Package ‘GeneralizedUmatrix’

Type Package

Title Credible Visualization for Two-Dimensional Projections of Data

Version 1.2.4

Date 2022-05-25

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

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] <DOI:10.1007/978-3-658-20540-9>. 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 derived from the book of Thrun, M.C.: "Projection Based Clustering through Self-Organization and Swarm Intelligence" (2018) <DOI:10.1007/978-3-658-20540-9> and the main algorithm called simplified self-organizing map for dimensionality reduction methods is published in <DOI:10.1016/j.mex.2020.101093>.

License GPL-3

Imports Rcpp (>= 1.0.8), RcppParallel (>= 5.1.4), ggplot2

Suggests DataVisualizations, rgl, grid, mgcv, png, reshape2, fields, ABCAnalysis, plotly, deldir, methods, knitr (>= 1.12), markdown (>= 0.9)

LinkingTo Rcpp, RcppArmadillo, RcppParallel

Depends R (>= 3.0)

NeedsCompilation yes

SystemRequirements C++11, GNU make, pandoc (>=1.12.3, needed for vignettes)

LazyLoad yes

LazyData TRUE

URL https://www.deepbionics.org

Encoding UTF-8
VignetteBuilder knitr

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

Author Michael Thrun [aut, cre, cph] (<https://orcid.org/0000-0001-9542-5543>), Felix Pape [ctb, ctr], Tim Schreier [ctb, ctr], Luis Winckelman [ctb, ctr], Quirin Stier [ctb, ctr], Alfred Ultsch [ths]

Repository CRAN

Date/Publication 2022-05-25 15:30:08 UTC

## R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GeneralizedUmatrix-package</td>
<td>3</td>
</tr>
<tr>
<td>addRowWiseC</td>
<td>5</td>
</tr>
<tr>
<td>CalcUstarmatrix</td>
<td>6</td>
</tr>
<tr>
<td>Chainlink</td>
<td>6</td>
</tr>
<tr>
<td>DefaultColorSequence</td>
<td>7</td>
</tr>
<tr>
<td>Delta3DWeightsC</td>
<td>8</td>
</tr>
<tr>
<td>EsomNeuronsAsList</td>
<td>8</td>
</tr>
<tr>
<td>ExtendToroidalUmatrix</td>
<td>9</td>
</tr>
<tr>
<td>GeneralizedUmatrix</td>
<td>10</td>
</tr>
<tr>
<td>GeneratePmatrix</td>
<td>12</td>
</tr>
<tr>
<td>ListAsEsomNeurons</td>
<td>13</td>
</tr>
<tr>
<td>LowLand</td>
<td>14</td>
</tr>
<tr>
<td>NormalizeUmatrix</td>
<td>15</td>
</tr>
<tr>
<td>plotTopographicMap</td>
<td>16</td>
</tr>
<tr>
<td>ReduceToLowLand</td>
<td>19</td>
</tr>
<tr>
<td>sESOM4BMUs</td>
<td>20</td>
</tr>
<tr>
<td>setdiffMatrix</td>
<td>21</td>
</tr>
<tr>
<td>TopviewTopographicMap</td>
<td>21</td>
</tr>
<tr>
<td>trainstepC</td>
<td>23</td>
</tr>
<tr>
<td>Uheights4Data</td>
<td>24</td>
</tr>
<tr>
<td>UmatrixColormap</td>
<td>25</td>
</tr>
<tr>
<td>UniqueBestMatchingUnits</td>
<td>25</td>
</tr>
<tr>
<td>upscaleUmatrix</td>
<td>26</td>
</tr>
<tr>
<td>XYcoords2LinesColumns</td>
<td>27</td>
</tr>
</tbody>
</table>

Index 28
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] <DOI: 10.1007/978-3-658-20540-9>. 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 derived from the book of Thrun, M.C.: "Projection Based Clustering through Self-Organization and Swarm Intelligence" (2018) <DOI:10.1007/978-3-658-20540-9> and the main algorithm called simplified self-organizing map for dimensionality reduction methods is published in <DOI: 10.1016/j.mex.2020.101093>.

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, or [Thrun/Ultsch, 2020].

