Package ‘mlmts’

April 19, 2022

Type Package

Title Machine Learning Algorithms for Multivariate Time Series

Version 1.0.1

Description An implementation of several machine learning algorithms for multivariate time series. The package includes functions allowing the execution of clustering, classification or outlier detection methods, among others. It also incorporates a collection of multivariate time series datasets which can be used to analyse the performance of new proposed algorithms. Practitioners from a broad variety of fields could benefit from the general framework provided by 'mlmts'.

License GPL-2

Encoding UTF-8

LazyData true

LazyDataCompression xz

Depends R (>= 4.0.0)

RoxygenNote 7.1.2

Imports quantspec, waveslim, Rfast, TSclust, forecast, tseries, TSA, tsfeatures, tseriesChaos, freqdom, e1071, dtw, base, psych, complexplus, MTS, Matrix, ggplot2, multiwave, MASS, fda.usc, TSdist, evolqg, geigen, DescTools, pracma, pspline, Rdpack, stats, ClusterR, AID, caret, ranger

RdMacros Rdpack

NeedsCompilation no

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Author Angel Lopez-Oriona [aut, cre],
Jose A. Vilar [aut]

Maintainer Angel Lopez-Oriona <oriona38@hotmail.com>

Repository CRAN

Date/Publication 2022-04-19 14:42:29 UTC

R topics documented:

`AtrialFibrillation` *AtrialFibrillation*

Description

Multivariate time series (MTS) of two-channel ECG recordings of atrial fibrillation. The database has been created from data used in the Computers in Cardiology Challenge 2004.

Usage

```
data(AtrialFibrillation)
```

Format

A `list` with two elements, which are:

`data` A list with 30 MTS.

`classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a matrix formed by 640 rows (time points) indicating ECG measures and 2 columns (variables) indicating ECG leads. The first 15 elements correspond to the training set, whereas the last 15 elements correspond to the test set. The numeric vector `classes` is formed by integers from 1 to 3, indicating that there are 3 different classes in the database. Each class is associated with a different type of atrial fibrillation. For more information, see Bagnall et al. (2018).

References

Bagnall A, Dau HA, Lines J, Flynn M, Large J, Bostrom A, Southam P, Keogh E (2018). “The UEA multivariate time series classification archive, 2018.” *arXiv preprint arXiv:1811.00075*.

Ruiz AP, Flynn M, Large J, Middlehurst M, Bagnall A (2021). “The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances.” *Data Mining and Knowledge Discovery*, **35**(2), 401–449.

Bagnall A, Lines J, Vickers W, Keogh E (2022). “The UEA & UCR Time Series Classification Repository.” *www.timeseriesclassification.com*.

`BasicMotions` *BasicMotions*

Description

Multivariate time series (MTS) of four students performing four activities while wearing a smart watch.

Usage

```
data(BasicMotions)
```

Format

A `list` with two elements, which are:

- `data` A list with 80 MTS.
- `classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a matrix formed by 100 rows (time points) indicating movement and 6 columns (variables). The first 40 elements correspond to the training set, whereas the last 40 elements correspond to the test set. The numeric vector `classes` is formed by integers from 1 to 4, indicating that there are 4 different classes in the database. Each class is associated with a different physical activity. For more information, Bagnall et al. (2018).

References

Bagnall A, Dau HA, Lines J, Flynn M, Large J, Bostrom A, Southam P, Keogh E (2018). "The UEA multivariate time series classification archive, 2018." *arXiv preprint arXiv:1811.00075*.

Ruiz AP, Flynn M, Large J, Middlehurst M, Bagnall A (2021). "The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances." *Data Mining and Knowledge Discovery*, **35**(2), 401–449.

Bagnall A, Lines J, Vickers W, Keogh E (2022). "The UEA & UCR Time Series Classification Repository." *www.timeseriesclassification.com*.

dis_2dsvd *Constructs a pairwise distance matrix based on two-dimensional singular value decomposition (2dSVD)*

Description

`dis_2dsvd` returns a pairwise distance matrix based on the 2dSVD distance measure proposed by Weng and Shen (2008).

Usage

```
dis_2dsvd(X, var_u = 0.9, var_v = 0.9, features = FALSE)
```

Arguments

`X`	A list of MTS (numerical matrices).
`var_u`	Rate of retained variability concerning the row-row covariance matrix.
`var_v`	Rate of retained variability concerning the column-column covariance matrix.
`features`	Logical. If `features = FALSE` (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T^{(1)}$ and $\boldsymbol{X}_T^{(2)}$ is defined as

$$d_{Eros}(\boldsymbol{X}_T^{(1)}, \boldsymbol{X}_T^{(2)}$$

Value

If `features = FALSE` (default), returns a distance matrix based on the distance d_{2dSVD}. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute the distance d_{2dSVD}.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weng X, Shen J (2008). "Classification of multivariate time series using two-dimensional singular value decomposition." *Knowledge-Based Systems*, **21**(7), 535–539.

Examples

```
toy_dataset <- BasicMotions$data[1 : 10] # Selecting the first 10 MTS from the
# dataset BasicMotions
distance_matrix <- dis_2dsvd(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_2dsvd
feature_dataset <- dis_2dsvd(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

`dis_cor` *Constructs a pairwise distance matrix based on auto and cross-correlations*

Description

`dis_cor` returns a pairwise distance matrix based on a generalization of the dissimilarity introduced by D'Urso and Maharaj (2009).

Usage

```
dis_cor(X, lag_max = 1, features = FALSE)
```

Arguments

`X`	A list of MTS (numerical matrices).
`lag_max`	The maximum lag considered to compute the auto and cross-correlations.
`features`	Logical. If `features = FALSE` (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as

$$d_{COR}(\boldsymbol{X}_T, \boldsymbol{Y}_T) = \left| ||\widehat{\boldsymbol{\theta}}_{AC}^{\boldsymbol{X}_T} - \widehat{\boldsymbol{\theta}}_{AC}^{\boldsymbol{Y}_T}||^2 + ||\widehat{\boldsymbol{\theta}}_{CC}^{\boldsymbol{X}_T} - \widehat{\boldsymbol{\theta}}_{CC}^{\boldsymbol{Y}_T}||^2 \right|^{1/2},$$

where $\widehat{\boldsymbol{\theta}}_{AC}^{\boldsymbol{X}_T}$ and $\widehat{\boldsymbol{\theta}}_{AC}^{\boldsymbol{Y}_T}$ are vectors containing the estimated autocorrelations within $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, respectively, and $\widehat{\boldsymbol{\theta}}_{CC}^{\boldsymbol{X}_T}$ and $\widehat{\boldsymbol{\theta}}_{CC}^{\boldsymbol{Y}_T}$ are vectors containing the estimated cross-correlations within $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, respectively.

Value

If `features = FALSE` (default), returns a distance matrix based on the distance d_{COR}. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute the distance d_{COR}.

Author(s)

Ángel López-Oriona, José A. Vilar

References

D'Urso P, Maharaj EA (2009). "Autocorrelation-based fuzzy clustering of time series." *Fuzzy Sets and Systems*, **160**(24), 3565–3589.

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_cor(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_cor
distance_matrix <- dis_cor(toy_dataset, lag_max = 5) # Considering
# auto and cross-correlations up to lag 5 in the computation of the distance
feature_dataset <- dis_cor(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

`dis_dtw_1` *Constructs a pairwise distance matrix based on multivariate dynamic time warping*

Description

`dis_dtw_1` returns a pairwise distance matrix based on one of the multivariate extensions of the well-known dynamic time warping distance (Shokoohi-Yekta et al. 2017).

Usage

```
dis_dtw_1(X, normalization = FALSE, ...)
```

Arguments

`X` A list of MTS (numerical matrices).

`normalization` Logical. If `normalization = TRUE` (default), the normalized distance is computed. Otherwise (default), no normalization is taken into account

`...` Additional parameters for the function. See `dtw`.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS is defined as the sum of the standard dynamic time warping distances between each corresponding pair of dimensions (univariate time series)

Value

The computed pairwise distance matrix.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Shokoohi-Yekta M, Hu B, Jin H, Wang J, Keogh E (2017). "Generalizing DTW to the multi-dimensional case requires an adaptive approach." *Data mining and knowledge discovery*, **31**(1), 1–31.

See Also

dis_dtw_2, dis_mahalanobis_dtw

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 5] # Selecting the first 5 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_dtw_1(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_dtw_1 without normalization
distance_matrix_normalized <- dis_dtw_1(toy_dataset, normalization = TRUE) # Computing the pairwise
# distance matrix based on the distance dis_dtw_1 with normalization
```

dis_dtw_2 *Constructs a pairwise distance matrix based on multivariate dynamic time warping*

Description

dis_dtw1 returns a pairwise distance matrix based on one of the multivariate extensions of the well-known dynamic time warping distance (Shokoohi-Yekta et al. 2017).

