MDSRefMaps

November 27, 2016

Type Package

Title MDSRefMaps: an R Package for Multidimensional Scaling Reference Maps and Projections

Version 1.0.0

Author Francois BONNARDEL, Christophe BECAVIN and Nicolas TCHITCHEK

Maintainer Nicolas TCHITCHEK <nicolas.tchitchek@gmail.com> and Francois BONNARDEL <bonnardel.francois.bioinfo@gmail.com>

Description Multidimensional Scaling (MDS) methods aim to represent similarities between high-dimensional objects in a low dimensional space, generally in two or three dimensions for visualization purpose. MDS representations are especially useful to explore and interpret high-dimensional biological data. However, comparisons between the different MDS representations can be difficult because of the absence of a common structure. The MDSRefMaps algorithm allows the projection of additional high-dimensional objects over a predefined MDS representation. The predefined representations are named MDS Reference Maps and the overlaid representations are named MDS Projections. The MDSRefMaps algorithm is based on a molecular dynamics approach and has been implemented in C++ to handle large amount of objects. Moreover, the MDSRefMaps algorithm can handle both Euclidean and Manhattan distances and can also handle incomplete distance matrices (matrices with missing values).

License GPL-3 | file LICENSE

Depends R (>= 3.1),

Imports Rcpp (>= 0.12.4),
ggplot2,
plyr

biocViews DimensionReduction, Visualization, Transcriptomics

LinkingTo Rcpp

RoxygenNote 5.0.1

R topics documented:

`computeEntourageScore` *Computation of the Entourage Score*

Description

This function is used to compute the Entourage Score between two distance matrices, which quantifies the local quality of a MDS representation.

The Entourage Score corresponds to the normalised number of identical nearest neighbours for each object in the two distance matrices.

Usage

```
computeEntourageScore(dist1, dist2, k = 3)
```

Arguments

`dist1`	a numeric matrix of the first distance matrix
`dist2`	a numeric matrix of the second distance matrix
`k`	a numeric indicating the number of nearest neighbours to compare

Details

This function has been implemented in C++ for fast computations and can be executed from R using this wrapper.

Value

a numeric value of the Entourage Score

`computeKruskalStress` *Computation of the Kruskal Stress*

Description

This function is used to compute the Kruskal Stress between two distance matrices, which quantifies the global quality of a MDS representation.

The Kruskal Stress corresponds to the quantify of information lost during the dimentionality reduction process.

Usage

```
computeKruskalStress(dist1, dist2)
```

Arguments

`dist1`	a numeric matrix of the first distance matrix
`dist2`	a numeric matrix of the second distance matrix

Details

This function has been implemented in C++ for fast computations and can be executed from R using this wrapper.

Value

a numeric value the Kruskal Stress

`distEuclidean` *Computation of a distance matrix using the Euclidean metric*

Description

This function computes the distance matrix from a numeric matrix using the Euclidean metric.

Each row of the input matrix must corresponds to an object and each column must corresponds to an attribute.

Usage

```
distEuclidean(data)
```

Arguments

`data`	a numeric matrix. Rows must correspond to the particles and columns must correspond to the attributes

Details

This function has been implemented in C++ for fast computations and can be executed from R using this wrapper.

Value

a numeric matrix containing the Euclidean distances betweens all particles

`distManhattan` *Computation of a distance matrix using the Manhattan metric*

Description

This function computes the distance matrix from a numeric matrix using the Manhattan metric.

Each row of the input matrix must corresponds to an object and each column must corresponds to an attribute.

Usage

```
distManhattan(data)
```

Arguments

`data`	a numeric matrix. Rows must correspond to the objects and columns must correspond to the attributes

Details

This function has been implemented in C++ for fast computations and can be executed from R using this wrapper.

Value

a numeric matrix containing the Manhattan distances betweens all objects

---

`MDSProjection` *Construction of a MDS Projection*

Description

This function computes a MDS Projection based on MDS Reference Map and a distance matrix.

