Package ‘rEDM’

December 1, 2019

Type Package

Title Empirical Dynamic Modeling (EDM)

Version 1.0.1

Date 2019-12-1

Maintainer Joseph Park <JosephPark@IEEE.org>

Description An implementation of EDM algorithms based on research software developed for internal use at the Sugihara Lab (UCSD/SIO). The package is implemented with Rcpp wrappers around the cppEDM library. It supports both the simplex projection method from Sugihara & May (1990) and the S-map algorithm in Sugihara (1994). In addition, it implements convergent cross mapping as described in Sugihara et al. (2012) and multiview embedding as described in Ye & Sugihara (2016).

Note Version 1.0.1 is a major rewrite of the rEDM package as a wrapper for the cppEDM library. This allows a unified computation engine for EDM algorithms across C++, Python and R implementations. The cppEDM is written in a functional programming paradigm, facilitating clarity and ease of understanding to foster collaboration and extension. Additionally, cppEDM is intended for computationally intensive application, such as supercomputers including the AI Bridging Cloud Infrastructure (ABCI) where it has been applied to massively parallel neuronal timeseries cross mapping. The new package provides improved alignment between observed and forecast data rows, and, strict exclusion of partial data vectors.

License file LICENSE

LazyData true

LazyLoad yes

Imports methods, Rcpp (>= 1.0.1)

LinkingTo Rcpp

Suggests knitr, rmarkdown

VignetteBuilder knitr

R topics documented:

`block_3sp` *Time series for a three-species coupled model.*

Description

Time series generated from a discrete-time coupled Lotka-Volterra model exhibiting chaotic dynamics.

Usage

```
block_3sp
```

Format

A data frame with 198 rows and 10 columns:

`time` time index (# of generations)

`x_t` abundance of simulated species x at time t

`x_t-1` abundance of simulated species x at time $t-1$

`x_t-2` abundance of simulated species x at time $t-2$

`y_t` abundance of simulated species y at time t

`y_t-1` abundance of simulated species y at time $t-1$

`y_t-2` abundance of simulated species y at time $t-2$

`z_t` abundance of simulated species z at time t

`z_t-1` abundance of simulated species z at time $t-1$

`z_t-2` abundance of simulated species z at time $t-2$

Author(s)

Hao Ye

`CCM` *Convergent cross mapping using simplex projection*

Description

The state-space of a multivariate dynamical system (not a purely stochastic one) encodes coherent phase-space variable trajectories. If enough information is available, one can infer the presence or absence of cross-variable interactions associated with causal links between variables. `CCM` measures the extent to which states of variable Y can reliably estimate states of variable X. This happens only if X is causally influencing Y.

If cross-variable state predictability converges as more state-space information is provided, this indicates a causal link. `CCM` performs this cross-variable mapping using Simplex, with convergence assessed across a range of observational library sizes as described in *Sugihara et al. 2012*.

Usage

```
CCM(pathIn = "./", dataFile = "", dataFrame = NULL, pathOut = "./",
  predictFile = "", E = 0, Tp = 1, knn = 0, tau = 1, columns = "", target = "",
  libSizes = "", sample = 0, random = TRUE, replacement = FALSE, seed = 0,
  verbose = FALSE, showPlot = FALSE)
```

Arguments

`pathIn`	path to `dataFile`.
`dataFile`	.csv format data file name. The first column must be a time index or time values. The first row must be column names.
`dataFrame`	input data.frame. The first column must be a time index or time values. The columns must be named.
`pathOut`	path for `predictFile` containing output predictions.
`predictFile`	output file name.
`E`	embedding dimension.
`Tp`	prediction horizon (number of time column rows).
`knn`	number of nearest neighbors. If knn=0, knn is set to E+1.
`tau`	lag of time delay embedding specified as number of time column rows.
`columns`	string of whitespace separated column name(s) in the input data used to create the library.
`target`	column name in the input data used for prediction.
`libSizes`	string of 3 whitespace separated integer values specifying the intial library size, the final library size, and the library size increment.

`sample`	integer specifying the number of random samples to draw at each library size evaluation.
`random`	logical to specify random (`TRUE`) or sequential library sampling.
`replacement`	logical to specify sampling with replacement.
`seed`	integer specifying the random sampler seed. If `seed=0` then a random seed is generated.
`verbose`	logical to produce additional console reporting.
`showPlot`	logical to plot results.