Usage

```
dis_dtw_2(X, normalization = FALSE, ...)
```

Arguments

X A list of MTS (numerical matrices).

normalization Logical. If normalization = TRUE (default), the normalized distance is computed. Otherwise (default), no normalization is taken into account

... Additional parameters for the function. See dtw.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS is defined as the multivariate extension of the dynamic time warping distance which forces all dimensions to warp identically, in a single warping matrix.

Value

The computed pairwise distance matrix.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Shokoohi-Yekta M, Hu B, Jin H, Wang J, Keogh E (2017). "Generalizing DTW to the multidimensional case requires an adaptive approach." *Data mining and knowledge discovery*, **31**(1), 1–31.

See Also

dis_dtw_2, dis_mahalanobis_dtw

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_dtw_2(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_dtw1 without normalization
distance_matrix_normalized <- dis_dtw_2(toy_dataset, normalization = TRUE) # Computing the pairwise
# distance matrix based on the distance dis_dtw1 with normalization
```

dis_eros *Constructs a pairwise distance matrix based on the Eros distance measure*

Description

dis_eros returns a pairwise distance matrix based on the Eros distance proposed by Yang and Shahabi (2004).

Usage

```
dis_eros(X, method = "mean", normalization = FALSE, cor = TRUE)
```

Arguments

X	A list of MTS (numerical matrices).
method	The aggregated function to compute the weights.
normalization	Logical indicating whether the raw eigenvalues or the normalized eigenvalues should be used to compute the weights. Default is FALSE, i.e., the raw eigenvalues are used.
cor	Logical indicating whether the Singular Value Decomposition is applied over the covariance matrix or over the correlation matrix. Default is TRUE, i.e., the correlation matrix is employed to avoid issues of scale.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as $d_{Eros}(\boldsymbol{X}_T, \boldsymbol{Y}_T) = \sqrt{2 - 2Eros(\boldsymbol{X}_T, \boldsymbol{Y}_T)}$, where

$$Eros(\boldsymbol{X}_T, \boldsymbol{Y}_T) = \sum_{i=1}^{d} w_i | < \boldsymbol{x}_i, \boldsymbol{y}_i > | = \sum_{i=1}^{d} w_i | \cos \theta_i |,$$

where $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_d\}$, $\{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_d\}$ are sets of eigenvectors concerning the covariance or correlation matrix of series $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, respectively, $< \boldsymbol{x}_i, \boldsymbol{y}_i >$ is the inner product of $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$, $\boldsymbol{w} = (w_1, \ldots, w_d)$ is a vector of weights which is based on the eigenvalues of the MTS dataset with $\sum_{i=1}^{d} w_i = 1$ and θ_i is the angle between $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$.

Value

The computed pairwise distance matrix.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Yang K, Shahabi C (2004). "A PCA-based similarity measure for multivariate time series." In *Proceedings of the 2nd ACM international workshop on Multimedia databases*, 65–74.

Examples

```
toy_dataset <- BasicMotions$data[1 : 10] # Selecting the first 10 MTS from the
# dataset BasicMotions
distance_matrix <- dis_eros(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_eros
distance_matrix <- dis_eros(toy_dataset, method = 'max', normalization = TRUE)
# Considering the function max as aggregation function and the normalized
# eigenvalues for the computation of the weights
```

`dis_eucl` *Constructs a pairwise distance matrix based on the Euclidean distance*

Description

`dis_eucl` returns a pairwise distance matrix based on the Euclidean distance between MTS

Usage

```
dis_eucl(X)
```

Arguments

`X`	A list of MTS (numerical matrices).

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS is defined as the sum of the standard Euclidean distances between each corresponding pair of dimensions (univariate time series)

Value

The computed pairwise distance matrix.

Author(s)

Ángel López-Oriona, José A. Vilar

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_eucl(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_eucl
```

`dis_frechet` *Constructs a pairwise distance matrix based on the Frechet distance*

Description

`dis_frechet` returns a pairwise distance matrix based on the Frechet distance between MTS

Usage

```
dis_frechet(X, ...)
```

Arguments

`X`	A list of MTS (numerical matrices).
`...`	Additional parameters for the function. See `diss.FRECHET`.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS is defined as the sum of the standard Frechet distances between each corresponding pair of dimensions (univariate time series)

Value

The computed pairwise distance matrix.

Author(s)

Ángel López-Oriona, José A. Vilar

See Also

diss.FRECHET

Examples

```
toy_dataset <- Libras$data[1 : 5] # Selecting the first 5 MTS from the
# dataset Libras
distance_matrix <- dis_frechet(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_frechet
```

---

dis_gcc *Constructs a pairwise distance matrix based on the generalized cross-correlation*

Description

dis_gcc returns a pairwise distance matrix based on the generalized cross-correlation measure introduced by Alonso and Pena (2019).

Usage

```
dis_gcc(X, lag_max = 1, features = FALSE)
```

Arguments

X	A list of MTS (numerical matrices).
lag_max	The maximum lag considered to compute the generalized cross-correlation.
features	Logical. If features = FALSE (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as

$$d_{GCC}(\boldsymbol{X}_T, \boldsymbol{Y}_T) = \left[\sum_{j_1, j_2 = 1, j_1 \neq j_2}^{d} \Big(GCC(\boldsymbol{X}_{T,j_1}, \boldsymbol{X}_{T,j_2}) - GCC(\boldsymbol{Y}_{T,j_1}, \boldsymbol{Y}_{T,j_2}) \Big)^2 \right]^{1/2},$$

where $\boldsymbol{X}_{T,j}$ and $\boldsymbol{Y}_{T,j}$ are the jth dimensions (univariate time series) of $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, respectively, and $GCC(\cdot, \cdot)$ is the genelarized cross-correlation measure between univariate series proposed by .

Value

If features = FALSE (default), returns a distance matrix based on the distance d_{GCC}. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute the distance d_{GCC}.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Alonso AM, Pena D (2019). “Clustering time series by linear dependency.” *Statistics and Computing*, **29**(4), 655–676.

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_gcc(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_cor
feature_dataset <- dis_gcc(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

dis_hwl *Constructs a pairwise distance matrix based on feature extraction*

Description

dis_hwl returns a pairwise distance matrix based on the feature extraction procedure proposed by Hyndman et al. (2015).

Usage

```
dis_hwl(X, features = FALSE)
```

Arguments

X	A list of MTS (numerical matrices).
features	Logical. If features = FALSE (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS is defined as the Euclidean distance between the corresponding feature vectors

Value

If features = FALSE (default), returns a distance matrix based on the distance d_{HWL}. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute the distance d_{HWL}.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Hyndman RJ, Wang E, Laptev N (2015). "Large-scale unusual time series detection." In *2015 IEEE international conference on data mining workshop (ICDMW)*, 1616–1619. IEEE.

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_hwl(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_hwl
#' feature_dataset <- dis_hwl(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

`dis_lpp` *Constructs a pairwise distance matrix based on locality preserving projections (LPP)*

Description

`dis_lpp` returns a pairwise distance matrix based on the dissimilarity introduced by Weng and Shen (2008).

