleys and basins represent clusters, and the watersheds of hills and mountains represent the borders between clusters. In this 3D landscape, the borders of the visualization are cyclically connected with a periodicity (L,C). A central problem in clustering is the correct estimation of the number of clusters. This is addressed by the topographic map which allows assessing the number of clusters (Thrun et al., 2016). Please see chapter 5 of [Thrun, 2018] for further details.

Note

Algorithm is partly based on the Umatrix package.

Author(s)

Michael Thrun

References


See Also

GeneralizedUmatrix

Examples

data("Chainlink")
Data=Chainlink$Data
Cls=Chainlink$Cls
InputDistances=as.matrix(dist(Data))
res=cmdscale(d=InputDistances, k = 2, eig = TRUE, add = FALSE, x.ret = FALSE)
ProjectedPoints=as.matrix(res$points)
#see also ProjectionBasedClustering package for other common projection methods

resUmatrix=GeneralizedUmatrix(Data,ProjectedPoints)
## visualization
plotTopographicMap(GeneralizedUmatrix = resUmatrix$Umatrix,resUmatrix$Bestmatches)
## To save as STL for 3D printing
rgl::writeSTL("GeneralizedUmatrix_3d_model.stl")

## Save the visualization as a picture with
library(rgl)
rgl.snapshot('test.png')

---

**sESOM4BMUs**

*simplied ESOM*

---

**Description**

Internfunction for the simplified ESOM Algorithmus of [Thrun, 2018] for fixed BestMatchingUnits

**Usage**

`sESOM4BMUs(BMUs, Data, esom, toroid, CurrentRadius, ComputeInR)`

**Arguments**

- **BMUs**: [1:Lines,1:Columns], BestMatchingUnits generated by `ProjectedPoints2Grid()`
- **Data**: [1:n,1:d] array of data: n cases in rows, d variables in columns
- **esom**: [1:Lines,1:Columns,1:weights] array of NeuronWeights, see `ListAsEsomNeurons()`
- **toroid**: TRUE/FALSE - topology of points
- **CurrentRadius**: number between 1 to x
- **ComputeInR**: =T: Rcode, =F Cpp Codenumnber between 1 to x

**Details**

Algorithm is described in [Thrun, 2018, p. 48, Listing 5.1].

**Value**

- **esom**: array [1:Lines,1:Columns,1:d], d is the dimension of the weights, the same as in the ESOM algorithm. modified esomneuros regarding a predefined neighborhood defined by a radius

**Note**

Usually not for separated usage!

**Author(s)**

Michael Thrun
References


See Also

GeneralizedUmatrix

---

**TopviewTopographicMap**  
*Topview of Topographic Map ind 2D*

**Description**

Fast Visualization of the Generalized U-matrix in 2D which visualizes high-dimensional distance and density based structures of the combination two-dimensional scatter plots (projections) of high-dimensional data.

**Usage**

TopviewTopographicMap(GeneralizedUmatrix, BestMatchingUnits,  
Cls, ClsColors = NULL, Imx = NULL, Names = NULL, BmSize = 12, ...)

**Arguments**

- **GeneralizedUmatrix**
  
  (1:Lines,1:Columns), [1:Lines,1:Columns] Umatrix to be plotted, numerical matrix storing the U-heights, see [Thrun, 2018] for definition.

- **BestMatchingUnits**
  
  (1:n,1:2), Positions of bestmatches to be plotted onto the Umatrix

- **Cls**
  
  (1:n), numerical vector of classification of k classes for the bestmatch at the given point

- **ClsColors**
  
  Vector of colors that will be used to colorize the different classes

- **Imx**
  
  a mask (Imx) that will be used to cut out the umatrix

- **Names**
  
  If set: [1:k] character vector naming the k classes for the legend. In this case, further parameters with the possibility to adjust are: NamesCex: (size); NamesPosition: Legend position; NamesTitle: title of legend; NamesColors: colors if ClsColors are not default (NULL).

