Package 'GeneralizedUmatrix'

January 31, 2018

Type Package

Title Credible Visualization for Two-Dimensional Projections of Data

Version 1.0.0

Date 2018-01-28

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]. 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

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 3.0)

NeedsCompilation yes

SystemRequirements C++11

LazyLoad yes

LazyData TRUE

URL https://www.uni-marburg.de/fb12/datenbionik/software-en

Encoding UTF-8

Author Michael Thrun [aut, cre, cph], Alfred Ultsch [ths]

Repository CRAN

Date/Publication 2018-01-31 18:13:07 UTC

R topics documented:

`GeneralizedUmatrix-package` *GeneralizedUmatrix*

Description

Projections from a high dimensional data space onto a two dimensional plane are used to detect structures, such as clusters, in multivariate data. The generalized Umatrix is able to visualize errors of these two-dimensioal scatter plots by using a 3D topographic map.

Details

Package:	GeneralizedUmatrix
Type:	Package
Version:	0.9.7
Date:	2018-01-23
License:	CC BY-NC-SA 4.0

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

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

Author(s)

Michal Thrun

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

References

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, Heidelberg, ISBN: 978-3-658-20539-3, https://doi.org/10.1007/978-3-658-20540-9, 2018.

[Ultsch/Thrun, 2017] Ultsch, A., & Thrun, M. C.: Credible Visualizations for Planar Projections, 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("Lsun3D")
Data=Lsun3D$Data
Cls=Lsun3D$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 strcuture
 DataVisualizations::SilhouettePlot(Data,Cls)

## End(Not run)
```

DefaultColorSequence *Default color sequence for plots*

Description

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

Usage

```
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

```
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

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, Heidelberg, ISBN: 978-3-658-20539-3, https://doi.org/10.1007/978-3-658-20540-9, 2018.

GeneralizedUmatrix *Generalized U-Matrix for projection methods*

Description

Generalized U-Matrix visualizes high-dimensional distance and density based structurs in two-dimensional scatter plots of projectios methods like CCA,MDS,PCA or NeRV with the help of a topographic map with hypsometrioc 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]n by 2 matrix containing coordinates of the Projection: A matrix of the fitted configuration.
PlotIt	Optional,bool, defaut=FALSE, if =TRUE: U-Marix of every current Position of Databots will be shown
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] (see ReadUMX in package DataIO)
EsomNeurons	[Lines,Columns,weights] 3-dimensional numeric array (wide format), not wts (long format)
Bestmatches	[1:n,OutputDimension] GridConverted Projected Points information converted by convertProjectionProjectedPoints() to predefined Grid by Lines and Columns
gplotres	Ausgabe von ggplot

Author(s)

Michael Thrun

References

[Ultsch, 2003] Ultsch, A.: Maps for the visualization of high-dimensional data spaces, Proc. Workshop on Self organizing Maps (WSOM), pp. 225-230, Kyushu, Japan, 2003.

[Thrun et al., 2016] Thrun, M. C., Lerch, F., Loetsch, J., & Ultsch, A.: Visualization and 3D Printing of Multivariate Data of Biomarkers, in Skala, V. (Ed.), International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG), Vol. 24, Plzen, http://wscg.zcu.cz/wscg2016/short/A43-full.pdf, 2016.

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, Heidelberg, ISBN: 978-3-658-20539-3, https://doi.org/10.1007/978-3-658-20540-9, 2018.

[Ultsch/Thrun, 2017] Ultsch, A., & Thrun, M. C.: Credible Visualizations for Planar Projections, 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("Lsun3D")
Data=Lsun3D$Data
Cls=Lsun3D$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)
```

Lsun3D *Lsun3D inspired by FCPS*

Description

clearly defined clusters, different variances

Usage

```
data("Lsun3D")
```

Details

Size 404, Dimensions 3

Dataset defined discontinuites, where the clusters have different variances. Three main Clusters, and four Outliers (in Cluster 4). Further details are written down in [Thrun, 2018].

References

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, Heidelberg, ISBN: 978-3-658-20539-3, https://doi.org/10.1007/978-3-658-20540-9, 2018.

Examples

```
data(Lsun3D)
str(Lsun3D)
Cls=Lsun3D$Cls
Data=Lsun3D$Data
```

---

plotTopographicMap *shows generelized Umatrix visualization*

Description

visualizes high-dimensional distance and density based structurs in two-dimensional scatter plots using a atopographic map with hypsometric tints

Usage

```
plotTopographicMap(GeneralizedUmatrix, BestMatchingUnits=NULL,

Cls=NULL,ClsColors=NULL,Imx=NULL, Tiled=FALSE, BmSize=0.5,ShowAxis=F)
```

Arguments

GeneralizedUmatrix	(1:Lines,1:Columns), Umatrix to be plotted
BestMatchingUnits	(1:n,1:2), Positions of bestmatches to be plotted onto the Umatrix
Cls	(1:n), Class identifier 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
Tiled	Should the Umatrix be drawn 4times?
BmSize	size(diameter) of the points in the visualizations. The points represent the BestMatchingUnits
ShowAxis	shall the axis be shown?

Details

see chapter 5 of [Thrun, 2018] for descriptive details.

Note

Algorithm is partly based on the Umatrix package.

Author(s)

Michael Thrun

References

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, Heidelberg, ISBN: 978-3-658-20539-3, https://doi.org/10.1007/978-3-658-20540-9, 2018.

[Thrun et al., 2016] Thrun, M. C., Lerch, F., Loetsch, J., & Ultsch, A.: Visualization and 3D Printing of Multivariate Data of Biomarkers, in Skala, V. (Ed.), International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG), Vol. 24, Plzen, http://wscg.zcu.cz/wscg2016/short/A43-full.pdf, 2016.

See Also

GeneralizedUmatrix

Examples

```
data("Lsun3D")
Data=Lsun3D$Data
Cls=Lsun3D$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("GenerelizedUmatrix_3d_model.stl")

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

sESOM4BMUs *simplified 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 betweeen 1 to x
ComputeInR	=T: Rcode, =F Cpp Codenumber betweeen 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 seperated usage!

Author(s)

Michael Thrun

References

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, Heidelberg, ISBN: 978-3-658-20539-3, https://doi.org/10.1007/978-3-658-20540-9, 2018.

See Also

GeneralizedUmatrix

`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

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, Heidelberg, ISBN: 978-3-658-20539-3, https://doi.org/10.1007/978-3-658-20540-9, 2018.

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

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 numer 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

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, Heidelberg, ISBN: 978-3-658-20539-3, https://doi.org/10.1007/978-3-658-20540-9, 2018.

Examples

```
data("Lsun3D")
Data=Lsun3D$Data
InputDistances=as.matrix(dist(Data))
res=cmdscale(d=InputDistances, k = 2, eig = TRUE, add = FALSE, x.ret = FALSE)
ProjectedPoints=as.matrix(res$points)
GridConvertedPoints=XYcoords2LinesColumns(ProjectedPoints[,1],ProjectedPoints[,2],PlotIt=FALSE)
```

Index