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) [Thrun/Ultsch, 2020].
- DefaultColorSequence Default color sequence for plots
- Delta3DWeightsC intern function
- EsomNeuronsAsList Converts wts data (EsomNeurons) into the list form
- ExtendToroidalUmatrix Extend Toroidal Umatrix
- GeneralizedUmatrix-package Credible Visualization for Two-Dimensional Projections of Data
- GeneratePmatrix Generates the P-matrix
- ListAsEsomNeurons Converts List to WTS
- LowLand LowLand
- NormalizeUmatrix Normalize Umatrix
- ReduceToLowLand ReduceToLowLand
GeneralizedUmatrix-package

TopviewTopographicMap  Topview of Topographic Map in 2D
Uheights4Data          Uheights4Data
UmatrixColormap        U-Matrix colors
UniqueBestMatchingUnits UniqueBestMatchingUnits
XYcoords2LinesColumns  XYcoords2LinesColumns(X,Y) Converts points
given as x(i), y(i) coordinates to integer
coordinates Columns(i), Lines(i)
addRowWiseC            intern function
plotTopographicMap     Visualizes the Generalized U-matrix in 3D
sESOM4BMUs             simplified ESOM
setdiffMatrix          setdiffMatrix shortens Matrix2Curt by those
                        rows that are in both matrices.
trainstepC             internal function for s-esom
upscaleUmatrix         Upscale a Umatrix grid

Author(s)
Michal Thrun
Maintainer: Michael Thrun <mthrun@informatik.uni-marburg.de>

References
                      Projections from Dimensionality Reduction Methods, MethodsX, Vol. in press, pp. 101093. doi
[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence,
               9783658205409, 2018.
                    in Cottrell, M. (Ed.), 12th International Workshop on Self-Organizing Maps and Learning Vector
                    Quantization, Clustering and Data Visualization (WSOM), IEEE Xplore, France, 2017.

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
addRowWiseC

## 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)

---

### addRowWiseC

**intern function**

**Description**

Adds the Vector DataPoint to every row of the matrix WeightVectors

**Usage**

```r
addRowWiseC(WeightVectors, DataPoint)
```

**Arguments**

- **WeightVectors**: WeightVectors. n weights with m components each
- **DataPoint**: Vector with m components

**Value**

`WeightVectors[1:m,1:n]`
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]

Author(s)

Michael Thrun

References


Chainlink

Chainlink is part of the Fundamental Clustering Problem Suit (FCPS) [Thrun/Ultsch, 2020].

Description

linear not separable dataset of two interwined chains.

Usage

data("Chainlink")

Details

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

References


Examples

```r
data(Chainlink)
str(Chainlink)

## Not run:
require(DataVisualizations)
DataVisualizations::Plot3D(Chainlink$Data,Chainlink$Cls)

## End(Not run)
```

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  

**Description**

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

**Usage**

`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**


---

EsomNeuronsAsList  

**Description**

Converts wts data (EsomNeurons) into the list form

**Arguments**

- `EsomNeurons[1:Lines, 1:Columns, 1:Variables]`: high dimensional array with grid positions in the first two dimensions
ExtendToroidalUmatrix

Details

One could describe this function as a transformation or a special case of wide to long format, see also ListAsEsomNeurons.

Value

TrainedNeurons[1:(Lines*Columns),1:Variables]
List of Weights as a matrix (not list like in R) as matrix or two dimensional array.

Author(s)

Michael Thrun, Florian Lerch

References

Ultsch, A. Maps for the visualization of high-dimensional data spaces. in Proc. Workshop on Self organizing Maps. 2003.

---

**ExtendToroidalUmatrix  Extend Toroidal Umatrix**

Description

Extends Umatrix by toroidal continuation of the given Umatrix defined by ExtendBorders in all four directions.

Usage

ExtendToroidalUmatrix(Umatrix, Bestmatches, ExtendBorders)

Arguments

- **Umatrix** [1:Lines,1:Columns] Matrix of Umatrix Heights
- **Bestmatches** [1:n,1:2] Matrix with positions of Bestmatches for n datapoints, first columns is the position in Lines and second column in Columns
- **ExtendBorders** number of lines and columns the umatrix should be extended with

Details

Function assumes that U-matrix is not planaer (has no borders), i.e. is toroidal, and not tiled. Bestmatches are moved to new positions accordingly. Example is shown in conference talk of [Thrun et al., 2020].