Details

`CCM` computes the X:Y and Y:X cross-mappings in parallel using threads.

Value

A data.frame with 3 columns. The first column is `LibSize` specifying the subsampled library size. Columns 2 and 3 report Pearson correlation coefficients for the prediction of X from Y, and Y from X.

References

Sugihara G., May R., Ye H., Hsieh C., Deyle E., Fogarty M., Munch S., 2012. Detecting Causality in Complex Ecosystems. Science 338:496-500.

Examples

```
data(sardine_anchovy_sst)
df <- CCM( dataFrame=sardine_anchovy_sst, E=3, Tp=0, columns="anchovy",
target="np_sst", libSizes="10 70 10", sample=100 )
```

`circle` *2-D timeseries of a circle.*

Description

Time series of of circle in 2-D (sin and cos).

Usage

```
circle
```

Format

A data frame with 200 rows and 3 columns:

`Time` time index.

`x` sin component.

`y` cos component.

ComputeError — *Compute error*

Description

ComputeError evaluates the Pearson correlation coefficient, mean absolute error and root mean square error between two numeric vectors.

Usage

```
ComputeError(obs, pred)
```

Arguments

obs	vector of observations.
pred	vector of predictions.

Value

A name list with components:

rho	Pearson correlation
MAE	mean absolute error
RMSE	root mean square error

Examples

```
data(block_3sp)
smplx <- Simplex( dataFrame=block_3sp, lib="1 99", pred="105 190", E=3,
columns="x_t", target="x_t")
err <- ComputeError( smplx$Observations, smplx$Predictions )
```

Embed — *Embed data with time lags*

Description

Embed performs Takens time-delay embedding on `columns`.

Usage

```
Embed(path = "./", dataFile = "", dataFrame = NULL, E = 0, tau = 1,
columns = "", verbose = FALSE)
```

Arguments

`path`	path to `dataFile`.
`dataFile`	.csv format data file name. The first column must be a time index or time values. The first row must be column names. One of `dataFile` or `dataFrame` are required.
`dataFrame`	input data.frame. The first column must be a time index or time values. The columns must be named. One of `dataFile` or `dataFrame` are required.
`E`	embedding dimension.
`tau`	positive integer time delay embedding lag specified as number of time column rows.
`columns`	string of whitespace separated column name(s) in the input data to be embedded.
`verbose`	logical to produce additional console reporting.

Details

Each `columns` item will have E-1 time-lagged vectors created. The column name is appended with `(t-n)`. For example, data columns X, Y, with E = 2 will have columns named `X(t-0) X(t-1) Y(t-0) Y(t-1)`.

The returned data.frame does not have a time column. The returned data.frame is truncated by tau * (E-1) rows to remove state vectors with partial data (NaN elements).

Value

A data.frame with lagged columns. E columns for each variable specified in `columns`.

Examples

```
data(circle)
embed <- Embed( dataFrame = circle, E = 2, tau = 1, columns = "x y" )
```

`EmbedDimension` *Optimal embedding dimension*

Description

`EmbedDimension` uses `Simplex` to evaluate prediction accuracy as a function of embedding dimension.

Usage

```
EmbedDimension(pathIn = "./", dataFile = "", dataFrame = NULL, pathOut = "",
  predictFile = "", lib = "", pred = "", maxE = 10, Tp = 1, tau = 1,
  columns = "", target = "", embedded = FALSE, verbose = FALSE, numThreads = 4,
  showPlot = TRUE)
```

Arguments

`pathIn`	path to `dataFile`.
`dataFile`	.csv format data file name. The first column must be a time index or time values. The first row must be column names.
`dataFrame`	input data.frame. The first column must be a time index or time values. The columns must be named.
`pathOut`	path for `predictFile` containing output predictions.
`predictFile`	output file name.
`lib`	string with start and stop indices of input data rows used to create the library of observations. A single contiguous range is supported.
`pred`	string with start and stop indices of input data rows used for predictions. A single contiguous range is supported.
`maxE`	maximum value of E to evalulate.
`Tp`	prediction horizon (number of time column rows).
`tau`	lag of time delay embedding specified as number of time column rows.
`columns`	string of whitespace separated column name(s) in the input data used to create the library.
`target`	column name in the input data used for prediction.
`embedded`	logical specifying if the input data are embedded.
`verbose`	logical to produce additional console reporting.
`numThreads`	number of parallel threads for computation.
`showPlot`	logical to plot results.