Usage

```
dis_lpp(X, approach = 1, k = 2, t = 1, features = FALSE)
```

Arguments

`X`	A list of MTS (numerical matrices).
`approach`	Parameter indicating whether the feature vector representing each MTS is constructed by means of Li's first (`approach=1`, default) or Li's second (`approach=2`) approach.
`k`	Number of neighbors determining the construction of the local structure matrix $\boldsymbol{S}$.
`t`	Parameter determining the construction of the local structure matrix $\boldsymbol{S}$ (denominator in the exponential transformation).
`features`	Logical. If `features = FALSE` (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T^{(i)}$ and $\boldsymbol{X}_T^{(j)}$ is defined as

$$d_{LPP}(\boldsymbol{X}_T^{(i)}, \boldsymbol{X}_T^{(j)}) = \big|\big|\boldsymbol{\varphi}^{(i)}\boldsymbol{A}_{LPP} - \boldsymbol{\varphi}^{(j)}\boldsymbol{A}_{LPP}\big|\big|,$$

where $\boldsymbol{\varphi}^{(i)}$ and $\boldsymbol{\varphi}^{(j)}$ are the feature vectors constructed from Li's first (`approach=1`) or Li's second (`approach=2`) approach with respect to series $\boldsymbol{X}_T^{(i)}$ and $\boldsymbol{X}_T^{(j)}$, respectively and $\boldsymbol{A}_{LPP}$ is the matrix of locality preserving projections whose columns are eigenvectors solving the generalized eigenvalue problem defined by matrix $\boldsymbol{S}$.

Value

If `features = FALSE` (default), returns a distance matrix based on the distance d_{QCD}. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features resulting from applying Li's first (`approach=1`) or Li's second (`approach=2`).

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weng X, Shen J (2008). "Classification of multivariate time series using locality preserving projections." *Knowledge-Based Systems*, **21**(7), 581–587.

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_lpp(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_lpp
feature_dataset <- dis_lpp(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

`dis_mahalanobis` *Constructs a pairwise distance matrix based on the Mahalanobis distance*

Description

`dis_mahalanobis` returns a pairwise distance matrix based on the Mahalanobis divergence introduced by Singhal and Seborg (2005).

Usage

```
dis_mahalanobis(X)
```

Arguments

X A list of MTS (numerical matrices).

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as

$$d^*_{MD}(\boldsymbol{X}_T, \boldsymbol{Y}_T) = \frac{1}{2}\Big(d_{MD}(\boldsymbol{X}_T, \boldsymbol{Y}_T) + d_{MD}(\boldsymbol{Y}_T, \boldsymbol{X}_T)\Big),$$

with

$$d_{MD}(\boldsymbol{X}_T, \boldsymbol{Y}_T) = \sqrt{(\overline{\boldsymbol{X}}_T - \overline{\boldsymbol{Y}}_T)\boldsymbol{\Sigma}^{*-1}_{\boldsymbol{X}_T}(\overline{\boldsymbol{X}}_T - \overline{\boldsymbol{Y}}_T)^\top},$$

where $\overline{\boldsymbol{X}}_T$ and $\overline{\boldsymbol{Y}}_T$ are vectors containing the column-wise means concerning series $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, respectively, $\boldsymbol{\Sigma}_{\boldsymbol{X}_T}$ is the covariance matrix of $\boldsymbol{X}_T$ and $\boldsymbol{\Sigma}^{*-1}_{\boldsymbol{X}_T}$ is the pseudo-inverse of $\boldsymbol{\Sigma}_{\boldsymbol{X}_T}$ calculated using SVD. In the computation of d^*_{MD}, MTS $\boldsymbol{X}_T$ is assumed to be the reference series.

Value

The computed pairwise distance matrix.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Singhal A, Seborg DE (2005). “Clustering multivariate time-series data.” *Journal of Chemometrics: A Journal of the Chemometrics Society*, **19**(8), 427–438.

See Also

dis_mahalanobis_dtw

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_mahalanobis(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_mahalanobis.
```

dis_mahalanobis_dtw *Constructs a pairwise distance matrix based on a dissimilarity combining both the dynamic time warping and the Mahalanobis distance.*

Description

`dis_mahalanobis_dtw` returns a pairwise distance matrix based on a dynamic time warping distance in which the local cost matrix is computed by using the Mahalanobis distance (Mei et al. 2015).

Usage

```
dis_mahalanobis_dtw(X, M = NULL, ...)
```

Arguments

`X`	A list of MTS (numerical matrices).
`M`	The matrix with respect to compute the Mahalanobis distance (default is the covariance matrix of concatenation of all MTS objects by rows).
`...`	Additional parameters for the function. See `dtw`.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as a dynamic time warping-type distance in which the local cost matrix is constructed by using the Mahalanobis distance.

Value

The computed pairwise distance matrix.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Mei J, Liu M, Wang Y, Gao H (2015). "Learning a mahalanobis distance-based dynamic time warping measure for multivariate time series classification." *IEEE transactions on Cybernetics*, **46**(6), 1363–1374.

See Also

`dis_dtw_1`, `dis_dtw_2`, `dis_mahalanobis_dtw`

Examples

```
toy_dataset <- Libras$data[1 : 10] # Selecting the first 10 MTS from the
# dataset Libras
distance_matrix <- dis_mahalanobis_dtw(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_mahalanobis_dtw
```

`dis_modwt` *Constructs a pairwise distance matrix based on the maximum overlap discrete wavelet transform*

Description

`dis_modwt` returns a pairwise distance matrix based on the dissimilarity introduced by D'Urso and Maharaj (2012).

Usage

```
dis_modwt(X, wf = "d4", J = floor(log(nrow(X[[1]]))) - 1, features = FALSE)
```

Arguments

`X`	A list of MTS (numerical matrices).
`wf`	The wavelet filter (default is 'd4').
`J`	The maximum allowable number of scales.
`features`	Logical. If `features = FALSE` (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as

$$d_{MODWT}(\boldsymbol{X}_T, \boldsymbol{Y}_T) = \left| ||\widehat{\boldsymbol{\theta}}_{WV}^{\boldsymbol{X}_T} - \widehat{\boldsymbol{\theta}}_{WV}^{\boldsymbol{Y}_T}||^2 + ||\widehat{\boldsymbol{\theta}}_{WC}^{\boldsymbol{X}_T} - \widehat{\boldsymbol{\theta}}_{WC}^{\boldsymbol{Y}_T}||^2 \right|^{1/2},$$

where $\widehat{\boldsymbol{\theta}}_{WV}^{\boldsymbol{X}_T}$ and $\widehat{\boldsymbol{\theta}}_{WV}^{\boldsymbol{Y}_T}$ are vectors containing the estimated wavelet variances within $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, respectively, and $\widehat{\boldsymbol{\theta}}_{WC}^{\boldsymbol{X}_T}$ and $\widehat{\boldsymbol{\theta}}_{WC}^{\boldsymbol{Y}_T}$ are vectors containing the estimated wavelet correlations within $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, respectively.

Value

If `features = FALSE` (default), returns a distance matrix based on the distance d_{MODWT}. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute the distance d_{MODWT}.

Author(s)

Ángel López-Oriona, José A. Vilar

References

D’Urso P, Maharaj EA (2012). “Wavelets-based clustering of multivariate time series.” *Fuzzy Sets and Systems*, **193**, 33–61.

See Also

`modwt`

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_modwt(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_cor
feature_dataset <- dis_modwt(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

`dis_pca` *Constructs a pairwise distance matrix based on Principal Component Analysis (PCA)*

Description

`dis_eros` returns a pairwise distance matrix based on the PCA similarity factor proposed by Singhal and Seborg (2005).

Usage