Value

- **Bestmatches** Array with positions of Bestmatches
GeneralizedUmatrix

Note

Currently can be only used if untiled U-Matrix (the default) is presented, but 4-tiled U-matrix does not work.

Author(s)

Michael Thrun

References


Examples

#ToDo


Description


Usage

GeneralizedUmatrix(Data, ProjectedPoints,

PlotIt=FALSE, Cls=NULL, Toroid=TRUE, Tiled=FALSE,

ComputeInR=FALSE, Parallel=FALSE, DataPerEpoch=1,...)

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-Map 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
GeneralizedUmatrix

Toroid  Optional, Default=TRUE,  
==FALSE planar computation with borders defined by projection method  
==TRUE: toroid borderless (toroidal) computation, the four borders defined by projection method are ignored.
Tiled  Optional, For plotting see plotUmatrix in package Umatrix
ComputeInR  Optional, =T: Rcode, =F Cpp Code
Parallel  Optional, =TRUE: compute parallel Cpp Code, =FALSE do not compute parallel Cpp Code
DataPerEpoch  Optional, scalar, value above zero and below 1 starts sampling and defines percentage of data points sampled in each epoch during the learning phase. Beware: Experimental!
...  Further parameters.

Details

Introduced first in the PhD thesis in [Thrun, 2018, p.46]. Furthermore the two parts of the work were peer-reviewed and published in [Ultsch/Thrun, 2017, Thrun/Ultsch, 2020].

Value

List with

EssomNeurons  [1:Lines,1:Columns,1:weights] 3-dimensional numeric array (wide format), not wts (long format)
Bestmatches  [1:n,1:2] Positions of GridConverted Projected Points on the Umatrix to the predefined Grid by Lines and Columns, First Columns has the content of the Line No and second Column of the Column number.
sehenparameters  internals for debugging
Lines  Number of Lines
Columns  Number of Columns
gplotres  output of ggplot2

Author(s)

Michael Thrun

References

GeneratePmatrix

Generates the P-matrix

Description

Generates a P-matrix to 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
... If set the Pmatrix will also be plotted

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)


ListAsEsomNeurons

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

ListAsEsomNeurons Converts List to WTS

Description
Converts wts data in list form into a 3 dimensional array

Arguments
wts_list[1:(Lines*Columns),1:Variables]
Matrix with weights in the 2nd dimension(not list() like in R)
Lines Lines/Height of the desired grid
Columns Columns/Width of the desired grid

Details
One could describe this function as a transformation or a special case of long to wide format, see also EsomNeuronsAsList

Value
EsomNeurons[1:Lines, 1:Columns, 1:Variables]
3 dimensional array containing the weights of the neural grid. For a more general explanation see reference
**Author(s)**

Michael Thrun, Florian Lerch

**References**


---

**LowLand**

**Description**

LowLand

**Usage**

LowLand(BestMatchingUnits, GeneralizedUmatrix, Data, Cls, Key, LowLimit)

**Arguments**

- **BestMatchingUnits**
  
  (1:n,1:n,1:n) BestMatchingUnits = [BMkey, BMLineCoords, BMColCoords]

- **GeneralizedUmatrix**
  
  (1:l,1:c) U-Matrix heights in Matrix form

- **Data**
  
  (1:n,1:d) data cases in lines, variables in Columns or [] or 0

- **Cls**
  
  (1:n) a possible classification of the data or [] or 0

- **Key**
  
  (1:n) the keys of the data or [] or 0

- **LowLimit**
  
  GeneralizedUmatrix heights up to this are considered to lie in the low lands
  default: LowLimit = prctile(Uheights,80) nur die 80% tiefsten

**Value**

- **LowLandBM**
  
  the unique BestMatchingUnits in the low lands of an u-Matrix

- **LowLandInd**
  
  index such that UniqueBM = BestMatchingUnits(UniqueInd,]

- **LowLandData**
  
  Data reduced to LowLand: LowLandData = Data(LowLandInd,]