Value

A data.frame with columns `E, rho`.

Examples

```
data(TentMap)
E.rho <- EmbedDimension( dataFrame=TentMap, lib="1 100", pred="201 500",
columns="TentMap", target="TentMap", showPlot=FALSE)
```

`EvergladesFlow` *Water flow to NE Everglades*

Description

Cumulative weekly water flow into northeast Everglades from water control structures S12C, S12D and S333 from 1980 through 2005.

Usage

```
EvergladesFlow
```

Format

A data frame with 1379 rows and 2 columns:

`Date` Date.

`S12CD_S333_CFS` Cumulative weekly flow (CFS).

`Lorenz5D` *5-D Lorenz'96*

Description

5-D Lorenz'96 timeseries with F = 8.

Usage

```
Lorenz5D
```

Format

Data frame with 1000 rows and 6 columns

`Time` Time.

`V1` variable 1.

`V2` variable 2.

`V3` variable 3.

`V4` variable 4.

`V5` variable 5.

References

Lorenz, Edward (1996). Predictability - A problem partly solved, Seminar on Predictability, Vol. I, ECMWF.

MakeBlock *Make embedded data block*

Description

MakeBlock performs Takens time-delay embedding on `columns`. It is an internal function called by Embed that does not perform input error checking or validation.

Usage

```
MakeBlock(dataFrame, E = 0, tau = 1, columns = "", verbose = FALSE)
```

Arguments

`dataFrame`	input data.frame. The first column must be a time index or time values. The columns must be named.
`E`	embedding dimension.
`tau`	positive integer time delay embedding lag specified as number of time column rows.
`columns`	string of whitespace separated column name(s) in the input data to be embedded.
`verbose`	logical to produce additional console reporting.

Details

Each `columns` item will have E-1 time-lagged vectors created. The column name is appended with `(t-n)`. For example, data columns X, Y, with E = 2 will have columns named `X(t-0) X(t-1) Y(t-0) Y(t-1)`.

The returned data.frame does not have a time column. The returned data.frame is truncated by tau * (E-1) rows to remove state vectors with partial data (NaN elements).

Value

A data.frame with lagged columns. E columns for each variable specified in `columns`.

Examples

```
data(TentMap)
embed <- MakeBlock(TentMap, 3, 1, "TentMap")
```

Multiview *Forecasting using multiview embedding*

Description

Multiview applies the method of *Ye & Sugihara* to find optimal combinations of variables that best represent the dynamics.

Usage

```
Multiview(pathIn = "./", dataFile = "", dataFrame = NULL, pathOut = "./",
  predictFile = "", lib = "", pred = "", E = 0, Tp = 1, knn = 0,
  tau = 1, columns = "", target = "", multiview = 0, exclusionRadius = 0,
  verbose = FALSE, numThreads = 4, showPlot = FALSE)
```

Arguments

pathIn	path to dataFile.
dataFile	.csv format data file name. The first column must be a time index or time values. The first row must be column names.
dataFrame	input data.frame. The first column must be a time index or time values. The columns must be named.
pathOut	path for predictFile containing output predictions.
predictFile	prediction output file name.
lib	string with start and stop indices of input data rows used to create the library of observations. A single contiguous range is supported.
pred	string with start and stop indices of input data rows used for predictions. A single contiguous range is supported.
E	multivariate dimension.
Tp	prediction horizon (number of time column rows).
knn	number of nearest neighbors. If knn=0, knn is set to E+1.
tau	lag of time delay embedding specified as number of time column rows.
columns	string of whitespace separated column name(s) in the input data used to create multivariable data sets.
target	column name in the input data used for prediction.
multiview	number of multiview ensembles to average for the final prediction estimate.
exclusionRadius	number of adjacent observation vector rows to exclude as nearest neighbors in prediction.
verbose	logical to produce additional console reporting.
numThreads	number of CPU threads to use in multiview processing.
showPlot	logical to plot results.