- **BmSize**
  
  size(diameter) of the points in the visualizations. The points represent the Best-MatchingUnits

...  
- **Tiled** Should the Umatrix be drawn 4times?
- **Further Arguments relevant for interactive shiny application**
Details

Please see `plotTopographicMap`.

Value

plotly handler

Note

Names and Imx are currently under development

Author(s)

Tim Schreier, Luis Winckelmann, Michael Thrun

References


See Also

`plotTopographicMap`

Examples

data("Chainlink")
Data=Chainlink$Data
Cls=Chainlink$Cls
InputDistances=as.matrix(dist(Data))
res=cmdscale(d=InputDistances, k = 2, eig = TRUE, add = FALSE, x.ret = FALSE)
ProjectedPoints=as.matrix(res$points)
#see also ProjectionBasedClustering package for other common projection methods

resUmatrix=GeneralizedUmatrix(Data,ProjectedPoints)
## visualization
TopviewTopographicMap(GeneralizedUmatrix = resUmatrix$Umatrix,resUmatrix$Bestmatches)
**trainstepC**

*internal function for s-esom*

**Description**

Does the training for fixed bestmatches in one epoch of the sESOM.

**Usage**

```
trainstepC(vx, vy, DataSampled, BMUsampled, Lines, Columns, Radius, toroid)
```

**Arguments**

- **vx**: array (1:Lines,1:Columns,1:Weights), WeightVectors that will be trained, internally transformed von NumericVector to cube
- **vy**: array (1:Lines,1:Columns,1:2), meshgrid for output distance computation
- **DataSampled**: NumericMatrix, n cases shuffled Dataset[1:n,1:d] by sample
- **BMUsampled**: NumericMatrix, n cases shuffled BestMatches[1:n,1:2] by sample in the same way as DataSampled
- **Lines**: double, Height of the grid
- **Columns**: double, Width of the grid
- **Radius**: double, The current Radius that should be used to define neighbours to the bm
- **toroid**: bool, Should the grid be considered with cyclically connected borders?

**Details**

Algorithm is described in [Thrun, 2018, p. 48, Listing 5.1].

**Value**

- WeightVectors, array[1:Lines,1:Columns,1:weights] with the adjusted Weights

**Note**

Usually not for seperated usage!

**Author(s)**

Michael Thrun

**References**

**UmatrixColormap**  
*U-Matrix colors*

**Description**

Defines the default color sequence for plots made for Umatrix

**Usage**

```r
data("UmatrixColormap")
```

**Format**

Returns the vectors for a (heat) colormap.

---

**upscaleUmatrix**  
*Upscale a Umatrix grid*

**Description**

Use linear interpolation to increase the size of a umatrix. This can be used to produce nicer ggplot plots in `plotTopographicMap` and is going to be used for further normalization of the umatrix.

**Usage**

```r
upscaleUmatrix(Umatrix, Factor = 2, BestMatches, Imx)
```

**Arguments**

- `Umatrix`: The umatrix which should be upscaled
- `BestMatches`: The BestMatches which should be upscaled
- `Factor`: Optional: The factor by which the axes will be scaled. Be aware that the size of the matrix will grow by Factor squared. Default: 2
- `Imx`: Optional: Island cutout of the umatrix. Should also be scaled to the new size of the umatrix.

**Value**

A List consisting of:

- `Umatrix`: A matrix representing the upscaled umatrix.
- `BestMatches`: If BestMatches was given as parameter: The rescaled BestMatches for an island cutout. Otherwise: NULL
- `Imx`: If Imx was given as parameter: The rescaled matrix for an island cutout. Otherwise: NULL
Author(s)
Felix Pape

Description

XYcoords2LinesColumns(X,Y) Converts points given as x(i),y(i) coordinates to integer coordinates Columns(i),Lines(i)

Arguments

X(1:n), Y(1:n) coordinates: x(i),y(i) is the i-th point on a plane
minNeurons minimal size of the corresponding grid i.e max(Lines)*max(Columns) >= MinGridSize, default MinGridSize = 4096 defined by the number of neurons
MaxDifferentPoints TRUE: the discretization error is minimal FALSE: number of Lines and Columns is minimal
PlotIt Plots the result

Details

Details are written down in [Thrun, 2018, p. 47].