- **LowLandCls**
  
  Cls reduced to LowLand: LowLandCls = Cls(LowLandInd)

- **LowLandKey**
  
  Key reduced to LowLand: LowLandKey = Key(LowLandInd)

**Author(s)**

ALU 2021 in matlab, MCT reimplemented in R
Normalize U-matrix

Description


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


Author(s)

Felix Pape, 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)
#see also ProjectionBasedClustering package for other common projection methods

resUmatrix=GeneralizedUmatrix(Data,ProjectedPoints)
## Normalization
### plotTopographicMap

**Visualizes the Generalized U-matrix in 3D**

#### Description

The generalized U-matrix is visualized as the topographic map with hypsometric tints. The topographic map represents high-dimensional distance and density-based structures in form of a 3D landscape.

#### Usage

```r
plotTopographicMap(GeneralizedUmatrix, BestMatchingUnits,
                    Cls=NULL, ClsColors=NULL, Imx=NULL, Names=NULL,
                    BmSize=0.5, RenderingContourLines=TRUE,...)
```

#### 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 as spheres onto the topographic map

- **Cls**
  
  `(1:n), numerical vector of classification of k clusters, one label for each best-match at that given point

- **ClsColors**
  
  Vector of colors that will be used to colorize the different clusters, default is `GeneralizedUmatrix::DefaultColorSequence`

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

- **Names**
  
  If set: `[1:k] character vector naming the k clusters 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), etc.

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

- **RenderingContourLines**
  
  `FALSE`: disables plotting of contour lines resulting in a much faster plot.
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
- **ExtendBorders** scalar, extends Umatrix by toroidal continuation of the given Umatrix
- **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

**NamesPosition** same as in bgplot3d

**NamesColors** same as col in bgplot3d

**NamesCex** same as cex in bgplot3d

**NamesTitle** same as title in bgplot3d

**NamesPch** same as pch in bgplot3d

### Details

The visualization 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 (Patterson and Kelso 2004). 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 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). The number of clusters can be estimated by the number of valleys of the visualization. The clustering is valid if mountains do not partition clusters indicated by colored points of the same color and colored regions of points (see examples in section 4.1 and 4.2)." [Thrun/Ultsch, 2020].

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 as the number of valleys (Thrun et al., 2016). Please see chapter 5 of [Thrun, 2018] for further details.

### Value

An object of class "htmlwidget" in mode invisible, please rglwidget for details.

### Note

First version of algorithm was 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)

## Open window in specific resolution
#relevant if Names given

library(rgl)
r3dDefaults$windowRect = c(0,0,1200,1200)
plotTopographicMap(GeneralizedUmatrix = resUmatrix$Umatrix,resUmatrix$Bestmatches)

## Not run:
## To save as STL for 3D printing
rgl::writeSTL("GenerelizedUmatrix_3d_model.stl")

## Save the visualization as a picture with
library(rgl)
rgl.snapshot("test.png")
## End(Not run)

## Save interactive html file
## Not run:
widgets=plotTopographicMap(GeneralizedUmatrix = resUmatrix$Umatrix,resUmatrix$Bestmatches)
if(requireNamespace("htmlwidgets"))
  htmlwidgets::saveWidget(widgets,file = "interactiveTopographicMap.html")
## End(Not run)

ReduceToLowLand

---

### Description

ReduceToLowLand

### Usage

ReduceToLowLand(BestMatchingUnits, GeneralizedUmatrix, Data = NULL, Cls = NULL, Key = NULL, LowLimit, Force=FALSE)

### Arguments

- **BestMatchingUnits**
  - (1:n,1:n,1:n) BestMatchingUnits =BMkey, BMLineCoords, BMColCoords[
- **GeneralizedUmatrix**
  - (1:1:n,1:1:c) U-Matrix heights in Matrix form
- **Data**
  - (1:n,1:d) data cases in lines, variables in Columns or [] or 0
- **Cls**
  - (1:n) a possible classification of the data or [] or 0
- **Key**
  - (1:n) the keys of the data or [] or 0
- **LowLimit**
  - GeneralizedUmatrix heights up to this are considered to lie in the low lands
default: LowLimit = prctile(Uheights,80) nur die 80# tiefsten
- **Force**
  - ==TRUE: Always perform reduction