```
dis_pca(X, retained_components = 3)
```

Arguments

`X`	A list of MTS (numerical matrices).
`retained_components`	Number of retained principal components.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as $d_{PCA}(\boldsymbol{X}_T, \boldsymbol{Y}_T) = 1 - S_{PCA}(\boldsymbol{X}_T, \boldsymbol{Y}_T)$, with

$$S_{PCA}(\boldsymbol{X}_T, \boldsymbol{Y}_T) = \frac{\sum_{i=1}^{k} \sum_{j=1}^{k} (\lambda^i_{\boldsymbol{X}_T} \lambda^j_{\boldsymbol{Y}_T}) \cos^2 \theta_{ij}}{\sum_{i=1}^{k} \lambda^i_{\boldsymbol{X}_T} \lambda^i_{\boldsymbol{Y}_T}},$$

where θ_{ij} is the angle between the ith eigenvector of $\boldsymbol{X}_T$ and the jth eigenvector of series $\boldsymbol{Y}_T$, respectively, and $\lambda^i_{\boldsymbol{Y}_T}$ and $\lambda^i_{\boldsymbol{Y}_T}$ are the ith eigenvalues of $\boldsymbol{X}_T$ and the jth eigenvalues of series $\boldsymbol{Y}_T$ respectively.

Value

The computed pairwise distance matrix.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Singhal A, Seborg DE (2005). "Clustering multivariate time-series data." *Journal of Chemometrics: A Journal of the Chemometrics Society*, **19**(8), 427–438.

Examples

```
toy_dataset <- BasicMotions$data[1 : 10] # Selecting the first 10 MTS from the
# dataset BasicMotions
distance_matrix <- dis_pca(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_pca
```

dis_qcd *Constructs a pairwise distance matrix based on the quantile cross-spectral density (QCD)*

Description

dis_qcd returns a pairwise distance matrix based on the dissimilarity introduced by Lopez-Oriona and Vilar (2021).

Usage

```
dis_qcd(X, levels = c(0.1, 0.5, 0.9), freq = NULL, features = FALSE, ...)
```

Arguments

X	A list of MTS (numerical matrices).
levels	The set of probability levels.
freq	Vector of frequencies in which the smoothed CCR-periodograms must be computed. If freq=NULL (default), the set of Fourier frequencies is considered.
features	Logical. If features = FALSE (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.
...	Additional parameters for the function. See smoothedPG.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as

$$d_{QCF}(\boldsymbol{X}_T, \boldsymbol{Y}_T) = \Bigg[\sum_{j_1=1}^{d} \sum_{j_2=1}^{d} \sum_{i=1}^{r} \sum_{i'=1}^{r} \sum_{k=1}^{K} \Big(\Re\big(\widehat{G}^{\boldsymbol{X}_T}_{j_1,j_2}(\omega_k, \tau_i, \tau_{i'})\big) - \Re\big(\widehat{G}^{\boldsymbol{Y}_T}_{j_1,j_2}(\omega_k, \tau_i, \tau_{i'})\big) \Big)^2 +$$

$$\sum_{j_1=1}^{d} \sum_{j_2=1}^{d} \sum_{i=1}^{r} \sum_{i'=1}^{r} \sum_{k=1}^{K} \Big(\Im\big(\widehat{G}^{\boldsymbol{X}_T}_{j_1,j_2}(\omega_k, \tau_i, \tau_{i'})\big) - \Im\big(\widehat{G}^{\boldsymbol{Y}_T}_{j_1,j_2}(\omega_k, \tau_i, \tau_{i'})\big) \Big)^2 \Bigg]^{1/2},$$

where $\widehat{G}^{\boldsymbol{X}_T}_{j_1,j_2}(\omega_k, \tau_i, \tau_{i'})$ and $\widehat{G}^{\boldsymbol{Y}_T}_{j_1,j_2}(\omega_k, \tau_i, \tau_{i'})$ are estimates of the quantile cross-spectral densities (so-called smoothed CCR-periodograms) with respect to the variables j_1 and j_2 and probability levels τ_i and $\tau_{i'}$ for series $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, respectively, and $\Re(\cdot)$ and $\Im(\cdot)$ denote the real part and imaginary part operators, respectively.

Value

If `features = FALSE` (default), returns a distance matrix based on the distance d_{QCD}. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute the distance d_{QCF}.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Lopez-Oriona A, Vilar JA (2021). "Quantile cross-spectral density: A novel and effective tool for clustering multivariate time series." *Expert Systems with Applications*, **185**, 115677.

See Also

`dis_qcf`

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 5] # Selecting the first 5 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_qcd(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_qcd
distance_matrix <- dis_qcd(toy_dataset, levels = c(0.4, 0.8)) # Changing
# the probability levels to compute the QCD-based estimators
distance_matrix <- dis_qcd(toy_dataset, freq = 0.5) # Considering only
# a single frequency for the computation of d_qcd
feature_dataset <- dis_qcd(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

dis_qcf *Constructs a pairwise distance matrix based on the quantile cross-covariance function*

Description

dis_qcf returns a pairwise distance matrix based on a generalization of the dissimilarity introduced by Lafuente-Rego and Vilar (2016).

Usage

```
dis_qcf(X, levels = c(0.1, 0.5, 0.9), max_lag = 1, features = FALSE)
```

Arguments

X	A list of MTS (numerical matrices).
levels	The set of probability levels.
max_lag	The maximum lag considered to compute the cross-covariances.
features	Logical. If features = FALSE (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as

$$d_{QCF}(\boldsymbol{X}_T, \boldsymbol{Y}_T) = \Bigg(\sum_{l=1}^{L} \sum_{i=1}^{r} \sum_{i'=1}^{r} \sum_{j_1=1}^{d} \sum_{j_2=1}^{d} \left(\widehat{\gamma}_{j_1,j_2}^{\boldsymbol{X}_T}(l, \tau_i, \tau_{i'}) - \widehat{\gamma}_{j_1,j_2}^{\boldsymbol{Y}_T}(l, \tau_i, \tau_{i'}) \right)^2 +$$

$$\sum_{i=1}^{r} \sum_{i'=1}^{r} \sum_{j_1,j_2=1:j_1>j_2}^{d} \left(\widehat{\gamma}_{j_1,j_2}^{\boldsymbol{X}_T}(0, \tau_i, \tau_{i'}) - \widehat{\gamma}_{j_1,j_2}^{\boldsymbol{Y}_T}(0, \tau_i, \tau_{i'}) \right)^2 \Bigg]^{1/2},$$

where $\widehat{\gamma}_{j_1,j_2}^{\boldsymbol{X}_T}(l, \tau_i, \tau_{i'})$ and $\widehat{\gamma}_{j_1,j_2}^{\boldsymbol{Y}_T}(l, \tau_i, \tau_{i'})$ are estimates of the quantile cross-covariances with respect to the variables j_1 and j_2 and probability levels τ_i and $\tau_{i'}$ for series $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, respectively.

Value

If features = FALSE (default), returns a distance matrix based on the distance d_{QCF}. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute the distance d_{QCF}.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Lafuente-Rego B, Vilar JA (2016). "Clustering of time series using quantile autocovariances." *Advances in Data Analysis and classification*, **10**(3), 391–415.

See Also

dis_qcd

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_qcf(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_qcf
feature_dataset <- dis_qcf(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

dis_spectral *Constructs a pairwise distance matrix based on estimated spectral matrices*

Description

dis_spectral returns a pairwise distance matrix based on the dissimilarities introduced by Kakizawa et al. (1998).

Usage