Value

GridConvertedPoints[1:Columns,1:Lines,2] IntegerPositions on a grid corresponding to x,y

Author(s)
Michael Thrun

References


Examples

data("Chainlink")
Data=Chainlink$Data
InputDistances=as.matrix(dist(Data))
res=cmdscale(d=InputDistances, k = 2, eig = TRUE, add = FALSE, x.ret = FALSE)
ProjectedPoints=as.matrix(res$points)
GridConvertedPoints=XYcoords2LinesColumns(ProjectedPoints[,1],ProjectedPoints[,2],PlotIt=FALSE)
Index

*Topic **AU-Matrix**
  NormalizeUmatrix, 10
*Topic **Chainlink**
  Chainlink, 5
*Topic **DR**
  GeneralizedUmatrix-package, 2
*Topic **DimensionalityReduction**
  GeneralizedUmatrix-package, 2
*Topic **ESOM**
  sESOM4BMUs, 13
*Topic **FCPS**
  Chainlink, 5
*Topic **GeneralizedUmatrix**
  GeneralizedUmatrix, 7
  GeneralizedUmatrix-package, 2
  plotTopographicMap, 11
  TopviewTopographicMap, 14
*Topic **GeneratePswarmVisualization**
  sESOM4BMUs, 13
*Topic **P-Matrix**
  GeneratePmatrix, 9
*Topic **P-matrix**
  GeneratePmatrix, 9
*Topic **Pmatrix**
  GeneratePmatrix, 9
*Topic **Projection**
  GeneralizedUmatrix-package, 2
*Topic **U*-Matrix**
  CalcUstarmatrix, 4
*Topic **U*-matrix**
  CalcUstarmatrix, 4
*Topic **Umatrix**
  CalcUstarmatrix, 4
*Topic **U-Matrix**
  GeneralizedUmatrix, 7
  GeneralizedUmatrix-package, 2
  plotTopographicMap, 11
  TopviewTopographicMap, 14
*Topic **U-matrix**
  GeneralizedUmatrix-package, 2
*Topic **Umatrix**
  GeneralizedUmatrix, 7
  GeneralizedUmatrix-package, 2
  plotTopographicMap, 11
  TopviewTopographicMap, 14
*Topic **Ustar-Matrix**
  CalcUstarmatrix, 4
*Topic **Ustar-matrix**
  CalcUstarmatrix, 4
*Topic **Ustarmatrix**
  CalcUstarmatrix, 4
*Topic **Visualization**
  GeneralizedUmatrix-package, 2
*Topic **abstract U-matrix**
  NormalizeUmatrix, 10
*Topic **datasets**
  Chainlink, 5
*Topic **map**
  GeneralizedUmatrix, 7
  plotTopographicMap, 11
  TopviewTopographicMap, 14
*Topic **topographic**
  GeneralizedUmatrix, 7
  plotTopographicMap, 11
  TopviewTopographicMap, 14
*Topic **visualization**
  GeneralizedUmatrix, 7
  plotTopographicMap, 11
  TopviewTopographicMap, 14
CalcUstarmatrix, 4
calcUstarmatrix (CalcUstarmatrix), 4
Chainlink, 5
DefaultColorSequence, 6
Delta3DWeightsC, 6
GeneralizedUmatrix, 6, 7, 12, 14
GeneralizedUmatrix-package, 2
GeneratePmatrix, 9
getUmatrix4Projection (GeneralizedUmatrix), 7
NormalizeUmatrix, 10
plot, 11
plotTopographicMap, 7, 11, 15, 17
sESOM4BMUs, 13
TopviewTopographicMap, 14
trainstepC, 16
UmatrixColormap, 17
upscaleUmatrix, 17
XYcoords2LinesColumns, 18