### Value

- **LowLandBM**
  - the unique BestMatchingUnits in the low lands of an u-Matrix
- **LowLandInd**
  - index such that UniqueBM = BestMatchingUnits(UniqueInd,]
- **LowLandData**
  - Data reduced to LowLand: LowLandData = Data(LowLandInd,]
- **LowLandCls**
  - Cls reduced to LowLand: LowLandCls = Cls(LowLandInd)
- **LowLandKey**
  - Key reduced to LowLand: LowLandKey = Key(LowLandInd)

### Author(s)

ALU 2021 in matlab, MCT reimplemented in R
sESOM4BMUs  

**simplified ESOM**

---

**Description**

Internfunction for the simplified ESOM Algorithmus [Thrun/Ultsch, 2020] for fixed BestMatchingUnits

**Usage**

```r
sESOM4BMUs(BMUs, Data, esom, toroid,
            CurrentRadius, ComputeInR=FALSE, Parallel=FALSE)
```

**Arguments**

- **BMUs**: [1:Lines,1:Columns], BestMAtingUnits 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 Code
- **Parallel** =T: Rcode, =F Cpp Code

**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
setdiffMatrix

References


See Also

GeneralizedUmatrix

Description

setdiffMatrix shortens Matrix2Curt by those rows that are in both matrices.

Arguments

Matrix2Curt[n,k] matrix, which will be shortened by x rows
Matrix2compare[m,k] matrix whose rows will be compared to those of Matrix2Curt x rows in Matrix2compare equal rows of Matrix2Curt (order of rows is irrelevant). Has the same number of columns as Matrix2Curt.

Value

V$CurtedMatrix[n-x,k] Shortened Matrix2Curt

Author(s)

Michael Thrun with the help of Catharina Lippmann

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) with high-dimensional data.
Usage

TopviewTopographicMap(GeneralizedUmatrix, BestMatchingUnits,

Cls, ClsColors = NULL, Imx = NULL,

ClsNames = NULL, BmSize = 6, DotLineWidth = 2,

alpha = 1, ...)

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

ClsNames
If set: [1:k] character vector naming the k classes for the legend. In this case, further parameters with the possibility to adjust are: LegendCex: (size); NamesOrientation: Legend position "v" or "h"; NamesTitle: title of legend.

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

DotLineWidth
...

alpha
...

... Tiled Should the Umatrix be drawn 4times?

main set specific title in plot

ExtendBorders scalar, extends Umatrix by toroidal continuation of the given Umatrix

MainCex scalar, magnification to be used for legend

LegendCex scalar, magnification to be used for main titles

Further Arguments relevant for interactive shiny application

Details

Please see plotTopographicMap. This function is currently still experimental because not all functionality is fully tested yet.

Value

plotly handler
Note
Names are currently under development, Imx in testing phase.

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**

**Arguments**

BestMatchingUnits

(1:n,1:d) BMKey = BestMatchingUnits[,1]

GeneralizedUmatrix

(1:Lines,1:Columns) a GeneralizedUmatrix

**Value**

<table>
<thead>
<tr>
<th>Uheights</th>
<th>Uheights</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMLineCoords</td>
<td>BMLineCoords</td>
</tr>
<tr>
<td>BMColCoords</td>
<td>BMColCoords</td>
</tr>
</tbody>
</table>

**Author(s)**

ALU 2021 in matlab, MCT reimplemented in

**UmatrixColormap**  
*U-Matrix colors*

**Description**

Defines the default color sequence for plots made for Umatrix

**Usage**

data("UmatrixColormap")

**Format**

Returns the vectors for a (heat) colormap.