```
dis_spectral(X, method = "j_divergence", alpha = 0.5, features = FALSE)
```

Arguments

X	A list of MTS (numerical matrices).
method	Parameter indicating the method to be used for the computation of the distance. If method="j_divergence" (default), the J divergence is considered. If method="chernoff_divergence", the Chernoff information divergence is considered
alpha	If method="chernoff_divergence", parameter alpha in (0,1) used for the computation of the Chernoff divergence (default is 0.5).
features	Logical. If features = FALSE (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns a pairwise distance matrix. If `method="j_divergence"` then the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as

$$d_{JSPEC}(\boldsymbol{X}_T,\boldsymbol{Y}_T)=\frac{1}{2T}\sum_{k=1}^{K}\Big(tr\Big(\widehat{\boldsymbol{f}}_{\boldsymbol{X}_T}(\omega_k)\widehat{\boldsymbol{f}}^{-1}_{\boldsymbol{Y}_T}(\omega_k)\Big)+tr\Big(\widehat{\boldsymbol{f}}_{\boldsymbol{Y}_T}(\omega_k)\widehat{\boldsymbol{f}}^{-1}_{\boldsymbol{X}_T}(\omega_k)\Big)-2d\Big),$$

where $\widehat{\boldsymbol{f}}_{\boldsymbol{X}_T}(\omega_k)$ and $\widehat{\boldsymbol{f}}_{\boldsymbol{Y}_T}(\omega_k)$ are the estimated spectral density matrices from the series $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, respectively, evaluated at frequency ω_k, and $tr(\cdot)$ denotes the trace of a square matrix. If `method="chernoff_divergence"`, then the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as

$$d_{CSPEC}(\boldsymbol{X}_T,\boldsymbol{Y}_T)=$$

$$\frac{1}{2T}\sum_{k=1}^{K}\left(\log\frac{\left|\alpha\widehat{\boldsymbol{f}}^{\boldsymbol{X}_T}(\omega_k)+(1-\alpha)\widehat{\boldsymbol{f}}^{\boldsymbol{Y}_T}(\omega_k)\right|}{\left|\widehat{\boldsymbol{f}}^{\boldsymbol{Y}_T}(\omega_k)\right|}+\log\frac{\left|\alpha\widehat{\boldsymbol{f}}^{\boldsymbol{Y}_T}(\omega_k)+(1-\alpha)\widehat{\boldsymbol{f}}^{\boldsymbol{X}_T}(\omega_k)\right|}{\left|\widehat{\boldsymbol{f}}^{\boldsymbol{X}_T}(\omega_k)\right|}\right),$$

where $\alpha\in(0,1)$.

Value

If `features = FALSE` (default), returns a distance matrix based on the distance d_{JSPEC} if `method="j_divergence"`, and based on the distance d_{CSPEC} if `method="chernoff_divergence"`. Otherwise, if `features = TRUE`, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute either d_{JSPEC} or d_{CSPEC}. These vectors are vectorized versions of the estimated spectral matrices.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Kakizawa Y, Shumway RH, Taniguchi M (1998). "Discrimination and clustering for multivariate time series." *Journal of the American Statistical Association*, **93**(441), 328–340.

Examples

```
toy_dataset <- Libras$data[1 : 10] # Selecting the first 10 MTS from the
# dataset Libras
distance_matrix_j <- dis_spectral(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_jspec
distance_matrix_c <- dis_spectral(toy_dataset,
method = 'chernoff_divergence') # Computing the pairwise
# distance matrix based on the distance dis_cspec
feature_dataset <- dis_qcd(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features for d_cpec
```

dis_swmd *Constructs a pairwise distance matrix based on VPCA and SWMD*

Description

`dis_swmd` returns a pairwise distance matrix based on variable-based principal component analysis (VPCA) and a spatial weighted matrix distance (SWMD) (He and Tan 2018).

Usage

```
dis_swmd(X, var_rate = 0.9, features = FALSE)
```

Arguments

`X`	A list of MTS (numerical matrices).
`var_rate`	Rate of retained variability concerning the dimensionality-reduced MTS samples (default is 0.90).
`features`	Logical. If `features = FALSE` (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T^{(i)}$ and $\boldsymbol{X}_T^{(j)}$ is defined as

$$d_{SWMD}(\boldsymbol{X}_T^{(i)}, \boldsymbol{X}_T^{(j)}) = \Big[\big(vec(\boldsymbol{Y}_i) - vec(\boldsymbol{Y}_j)\big)\boldsymbol{S}\big(vec(\boldsymbol{Y}_i) - vec(\boldsymbol{Y}_j)\big)^\top\Big]^{1/2},$$

where $\boldsymbol{Y}_i$ and $\boldsymbol{Y}_j$ are the dimensionality- reduced MTS samples associated with $\boldsymbol{X}_T^{(i)}$ and $\boldsymbol{X}_T^{(j)}$, respectively, the operator $vec(\cdot)$ creates a vector by concatenating the columns of the matrix received as input and $\boldsymbol{S}$ is a matrix integrating the spatial dimensionality difference between the corresponding elements.

Value

If `features = FALSE` (default), returns a distance matrix based on the distance d_{SWMD}. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute the distance d_{SWMD}.

Author(s)

Ángel López-Oriona, José A. Vilar

References

He H, Tan Y (2018). “Unsupervised classification of multivariate time series using VPCA and fuzzy clustering with spatial weighted matrix distance.” *IEEE transactions on cybernetics*, **50**(3), 1096–1105.

See Also

vpca_clustering

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_swmd(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_swmd
feature_dataset <- dis_swmd(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

---

dis_var_1 *Constructs a pairwise distance matrix based on the estimated VAR coefficients of the series*

Description

dis_cor returns a pairwise distance matrix based on a generalization of the dissimilarity introduced by Piccolo (1990).

Usage

```
dis_var_1(X, max_p = 1, criterion = "AIC", features = FALSE)
```

Arguments

X	A list of MTS (numerical matrices).
max_p	The maximum order considered with respect to the fitting of VAR models.
criterion	The criterion used to determine the VAR order.
features	Logical. If features = FALSE (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as

$$d_{VAR}(\boldsymbol{X}_T, \boldsymbol{Y}_T) = ||\widehat{\boldsymbol{\theta}}^{\boldsymbol{X}_T}_{VAR} - \widehat{\boldsymbol{\theta}}^{\boldsymbol{Y}_T}_{VAR}||,$$

where $\widehat{\boldsymbol{\theta}}^{\boldsymbol{X}_T}_{VAR}$ and $\widehat{\boldsymbol{\theta}}^{\boldsymbol{Y}_T}_{VAR}$ are vectors containing the estimated VAR parameters for $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, respectively. If VAR models of different orders are fitted to $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$, then the shortest vector is padded with zeros until it reaches the length of the longest vector.

Value

If features = FALSE (default), returns a distance matrix based on the distance d_{COR}. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute the distance d_{VAR}.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Piccolo D (1990). "A distance measure for classifying ARIMA models." *Journal of time series analysis*, **11**(2), 153–164.

See Also

dis_var_2, diss.AR.PIC

Examples

```
toy_dataset <- Libras$data[1 : 2] # Selecting the first 2 MTS from the
# dataset Libras
distance_matrix <- dis_var_1(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_var_1
feature_dataset <- dis_var_1(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

---

dis_var_2 *Model-based dissimilarity proposed by Maharaj (1999)*

Description

dis_var_2 returns a pairwise distance matrix based on testing whether each pair of series are or not generated from the same VARMA model (Maharaj 1999).

Usage

```
dis_var_2(X, max_p = 2, criterion = "BIC")
```

Arguments

X	A list of MTS (numerical matrices).
max_p	The maximum order considered with respect to the fitting of VAR models.
criterion	The criterion used to determine the VAR order.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS $\boldsymbol{X}_T$ and $\boldsymbol{Y}_T$ is defined as $1 - p$, where p is the p-value of the test of hypothesis proposed by . This test is based on checking the equality of the underlying VARMA models of both series. The VARMA structures are approximated by truncated VAR(∞) models with a common order $k = \max(k_x, k_y)$, where k_x and k_y are determined by the BIC or AIC criterion. The VAR coefficients are automatically fitted. The dissimilarity between both series is given by $1-p$ because this quantity is expected to take larger values the more different both generating processes are. The procedure is able to compare two dependent MTS.

Value

The computed pairwise distance matrix.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Maharaj EA (1999). "Comparison and classification of stationary multivariate time series." *Pattern Recognition*, **32**(7), 1129–1138.