Details

Multiview embedding is a method to identify variables in a multivariate dynamical system that are most likely to contribute to the observed dynamics. It is a multistep algorithm with these general steps:

1. Evaluate E-dimensional variable combinations within the library itself (in-sample).
2. Rank in-sample (library) forecasts.
3. Compute Multiview averaged out-of-sample prediction

Value

Named list with data.frames `[[Combo_rho, Predictions]]`.

data.frame `Combo_rho` columns:

Col_1	column index
...	column index
Col_E	column index
rho	Pearson correlation
MAE	mean absolute error
RMSE	root mean square error

References

Ye H., and G. Sugihara, 2016. Information leverage in interconnected ecosystems: Overcoming the curse of dimensionality. Science 353:922-925.

Examples

```
data(block_3sp)
L = Multiview( dataFrame = block_3sp, lib = "1 99", pred = "105 190",
E = 2, columns = "x_t y_t z_t", target = "x_t" )
```

PredictInterval *Forecast interval accuracy*

Description

PredictInterval uses Simplex to evaluate prediction accuracy as a function of forecast interval Tp.

Usage

```
PredictInterval(pathIn = "./", dataFile = "", dataFrame = NULL, pathOut = "./",
  predictFile = "", lib = "", pred = "", maxTp = 10, E = 1, tau = 1,
  columns = "", target = "", embedded = FALSE, verbose = FALSE,
  numThreads = 4, showPlot = TRUE)
```

Arguments

`pathIn`	path to `dataFile`.
`dataFile`	.csv format data file name. The first column must be a time index or time values. The first row must be column names.
`dataFrame`	input data.frame. The first column must be a time index or time values. The columns must be named.
`pathOut`	path for `predictFile` containing output predictions.
`predictFile`	output file name.
`lib`	string with start and stop indices of input data rows used to create the library of observations. A single contiguous range is supported.
`pred`	string with start and stop indices of input data rows used for predictions. A single contiguous range is supported.
`maxTp`	maximum value of Tp to evalulate.
`E`	embedding dimension.
`tau`	lag of time delay embedding specified as number of time column rows.
`columns`	string of whitespace separated column name(s) in the input data used to create the library.
`target`	column name in the input data used for prediction.
`embedded`	logical specifying if the input data are embedded.
`verbose`	logical to produce additional console reporting.
`numThreads`	number of parallel threads for computation.
`showPlot`	logical to plot results.

Value

A data.frame with columns `Tp, rho`.

Examples

```
data(TentMap)
Tp.rho <- PredictInterval( dataFrame=TentMap, lib="1 100",
pred="201 500", E=2, columns="TentMap", target="TentMap", showPlot = FALSE)
```

`PredictNonlinear` *Test for nonlinear dynamics*

Description

`PredictNonlinear` uses `SMap` to evaluate prediction accuracy as a function of the localisation parameter `theta`.

Usage

```
PredictNonlinear(pathIn = "./", dataFile = "", dataFrame = NULL,
  pathOut = "./",  predictFile = "", lib = "", pred = "", theta = "",
  E = 1, Tp = 1, tau = 1, columns = "", target = "", embedded = FALSE,
  verbose = FALSE, numThreads = 4, showPlot = TRUE)
```

Arguments

`pathIn`	path to `dataFile`.
`dataFile`	.csv format data file name. The first column must be a time index or time values. The first row must be column names.
`dataFrame`	input data.frame. The first column must be a time index or time values. The columns must be named.
`pathOut`	path for `predictFile` containing output predictions.
`predictFile`	output file name.
`lib`	string with start and stop indices of input data rows used to create the library of observations. A single contiguous range is supported.
`pred`	string with start and stop indices of input data rows used for predictions. A single contiguous range is supported.
`theta`	A whitespace delimeted string with values of the S-map localisation parameter. An empty string will use default values of `[0.01 0.1 0.3 0.5 0.75 1 1.5 2 3 4 5 6 7 8 9]`.
`E`	embedding dimension.
`Tp`	prediction horizon (number of time column rows).
`tau`	lag of time delay embedding specified as number of time column rows.
`columns`	string of whitespace separated column name(s) in the input data used to create the library.
`target`	column name in the input data used for prediction.
`embedded`	logical specifying if the input data are embedded.
`verbose`	logical to produce additional console reporting.
`numThreads`	number of parallel threads for computation.
`showPlot`	logical to plot results.