A MDS Projection consists on a MDS Reference Map on which additional objects have been overlayed. MDS Projections can be computed based on the Euclidean or Manhattan metrics using the 'metric' parameter. The initialization space of object positions can be specified using the 'init' parameter.

Usage

```
MDSProjection(refmap, dist, k = 2, init = "svd", metric = "euclidean",
  max_it = 6 * 10^6, stress_sd_th = 10^-4, stack_length = 500,
  verbose = TRUE)
```

Arguments

`refmap`	a result from RefMap function
`dist`	a numeric matrix with all pairwise distances between objects of the representation
`k`	a numeric value specifying the desired number of dimensions in the resulting MDS Projection representation
`init`	a character or a numeric matrix specifying how the objects are positioned in the initial configuration. Possible character values are 'rand', 'center', 'svd' (please refer to the details section for more details). Object positions in the initial configuration can be explicitly specified using a numeric matrix where the rows correspond to the objects and where the columns correspond to the MDS dimensions.
`metric`	a character indicating the distance metric to use ("euclidean" or "manhattan")
`max_it`	a numeric defining the maximal number of steps the algorithm can perform
`stress_sd_th`	a numeric defining the threshold for the standard deviation of Kruskal Stress
`stack_length`	a numeric defining the length of the Kruskal Stress stack (used to compute the standard deviation of the Kruskal Stress)
`verbose`	a boolean enabling the display of debug information at each step of the algorithm

Details

Use RefMaps algorithm to compute the projection of additional objects. Efficient to compare multiple projection, for example made on subsets of large biologicals datasets.

This implementation allows to used incomplete distance matrices (distance matrices with missing values modeled by NA). Furthermore, distance matrices can be computed based on the Euclidean or Manhattan metrics.

This implementation has been implemented in C++ to handle large sets of high-dimensional objects.

Value

a list of 3 elements containing the position of the objects ('points' element), the Kruskal Stress ('stress' element), and the Entourage Score ('entourage' element)

`MDSReferenceMap` *Construction of a MDS Reference Map*

Description

This function computes a MDS Reference Map based on a distance matrix.

A MDS Reference Map corresponds to a regular MDS representation on which additional objects can be projected. MDS Reference Maps can be computed based on the Euclidean or Manhattan metrics using the 'metric' parameter. The initialization space of object positions can be specified using the 'init' parameter.

Usage

```
MDSReferenceMap(dist, k = 2, init = "svd", metric = "euclidean",
  max_it = 6 * 10^6, stress_sd_th = 10^-4, stack_length = 500,
  verbose = TRUE)
```

Arguments

`dist`	a numeric matrix with all pairwise distances between objects of the representation
`k`	a numeric value specifying the desired number of dimensions in the resulting Reference Map representation
`init`	a character value or a numeric matrix specifying how the objects are positioned in the initial configuration. Possible character values are 'rand', 'center', 'svd' (please refer to the details section for more details). Object positions in the initial configuration can be explicitly specified using a numeric matrix where the rows correspond to the objects and where the columns correspond to the MDS dimensions (in k dimensions).
`metric`	a character indicating the distance metric to use ("euclidean" or "manhattan")
`max_it`	a numeric defining the maximal number of steps the algorithm can perform
`stress_sd_th`	a numeric defining the threshold for the standard deviation of Kruskal Stress
`stack_length`	a numeric defining the length of the Kruskal Stress stack (used to compute the standard deviation of the Kruskal Stress)
`verbose`	a boolean enabling the display of debug information at each step of the algorithm

Details

The RefMaps algorithm implements SVD-MDS algorithm which is based on a molecular dynamic approach (Becavin et al.). This metric performs a dimensionality reduction of the original space by modeling objects by particles and pairwise distances between them by repulsion and attraction forces. SVD-MDS metric use Verlet algorithm (Loup Verlet in 1967) to compute the MDS representation. Algorithm constants can be specified via the 'setConts()' function.