See Also

dis_var_1, diss.AR.MAH

Examples

```
toy_dataset <- Libras$data[c(1, 2)] # Selecting the first two MTS from the
# dataset Libras
distance_matrix <- dis_var_2(toy_dataset, max_p = 1) # Computing the pairwise
# distance matrix based on the distance dis_var_2
```

---

dis_www *Constructs a pairwise distance matrix based on feature extraction*

Description

dis_www returns a pairwise distance matrix based on the feature extraction procedure proposed by Wang et al. (2007).

Usage

```
dis_www(X, h = 20, features = FALSE)
```

Arguments

X	A list of MTS (numerical matrices).
h	Maximum lag for the computation of the Box-Pierce statistic.
features	Logical. If features = FALSE (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS is defined as the Euclidean distance between the corresponding feature vectors

Value

If `features = FALSE` (default), returns a distance matrix based on the distance d_{WWW}. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute the distance d_{WWW}.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Wang X, Wirth A, Wang L (2007). "Structure-based statistical features and multivariate time series clustering." In *Seventh IEEE international conference on data mining (ICDM 2007)*, 351–360. IEEE.

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_www(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_www
feature_dataset <- dis_www(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

`dis_zagorecki` *Constructs a pairwise distance matrix based on feature extraction*

Description

`dis_zagorecki` returns a pairwise distance matrix based on the feature extraction procedure proposed by Zagorecki (2015).

Usage

```
dis_zagorecki(set, features = FALSE)
```

Arguments

`set`	A list of MTS (numerical matrices).
`features`	Logical. If `features = FALSE` (default), a distance matrix is returned. Otherwise, the function returns a dataset of feature vectors.

Details

Given a collection of MTS, the function returns the pairwise distance matrix, where the distance between two MTS is defined as the Euclidean distance between the corresponding feature vectors

Value

If `features` = `FALSE` (default), returns a distance matrix based on the distance $d_{ZAGORECKI}$. Otherwise, the function returns a dataset of feature vectors, i.e., each row in the dataset contains the features employed to compute the distance $d_{ZAGORECKI}$.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Zagorecki A (2015). "A versatile approach to classification of multivariate time series data." In *2015 Federated Conference on Computer Science and Information Systems (FedCSIS)*, 407–410. IEEE.

Examples

```
toy_dataset <- AtrialFibrillation$data[1 : 10] # Selecting the first 10 MTS from the
# dataset AtrialFibrillation
distance_matrix <- dis_zagorecki(toy_dataset) # Computing the pairwise
# distance matrix based on the distance dis_zagorecki
feature_dataset <- dis_zagorecki(toy_dataset, features = TRUE) # Computing
# the corresponding dataset of features
```

`Epilepsy` *Epilepsy*

Description

Multivariate time series (MTS) of some participants simulating several activities. In particular, data was collected from 6 participants using a tri-axial accelerometer on the dominant wrist while conducting 4 different activities

Usage

```
data(Epilepsy)
```

Format

A `list` with two elements, which are:

- `data` A list with 275 MTS.
- `classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a matrix formed by 206 rows (time points) indicating acceleration trajectory and 3 columns (variables) indicating the axis in the accelerometer. The first 137 elements correspond to the training set, whereas the last 138 elements correspond to the test set. The numeric vector `classes` is formed by integers from 1 to 4, indicating that there are 4 different classes in the database. Each class is associated with a different activity. For more information, see Bagnall et al. (2018).

References

Bagnall A, Dau HA, Lines J, Flynn M, Large J, Bostrom A, Southam P, Keogh E (2018). "The UEA multivariate time series classification archive, 2018." *arXiv preprint arXiv:1811.00075*.

Ruiz AP, Flynn M, Large J, Middlehurst M, Bagnall A (2021). "The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances." *Data Mining and Knowledge Discovery*, **35**(2), 401–449.

Bagnall A, Lines J, Vickers W, Keogh E (2022). "The UEA & UCR Time Series Classification Repository." *www.timeseriesclassification.com*.

`ERing` *ERing*

Description

Multivariate time series (MTS) indicating the movement of the worm Caenorhabditis elegans. The motion of worms in an agar plate is recorded as a combination of six base shapes.

Usage

```
data(ERing)
```

Format

A `list` with two elements, which are:

`data` A list with 60 MTS.

`classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a matrix formed by 65 rows (time points) indicating time measurements and 4 columns (variables) indicating electrodes. The first 30 elements correspond to the training set, whereas the last 30 elements correspond to the test set. The numeric vector `classes` is formed by integers from 1 to 6, indicating that there are 6 different classes in the database. Each class is associated with a different posture of the hand. For more information, see Bagnall et al. (2018).

References

Bagnall A, Dau HA, Lines J, Flynn M, Large J, Bostrom A, Southam P, Keogh E (2018). “The UEA multivariate time series classification archive, 2018.” *arXiv preprint arXiv:1811.00075*.

Ruiz AP, Flynn M, Large J, Middlehurst M, Bagnall A (2021). “The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances.” *Data Mining and Knowledge Discovery*, **35**(2), 401–449.

Bagnall A, Lines J, Vickers W, Keogh E (2022). “The UEA & UCR Time Series Classification Repository.” *www.timeseriesclassification.com*.

`f4_classifier` *Constructs the F4 classifier of López-Oriona and Vilar (2021)*

Description

`f4_classifier` computes the F4 classifier for MTS proposed by Lopez-Oriona and Vilar (2021).

Usage

```
f4_classifier(
  training_data,
  new_data = NULL,
  classes,
  levels = c(0.1, 0.5, 0.9),
  cv_folds = 5,
  var_rate = 0.9
)
```

Arguments

`training_data` A list of MTS constituting the training set to fit classifier F4.

`new_data` A list of MTS for which the class labels have to be predicted.

`classes` A vector containing the class labels associated with the elements in `training_data`.

`levels` The set of probability levels to compute the QCD-estimates.

`cv_folds` The number of folds concerning the cross-validation procedure used to fit F4 with respect to `training_data`.

`var_rate` Rate of desired variability to select the principal components associated with the QCD-based features.

Details

This function constructs the classifier F4 of . Given a set of MTS with associated class labels, estimates of the quantile cross-spectral density (QCD) and the maximum overlap discrete wavelet transform (MODWT) are first computed for each series. Then Principal Components Analysis (PCA) is applied over the dataset of QCD-based features and a given number of principal components are retained according to a criterion of explained variability. Next, each series is decribed by means of the concatenation of the QCD-based transformed features and the MODWT-based features. Finally, a traditional random forest classifier is executed in the resulting dataset.

Value

If new_data = NULL (default), returns a fitted model of class train (see train). Otherwise, the function returns the predicted class labels for the elements in new_data.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Lopez-Oriona A, Vilar JA (2021). "F4: An All-Purpose Tool for Multivariate Time Series Classification." *Mathematics*, **9**(23), 3051.

Examples

```
predictions <- f4_classifier(training_data = Libras$data[1 : 20],
new_data = Libras$data[181 : 200], classes = Libras$classes[181 : 200])
# Computing the predictions for the test set of dataset Libras
```

FinancialData *FinancialData*

Description

Dataset containing 50 financial MTS associated with companies in the S&P 500 index.

Usage

```
data(FinancialData)
```

Format

A list with two elements, which are:

data A list with 50 MTS.

classes A character vector indicating the abbreviations associated with the series (companies) in data.

Details

Each element in data is a matrix formed by 654 rows (series length) and 2 columns (dimensions). Each MTS represents a company in the top 50 of the S&P 500 index according to market capitalization. One dimension measures the daily returns of the company, whereas the other measures the daily change in trading volume. The sample period spans from 6th July 2015 to 7th February 2018.

`knn_classifier` *Constructs a nearest neighbours-based classifier and returns the predictions for a test set*

Description

`knn_classifier` returns the predictions for a test set concerning a nearest neighbours-based classifier.