Details

The localisation parameter `theta` weights nearest neighbors according to exp((-theta D / D_avg)) where D is the distance between the observation vector and neighbor, D_avg the mean distance. If theta = 0, weights are uniformally unity corresponding to a global autoregressive model. As theta increases, neighbors in closer proximity to the observation are considered.

Value

A data.frame with columns `Theta, rho`.

Examples

```
data(TentMapNoise)
theta.rho <- PredictNonlinear( dataFrame=TentMapNoise, E=2,lib="1 100",
pred="201 500", columns="TentMap", target="TentMap", showPlot = FALSE)
```

---

rEDM *Empirical dynamic modeling*

Description

rEDM provides tools for data-driven time series analyses. It is based on reconstructing multivariate state space (phase space) representations from uni- or multivariate time series, then evaluating or projecting state changes using nearest neighbor and phase space permutation algorithms.

rEDM is a **Rcpp** interface to the cppEDM library of Empirical Dynamic Modeling tools. Functionality includes:

- Simplex projection (Sugihara and May 1990)
- Sequential Locally Weighted Global Linear Maps (S-map) (Sugihara 1994)
- Multivariate embeddings (Dixon et. al. 1999)
- Convergent cross mapping (Sugihara et. al. 2012)
- Multiview embedding (Ye and Sugihara 2016)

Note: Version 1.0.1 is a major rewrite of the 'rEDM' package as a wrapper for the cppEDM library. This provides a unified computation engine for EDM algorithms across C++, Python and R implementations. The cppEDM is written in a functional programming paradigm, facilitating clarity and ease of understanding to foster collaboration and extension. Additionally, cppEDM is intended for computationally intensive application, such as supercomputers including the AI Bridging Cloud Infrastructure (ABCI) where it has been applied to massively parallel neuronal timeseries cross mapping. The new package provides improved alignment between observed and forecast data rows, and, strict exclusion of partial data vectors.

Details

Main Functions:

- `Simplex` - simplex projection
- `SMap` - S-map projection
- `CCM` - convergent cross mapping
- `Multiview` - multiview forecasting

Helper Functions:

- `Embed` - time delay embedding
- `ComputeError` - forecast skill metrics
- `EmbedDimension` - optimal embedding dimension
- `PredictInterval` - optimal prediction interval
- `PredictNonlinear` - evaluate nonlinearity

Author(s)

Maintainer: Joseph Park & Cameron Smith

Authors: George Sugihara, Joseph Park, Ethan Deyle, Cameron Smith, Erik Saberski

Contributors: Alan Trombla, Richard Penner, Victor Wong, Martin Casdagli, Jerome Cartagena, Mohsen Azarbayejani, Ava Pierce, Jennifer Trezzo

References

Sugihara G. and May R. 1990. Nonlinear forecasting as a way of distinguishing chaos from measurement error in time series. Nature, 344:734-741. <DOI:10.1038/344734a0>

Sugihara G. 1994. Nonlinear forecasting for the classification of natural time series. Philosophical Transactions: Physical Sciences and Engineering, 348 (1688) : 477-495. <DOI:10.1098/rsta.1994.0106>

Dixon, P. A., M. Milicich, and G. Sugihara, 1999. Episodic fluctuations in larval supply. Science 283:1528-1530.

Sugihara G., May R., Ye H., Hsieh C., Deyle E., Fogarty M., Munch S., 2012. Detecting Causality in Complex Ecosystems. Science 338:496-500. <DOI:10.1126/science.1227079>

Ye H., and G. Sugihara, 2016. Information leverage in interconnected ecosystems: Overcoming the curse of dimensionality. Science 353:922-925. <DOI:10.1126/science.aag0863>

`sardine_anchovy_sst` *Time series for the California Current Anchovy-Sardine-SST system*

Description

Time series of Pacific sardine landings (CA), Northern anchovy landings (CA), and sea-surface temperature (3-year average) at the SIO pier and Newport pier

Usage

```
sardine_anchovy_sst
```

Format

`year` year of measurement

`anchovy` anchovy landings, scaled to mean = 0, sd = 1

`sardine` sardine landings, scaled to mean = 0, sd = 1

`sio_sst` 3-year running average of sea surface temperature at SIO pier, scaled to mean = 0, sd = 1

`np_sst` 3-year running average of sea surface temperature at Newport pier, scaled to mean = 0, sd = 1

Simplex *Simplex forecasting*

Description

Simplex performs time series forecasting based on weighted nearest neighbors projection in the time series phase space as described in *Sugihara and May*.