**UniqueBestMatchingUnits**

**Description**

UniqueBestMatchingUnits

**Usage**

UniqueBestMatchingUnits(NonUniqueBestMatchingUnits)
Arguments

NonUniqueBestMatchingUnits
(1:n,1:n,1:n) UniqueBestMatchingUnits = [BMkey, BMLineCoords, BMColCoords]

Value

UniqueBM (1:u,1:u,1:u) UniqueBM = [UBMkey, UBMLineCoords, UBMColCoords]
UniqueInd Index such that UniqueBM = UniqueBestMatchingUnits(UniqueInd,:)
Uniq2AllInd Index such that UniqueBestMatchingUnits = UniqueBM(Uniq2AllInd,:)

Author(s)

ALU 2021 in matlab, MCT reimplemented in R

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

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
XYcoords2LinesColumns

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

* AU-Matrix
  NormalizeUmatrix, 15
* Chainlink
  Chainlink, 6
* DR
  GeneralizedUmatrix-package, 3
* DimensionalityReduction
  GeneralizedUmatrix-package, 3
* ESOM
  sESOM4BMUs, 20
* Extend Toroidal U-matrix
  ExtendToroidalUmatrix, 9
* Extend Toroidal U-matrix
  ExtendToroidalUmatrix, 9
* ExtendToroidalUmatrix
  ExtendToroidalUmatrix, 9
* FCPS
  Chainlink, 6
* Generalized U-matrix
  GeneralizedUmatrix, 10
  plotTopographicMap, 16
  TopviewTopographicMap, 21
  upscaleUmatrix, 26
* GeneralizedUmatrix
  GeneralizedUmatrix, 10
  GeneralizedUmatrix-package, 3
  plotTopographicMap, 16
  TopviewTopographicMap, 21
* GeneratePswarmVisualization
  sESOM4BMUs, 20
* P-Matrix
  GeneratePmatrix, 12
* P-matrix
  GeneratePmatrix, 12
* Pmatrix
  GeneratePmatrix, 12
* Projection
  GeneralizedUmatrix-package, 3
* U*-Matrix
  CalcUstarmatrix, 6
* U*-matrix
  CalcUstarmatrix, 6
* U*matrix
  CalcUstarmatrix, 6
* U-Matrix
  GeneralizedUmatrix, 10
  GeneralizedUmatrix-package, 3
  plotTopographicMap, 16
  TopviewTopographicMap, 21
* U-matrix
  GeneralizedUmatrix-package, 3
  upscaleUmatrix, 26
* Umatrix
  ExtendToroidalUmatrix, 9
  GeneralizedUmatrix, 10
  GeneralizedUmatrix-package, 3
  plotTopographicMap, 16
  TopviewTopographicMap, 21
* Ustar-Matrix
  CalcUstarmatrix, 6
* Ustar-matrix
  CalcUstarmatrix, 6
* Ustarmatrix
  CalcUstarmatrix, 6
* Visualization
  GeneralizedUmatrix-package, 3
* abstract U-matrix
  NormalizeUmatrix, 15
* datasets
  Chainlink, 6
* map
  plotTopographicMap, 16
  TopviewTopographicMap, 21
* topographic map
  plotTopographicMap, 16
  TopviewTopographicMap, 21
  upscaleUmatrix, 26
* topographic
plotTopographicMap, 16
TopviewTopographicMap, 21

* visualization *
GeneralizedUmatrix, 10
plotTopographicMap, 16
TopviewTopographicMap, 21

addRowWiseC, 5

bgplot3d, 17
CalcUstarmatrix, 6
calcUstarmatrix(CalcUstarmatrix), 6
Chainlink, 6

DefaultColorSequence, 7
Delta3DWeightsC, 8

EsomNeuronsAsList, 8, 13
ExtendToroidalUmatrix, 9

GeneralizedUmatrix, 8, 10, 18, 21
GeneralizedUmatrix-package, 3
GeneratePmatrix, 12
getUmatrix4Projection
(GeneralizedUmatrix), 10

list, 9
ListAsEsomNeurons, 9, 13
LowLand, 14

NormalizeUmatrix, 15

plot, 17
plotTopographicMap, 10, 16, 22, 23, 26

ReduceToLowLand, 19
rglwidget, 17

sESOM4BMUs, 20
setdiffMatrix, 21

TopviewTopographicMap, 21
trainstepC, 23

Uheights4Data, 24
UmatrixColormap, 25
UniqueBestMatchingUnits, 25
upscaleUmatrix, 26

XYcoords2LinesColumns, 27