Usage

```
knn_classifier(dataset, classes, index_test, distance, k, ...)
```

Arguments

`dataset`	A list of MTS (numerical matrices).
`classes`	A vector containing the class labels associated with the elements in `dataset`.
`index_test`	The indexes associated with the test elements in `dataset`, i.e., the elements for which predictions will be computed.
`distance`	The corresponding distance measure to compute the nearest neighbours-based classifier (must be one the functions implemented in **mlmts**, as a string).
`k`	The number of neighbours.
`...`	Additional parameters for the function with respect to the considered distance.

Details

Given a collection of MTS containing the training and test set, the function constructs a nearest neighbours-based classifier based on a given dissimilarity measure. The corresponding predictions for the elements in the test set are returned.

Value

The class labels for the elements in the test set.

Author(s)

Ángel López-Oriona, José A. Vilar

Examples

```
predictions_1_nn <- knn_classifier(BasicMotions$data[1 : 10], BasicMotions$classes[1 : 10],
index_test = 6 : 10, distance = 'dis_modwt', k = 1) # Computing the
# predictions for the test elements in dataset BasicMotions according to
# a 1-nearest neighbour classifier based on dis_modtw.
predictions_1_nn
```

`Libras` *Libras*

Description

Multivariate time series (MTS) indicating hand movement concerning the official brazilian sign language from 4 different people, during 2 sessions.

Usage

```
data(Libras)
```

Format

A `list` with two elements, which are:

- `data` A list with 360 MTS.
- `classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a matrix formed by 45 rows (time points) indicating time points in video recordings and 2 columns (variables) indicating video sessions. The first 180 elements correspond to the training set, whereas the last 180 elements correspond to the test set. The numeric vector `classes` is formed by integers from 1 to 15, indicating that there are 15 different classes in the database. Each class is associated with a hand movement type. For more information, see Bagnall et al. (2018).

References

Bagnall A, Dau HA, Lines J, Flynn M, Large J, Bostrom A, Southam P, Keogh E (2018). “The UEA multivariate time series classification archive, 2018.” *arXiv preprint arXiv:1811.00075*.

Ruiz AP, Flynn M, Large J, Middlehurst M, Bagnall A (2021). “The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances.” *Data Mining and Knowledge Discovery*, **35**(2), 401–449.

Bagnall A, Lines J, Vickers W, Keogh E (2022). “The UEA & UCR Time Series Classification Repository.” *www.timeseriesclassification.com*.

mc2pca_clustering *Performs the crisp clustering algorithm of Li (2019)*

Description

mc2pca_clustering performs the clustering algorithm proposed by Li (2019), which is based on common principal component analysis (CPCA).

Usage

```
mc2pca_clustering(X, k, var_rate = 0.9, max_it = 1000, tol = 1e-05)
```

Arguments

X	A list of MTS (numerical matrices).
k	The number of clusters.
var_rate	Rate of retained variability concerning the reconstructed MTS samples (default is 0.90).
max_it	The maximum number of iterations (default is 1000).
tol	The tolerance (default is 1e-5).

Details

This function executes the crisp clustering method proposed by . The algorithm is a K-means-type procedure where the distance between a given MTS and a centroid is given by the reconstruction error taking place when the series is reconstructed from the common space obtained by considering all the series in the cluster associated with the corresponding centroid (the common space is the centroid).

Value

A list with two elements:

- cluster. A vector defining the clustering solution.
- iterations. The number of iterations before the algorithm stopped.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Li H (2019). "Multivariate time series clustering based on common principal component analysis." *Neurocomputing*, **349**, 239–247.

Examples

```
clustering_algorithm <- mc2pca_clustering(BasicMotions$data, k = 4, var_rate = 0.30)
# Executing the clustering algorithm in the dataset BasicMotions (var_rate = 0.30,
# i.e., we keep only a few principal components for computing the reconstructed series)
clustering_algorithm$cluster # The clustering solution
clustering_algorithm$iterations # The number of iterations before the algorithm
library(ClusterR)
external_validation(clustering_algorithm$cluster, BasicMotions$classes,
summary_stats = TRUE) # Evaluating the clustering algorithms vs the true partition
# stopped
```

`mlmts` *mlmts: Machine Learning Algorithms for Multivariate Time Series.*

Description

mlmts provides an implementation of several machine learning algorithms for multivariate time series. The package includes functions allowing the execution of clustering, classification or outlier detection methods, among others. It also incorporates a collection of multivariate time series datasets which can be used to analyse the performance of new proposed algorithms. Practitioners from a broad variety of fields could benefit from the general framework provided by mlmts.

`mts_plot` *Constructs a plot of a MTS*

Description

`mts_plot` constructs a plot of a MTS. Each univariate series comprising the MTS object is displayed in a different colour.

Usage

```
mts_plot(series, title = "")
```

Arguments

`series`	A MTS (numerical matrix).
`title`	Title for the plot (string). Default corresponds to no title.

Details

Given a MTS, the function constructs the corresponding plot, in which a different colour is used for each univariate series comprising the MTS object. Therefore, the MTS is represented as a collection of univariate series in a single graph.

Value

The corresponding plot.

Author(s)

Ángel López-Oriona, José A. Vilar

Examples

```
mts_plot(BasicMotions$data[[1]]) # Represents the first MTS in dataset
# BasicMotions
```

`outlier_detection` *Constructs the outlier detection procedure of López-Oriona and Vilar (2021)*

Description

`outlier_detection` computes the outlier detection method for MTS proposed by Lopez-Oriona and Vilar (2021).

Usage

```
outlier_detection(X, levels = c(0.1, 0.5, 0.9), alpha = NULL)
```

Arguments

`X`	A list of MTS (numerical matrices).
`levels`	The set of probability levels to compute the QCD-estimates.
`alpha`	The desired rate of outliers to detect (a real number between 0 and 1).

Details

This function performs outlier detection according to the procedure proposed by . Specifically, each MTS in the original set is described by means of a multivariate functional datum by using an estimate of its quantile cross- spectral density. Given the corresponding set of multivariate functional data, the functional depth of each object is computed. Based on depth computations, the outlying elements are the objects with low values for the depths.

Value

A list with two elements:

- `Depths`. The functional depths associated with elements in `X`, sorted in increasing order.
- `Indexes`. The corresponding indexes associated with the vector `Depths`.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Lopez-Oriona A, Vilar JA (2021). "Outlier detection for multivariate time series: A functional data approach." *Knowledge-Based Systems*, **233**, 107527.

See Also

dis_qcd

Examples

```
outliers <- outlier_detection(SyntheticData2$data[c(1 : 3, 65)])
outliers$Indexes[1] # The first outlying MTS in dataset SyntheticData2
outliers$Depths[1] # The corresponding value for the depths
```

PenDigits *PenDigits*

Description

Multivariate time series (MTS) indicating writing of 44 people drawing the digits from 0 to 9. Each instance is made up of the x and y coordinates of the pen-tip traced accross a digital screen.

Usage

```
data(PenDigits)
```

Format

A `list` with two elements, which are:

- `data` A list with 10992 MTS.
- `classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a matrix formed by 8 rows (time points) spatial points and 2 columns (variables) indicating coordinate. The first 7494 elements correspond to the training set, whereas the last 3498 elements correspond to the test set. The numeric vector `classes` is formed by integers from 1 to 10, indicating that there are 10 different classes in the database. Each class is associated with a different digit. For more information, see Bagnall et al. (2018).

References

Bagnall A, Dau HA, Lines J, Flynn M, Large J, Bostrom A, Southam P, Keogh E (2018). "The UEA multivariate time series classification archive, 2018." *arXiv preprint arXiv:1811.00075*.

Ruiz AP, Flynn M, Large J, Middlehurst M, Bagnall A (2021). "The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances." *Data Mining and Knowledge Discovery*, **35**(2), 401–449.

Bagnall A, Lines J, Vickers W, Keogh E (2022). "The UEA & UCR Time Series Classification Repository." *www.timeseriesclassification.com*.