Usage

```
Simplex(pathIn = "./", dataFile = "", dataFrame = NULL, pathOut = "./",
  predictFile = "", lib = "", pred = "", E = 0, Tp = 1, knn = 0, tau = 1,
  exclusionRadius = 0, columns = "", target = "", embedded = FALSE,
  verbose = FALSE, const_pred = FALSE, showPlot = FALSE)
```

Arguments

pathIn	path to dataFile.
dataFile	.csv format data file name. The first column must be a time index or time values. The first row must be column names.
dataFrame	input data.frame. The first column must be a time index or time values. The columns must be named.
pathOut	path for predictFile containing output predictions.
predictFile	output file name.
lib	string with start and stop indices of input data rows used to create the library of observations. A single contiguous range is supported.
pred	string with start and stop indices of input data rows used for predictions. A single contiguous range is supported.
E	embedding dimension.
Tp	prediction horizon (number of time column rows).
knn	number of nearest neighbors. If knn=0, knn is set to E+1.
tau	lag of time delay embedding specified as number of time column rows.
exclusionRadius	excludes vectors from the search space of nearest neighbors if their relative time index is within exclusionRadius.
columns	string of whitespace separated column name(s) in the input data used to create the library.
target	column name in the input data used for prediction.
embedded	logical specifying if the input data are embedded.
verbose	logical to produce additional console reporting.
const_pred	logical to add a *constant predictor* column to the output. The constant predictor is X(t+1) = X(t).
showPlot	logical to plot results.

Details

If embedded is `FALSE`, the data `column(s)` are embedded to dimension `E` with time lag `tau`. This embedding forms an E-dimensional phase space for the `Simplex` projection. If embedded is `TRUE`, the data are assumed to contain an E-dimensional embedding with E equal to the number of `columns`. Predictions are made using leave-one-out cross-validation, i.e. observation vectors are excluded from the prediction simplex.

To assess an optimal embedding dimension `EmbedDimension` can be applied. Accuracy statistics can be estimated by `ComputeError`.

Value

A data.frame with columns `Observations, Predictions`. If `const_pred` is TRUE the column `Const_Predictions` is added. The first column contains the time values.

References

Sugihara G. and May R. 1990. Nonlinear forecasting as a way of distinguishing chaos from measurement error in time series. Nature, 344:734-741.

Examples

```
data(block_3sp)
smplx <- Simplex( dataFrame=block_3sp, lib="1 99", pred="105 190",
E=3, columns="x_t", target="x_t" )
ComputeError(smplx$Predictions, smplx$Observations)
```

`SMap` *SMap forecasting*

Description

`SMap` performs time series forecasting based on localised (or global) nearest neighbor projection in the time series phase space as described in *Sugihara 1994*.

Usage

```
SMap(pathIn = "./", dataFile = "", dataFrame = NULL, pathOut = "./",
  predictFile = "", lib = "", pred = "", E = 0, Tp = 1, knn = 0, tau = 1,
  theta = 0, exclusionRadius = 0, columns = "", target = "", smapFile = "",
  jacobians = "", embedded = FALSE, const_pred = FALSE, verbose = FALSE,
  showPlot = FALSE)
```

Arguments

`pathIn`	path to `dataFile`.
`dataFile`	.csv format data file name. The first column must be a time index or time values. The first row must be column names.
`dataFrame`	input data.frame. The first column must be a time index or time values. The columns must be named.
`pathOut`	path for `predictFile` containing output predictions.
`predictFile`	prediction output file name.
`lib`	string with start and stop indices of input data rows used to create the library of observations. A single contiguous range is supported.
`pred`	string with start and stop indices of input data rows used for predictions. A single contiguous range is supported.
`E`	embedding dimension.
`Tp`	prediction horizon (number of time column rows).
`knn`	number of nearest neighbors. If knn=0, knn is set to E+1.
`tau`	lag of time delay embedding specified as number of time column rows.
`theta`	neighbor localisation exponent.
`exclusionRadius`	excludes vectors from the search space of nearest neighbors if their relative time index is within exclusionRadius.
`columns`	string of whitespace separated column name(s) in the input data used to create the library.
`target`	column name in the input data used for prediction.
`smapFile`	output file containing SMap coefficients.
`jacobians`	not used.
`embedded`	logical specifying if the input data are embedded.
`const_pred`	logical to add a *constant predictor* column to the output. The constant predictor is X(t+1) = X(t).
`verbose`	logical to produce additional console reporting.
`showPlot`	logical to plot results.