This implementation allows to used incomplete distance matrices (distance matrices with missing values modeled by NA). Furthermore, distance matrices can be computed based on the Euclidean or Manhattan metrics.

This implementation has been implemented in C++ to handle large sets of high-dimensional objects. init. KS+entourage.

Value

a list of 3 elements containing the position of the objects ('points' element), the Kruskal Stress ('stress' element), and the Entourage Score ('entourage' element)

`MDSReferenceMapwithProjection`

Construction of a MDS Reference Map with a MDS Projection

Description

This function computes a MDS Reference Map with an associated MDS Projection based on a distance matrix.

A MDS Projection consists on a MDS Reference Map on which additional objects have been overlayed. MDS Projections can be computed based on the Euclidean or Manhattan metrics using the 'metric' parameter. The initialization space of object positions can be specified using the 'init' parameter.

Usage

```
MDSReferenceMapwithProjection(dist, ref_values, k = 2, init = "svd",
  metric = "euclidean", max_it = 6 * 10^6, stress_sd_th = 10^-4,
  stack_length = 500, verbose = TRUE)
```

Arguments

`dist`	a numeric matrix with all pairwise distances between objects of the representation
`ref_values`	a numeric vector indicating how the objects must be considered. Objects assigned with 1 are treated as reference objects, objects assigned with 0 are treated as projection objects and objects assigned with NA are not used
`k`	a numeric value specifying the desired number of dimensions in the resulting MDS Projection representation
`init`	a character or a numeric matrix specifying how the objects are positioned in the initial configuration. Possible character values are 'rand', 'center', 'svd' (please refer to the details section for more details). Object positions in the initial configuration can be explicitly specified using a numeric matrix where

	the rows correspond to the objects and where the columns correspond to the MDS dimensions.
`metric`	a character indicating the distance metric to use ("euclidean" or "manhattan")
`max_it`	a numeric indicating the maximal number of steps the algorithm can perform
`stress_sd_th`	a numeric defining the threshold for the standard deviation of Kruskal Stress
`stack_length`	a numeric defining the length of the Kruskal Stress stack (used to compute the standard deviation of the Kruskal Stress)
`verbose`	a boolean enabling the display of debug information at each step of the algorithm

Details

A numeric vector is used to specify the objects to use as reference and the objects to add on the Reference Map.

Value

a list of 3 elements containing the position of the objects ('points' element), the Kruskal Stress ('stress' element), and the Entourage Score ('entourage' element)

`plotMDS` *Generation of ggplot graphical representations for MDS Reference Maps and MDS Projections.*

Description

This function generates a graphical representation for MDS Reference Maps and MDS Projections. Objects can be shaped and colored using the 'color' and 'shape' parameters. Convex hulls for given sets of objects can also be displayed.

Usage

```
plotMDS(mds, color = NULL, shape = NULL, polygon = NULL, title = "MDS",
  display.legend = TRUE)
```

Arguments

`mds`	a MDS result provided by the 'MDSReferenceMap()', 'MDSProjection()', or 'MDSReferenceMapwithProjection()' functions
`color`	a vector used to color the points
`shape`	a vector used for the shape of the points
`polygon`	a vector used to color the convex hull
`title`	a character specifying the title of the representation
`display.legend`	a logical specifying if the graphic legend must be displayed

`setConstants` *Setting of the algorithm constants*

Description

This function is used to set the different constants of the MDSRefMaps algorithm.

Usage

```
setConstants(K = 1, F = 0.1, DELTA_T = 0.001, MASS = 10,
  ACC_THRESHOLD = 0)
```

Arguments

`K`	a numeric value indicating the spring strength between two particles
`F`	a numeric value indicating the friction of the springs
`DELTA_T`	a numeric value indicating the time between two steps of the algorithm
`MASS`	a numeric value indicating the mass of each particle
`ACC_THRESHOLD`	a numeric value indicating the acceleration threshold (a value of 0 disactivate this acceleration threshold)

Details

The 'ACC_THRESHOLD' parameter can be used to contrain the object accelerations.

Index