`plot_2d_scaling` *Constructs a 2-dimensional scaling plot based on a given dissimilarity matrix.*

Description

`plot_2d_scaling` represents a 2-dimensional scaling plane starting from a dissimilarity matrix.

Usage

```
plot_2d_scaling(distance_matrix, cluster_labels = NULL, title = "")
```

Arguments

`distance_matrix`
A distance matrix.

`cluster_labels` The labels associated with the elements involving the entries in `distance_matrix`. The points in the plot are coloured according to these labels. If no labels are provided (default), all points are represented in the same colour.

`title` The title of the graph (default is no title).

Details

Given a distance matrix, the function constructs the corresponding 2-dimensional scaling, which is a 2d plane in which the distances between the points represent the original distances as correctly as possible. If the vector `cluster_labels` is provided to the function, points in the 2d plane are coloured according to the given class labels.

Value

The 2-dimensional scaling plane.

Author(s)

Ángel López-Oriona, José A. Vilar

Examples

```
distance_matrix_qcd <- dis_qcd(SyntheticData1$data[1 : 30]) # Computing the pairwise
# distance matrix for the first 30 elements in dataset SyntheticData1 based on dis_qcd
plot_2d_scaling(distance_matrix_qcd, cluster_labels = SyntheticData1$classes)
# Constructing the corresponding 2d-scaling plot. Each class is represented
# in a different colour
```

RacketSports *RacketSports*

Description

Multivariate time series (MTS) collected from university students playing badminton or squash while wearing a smartwatch. The watch recorded the x, y, z coordinates for both a gyroscope and an accelerometer to an android phone.

Usage

```
data(RacketSports)
```

Format

A `list` with two elements, which are:

`data` A list with 303 MTS.

`classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a matrix formed by 30 rows (time points) indicating time recordings over an interval of 3 seconds and 6 columns (variables) indicating gyroscope or accelerometer and the corresponding coordinate. The first 151 elements correspond to the training set, whereas the last 152 elements correspond to the test set. The numeric vector `classes` is formed by integers from 1 to 4, indicating that there are 4 different classes in the database. Each class is associated with a sport and stroke a particular player is making. For more information, see Bagnall et al. (2018).

References

Bagnall A, Dau HA, Lines J, Flynn M, Large J, Bostrom A, Southam P, Keogh E (2018). “The UEA multivariate time series classification archive, 2018.” *arXiv preprint arXiv:1811.00075*.

Ruiz AP, Flynn M, Large J, Middlehurst M, Bagnall A (2021). “The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances.” *Data Mining and Knowledge Discovery*, **35**(2), 401–449.

Bagnall A, Lines J, Vickers W, Keogh E (2022). “The UEA & UCR Time Series Classification Repository.” *www.timeseriesclassification.com*.

`StandWalkJump` *StandWalkJump*

Description

Multivariate time series (MTS) involving short duration ECG signals recorded from a healthy 25-year-old male performing different physical activities

Usage

```
data(StandWalkJump)
```

Format

A `list` with two elements, which are:

`data` A list with 27 MTS.

`classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a matrix formed by 2500 rows (time points) indicating readings in a spectrogram and 4 columns (variables) indicating frequency band from the spectrogram. The first 12 elements correspond to the training set, whereas the last 15 elements correspond to the test set. The numeric vector `classes` is formed by integers from 1 to 3, indicating that there are 3 different classes in the database. Each class is associated with the label 'standing', 'walking' or 'jumping'. For more information, see Bagnall et al. (2018).

References

Bagnall A, Dau HA, Lines J, Flynn M, Large J, Bostrom A, Southam P, Keogh E (2018). "The UEA multivariate time series classification archive, 2018." *arXiv preprint arXiv:1811.00075*.

Ruiz AP, Flynn M, Large J, Middlehurst M, Bagnall A (2021). "The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances." *Data Mining and Knowledge Discovery*, **35**(2), 401–449.

Bagnall A, Lines J, Vickers W, Keogh E (2022). "The UEA & UCR Time Series Classification Repository." *www.timeseriesclassification.com*.

SyntheticData1 *SyntheticData1*

Description

Synthetic dataset containing 60 MTS generated from four different generating processes.

Usage

```
data(SyntheticData1)
```

Format

A `list` with two elements, which are:

`data` A list with 60 MTS.

`classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a matrix formed by 400 rows (series length) and 2 columns (dimensions). Series 1-15 were generated from a VAR(1) process and series 16-30 were generated from a VMA(1) process. Series 31-45 were generated from a QVAR(1) process and series 46-60 were generated from a different QVAR(1) process. Therefore, there are 4 different classes in the dataset.

SyntheticData2 *SyntheticData2*

Description

Synthetic dataset containing 65 MTS generated from five different generating processes.

Usage

```
data(SyntheticData1)
```

Format

A `list` with two elements, which are:

`data` A list with 65 MTS.

`classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in data is a matrix formed by 400 rows (series length) and 2 columns (dimensions). Series 1-15 were generated from a VAR(1) process and series 16-30 were generated from a VMA(1) process. Series 31-45 were generated from a QVAR(1) process and series 46-60 were generated from a different QVAR(1) process. Finally, series 61-65 were generated from a VAR(1) model different from the one associated with series 1-15. Note that series 61-65 can be seen as anomalous elements in the dataset.

---

vpca_clustering *Performs the fuzzy clustering algorithm of He and Tan (2020).*

Description

vpca_clustering performs the fuzzy clustering algorithm proposed by He and Tan (2018).

Usage

```
vpca_clustering(
  X,
  k,
  m,
  var_rate = 0.9,
  max_it = 1000,
  tol = 1e-05,
  crisp = FALSE
)
```

Arguments

X — A list of MTS (numerical matrices).

k — The number of clusters.

m — The fuzziness coefficient (a real number greater than one).

var_rate — Rate of retained variability concerning the dimensionality-reduced MTS samples (default is 0.90).

max_it — The maximum number of iterations (default is 1000).

tol — The tolerance (default is 1e-5).

crisp — Logical. If crisp = FALSE (default) a fuzzy partition is returned. Otherwise, the function returns the corresponding crisp partition, in which each series is placed in the cluster associated with the maximum membership degree.

Details

This function executes the fuzzy clustering procedure proposed by . The algorithm represents each MTS in the original collection by means of a dimensionality-reduced MTS constructed through variable-based principal component analysis (VPCA). Then, fuzzy K-means-type procedure is considered for the set of dimensionalityu-reduced samples. A spatial weighted matrix dissimilarity is considered to compute the distances between the reduced MTS and the centroids.

Value

A list with three elements:

- `U`. If `crisp` = `FALSE` (default), the membership matrix. Otherwise, a vector defining the corresponding crisp partition.
- `centroids`. If `crisp` = `FALSE` (default), a list containing the centroid series, which are dimensionality-reduced averaged MTS. Otherwise, this element is not returned.
- `iterations`. The number of iterations before the algorithm stopped.

Author(s)

Ángel López-Oriona, José A. Vilar

References

He H, Tan Y (2018). "Unsupervised classification of multivariate time series using VPCA and fuzzy clustering with spatial weighted matrix distance." *IEEE transactions on cybernetics*, **50**(3), 1096–1105.

See Also

`vpca_clustering`

Examples

```
fuzzy_clustering <- vpca_clustering(AtrialFibrillation$data, k = 3, m = 1.5)
# Executing the fuzzy clustering algorithm in the dataset AtrialFibrillation
# by considering 3 clusters and a value of 1.5 for the fuziness parameter
fuzzy_clustering$U # The membership matrix
crisp_clustering <- vpca_clustering(AtrialFibrillation$data, k = 3, m = 1.5, crisp = TRUE)
# The same as before, but we are interested in the corresponding crisp partition
crisp_clustering$U # The crisp partition
crisp_clustering$iterations # The number of iterations before the algorithm
# stopped
```

Index