Details

If embedded is `FALSE`, the data `column(s)` are embedded to dimension `E` with time lag `tau`. This embedding forms an E-dimensional phase space for the `SMap` projection. If embedded is `TRUE`, the data are assumed to contain an E-dimensional embedding with E equal to the number of `columns`. Predictions are made using leave-one-out cross-validation, i.e. observation vectors are excluded from the prediction regression.

In contrast to `Simplex`, `SMap` uses all available neighbors and then weights them with an exponential decay in phase space distance with exponent `theta`. `theta=0` uses all neighbors corresponding to a global autoregressive model. As `theta` increases, neighbors closer in vicinity to the observation are considered.

Value

A named list with two data.frames `[[predictions, coefficients]]`. `predictions` has columns `Observations, Predictions`. If `const_pred` is TRUE the column `Const_Predictions` is added. The first column contains time values.

`coefficients` data.frame has time values in the first column. Columns 2 through E+2 (E+1 columns) are the SMap coefficients.

Note

`SMap` should be called with columns explicity correspondng to dimensions E. This means that if a multivariate data set is used (number of `columns` > 1) it should Not be an embedding from `Embed` since `Embed` will add lagged coordinates for each variable. The added columns will not correspond to the intended dimensions in the matrix inversion and prediction reconstruction. In this case, use the `embedded = TRUE` flag so that the columns selected and their coefficients correspond to the proper dimension.

References

Sugihara G. 1994. Nonlinear forecasting for the classification of natural time series. Philosophical Transactions: Physical Sciences and Engineering, 348 (1688):477-495.

Examples

```
data(circle)
L = SMap( dataFrame=circle,lib="1 100", pred="110 190", theta=4, E=2,
embedded=TRUE,columns="x y", target="x")
```

`SurrogateData` *Generate surrogate data for permutation/randomization tests*

Description

`SurrogateData` generates surrogate data under several different null models.

Usage

```
SurrogateData( ts, method = c("random_shuffle", "ebisuzaki",
"seasonal"), num_surr = 100, T_period = 1, alpha = 0 )
```

Arguments

`ts`	the original time series
`method`	which algorithm to use to generate surrogate data
`num_surr`	the number of null surrogates to generate
`T_period`	the period of seasonality for seasonal surrogates (ignored for other methods)
`alpha`	additive noise factor: N(0,alpha)

Details

Method "random_shuffle" creates surrogates by randomly permuting the values of the original time series.

Method "Ebisuzaki" creates surrogates by randomizing the phases of a Fourier transform, preserving the power spectra of the null surrogates.

Method "seasonal" creates surrogates by computing a mean seasonal trend of the specified period and shuffling the residuals. It is presumed that the seasonal trend can be exracted with a smoothing spline. Additive Gaussian noise is included according to N(0,alpha).

Value

A matrix where each column is a separate surrogate with the same length as `ts`.

Examples

```
data("block_3sp")
ts <- block_3sp$x_t
SurrogateData(ts, method = "ebisuzaki")
```

TentMap — *Time series for a tent map with mu = 2.*

Description

First-differenced time series generated from the tent map recurrence relation with mu = 2.

Usage

```
TentMap
```

Format

Data frame with 999 rows and 2 columns

`Time` time index.

`TentMap` tent map values.

TentMapNoise *Time series of tent map plus noise.*

Description

First-differenced time series generated from the tent map recurrence relation with mu = 2 and random noise.

Usage

```
TentMapNoise
```

Format

Data frame with 999 rows and 2 columns

`Time` time index.

`TentMap` tent map values.

Thrips *Apple-blossom Thrips time series*

Description

Seasonal outbreaks of Thrips imaginis.

References

Davidson and Andrewartha, Annual trends in a natural population of Thrips imaginis *Thysanoptera*, Journal of Animal Ecology, 17, 193-199, 1948.

Index