Package 'rEDM'

October 22, 2016

Type Package

Title Applications of Empirical Dynamic Modeling from Time Series

Version 0.5.3

Date 2016-10-22

Maintainer Hao Ye <hye@ucsd.edu>

Author Hao Ye [aut, cre],
Adam Clark [aut],
Ethan Deyle [aut],
Oliver Keyes [ctb],
George Sugihara [ctb, ccp]

Description A new implementation of EDM algorithms and is based on research software previously developed for internal use in the Sugihara Lab (UCSD/SIO). Contains C++ compiled objects that use time delay embedding to perform state-space reconstruction and nonlinear forecasting and an R interface to those objects using 'Rcpp'. It supports both the simplex projection method from Sugihara & May (1990) <DOI:10.1038/344734a0> and the S-map algorithm in Sugihara (1994) <DOI:10.1098/rsta.1994.0106>. In addition, this package implements convergent cross mapping as described in Sugihara et al. (2012) <DOI:10.1126/science.1227079> and multiview embedding as described in Ye & Sugihara (2016) <DOI:10.1126/science.aag0863>.

NeedsCompilation yes

Imports Rcpp (>= 0.11.5), methods

LinkingTo Rcpp, RcppEigen

RcppModules lnlp_module, block_lnlp_module, ccm_module

Suggests knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 5.0.1

Repository CRAN

Date/Publication 2016-10-22 19:06:25

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.

Author(s)

Hao Ye

`block_lnlp` *Perform generalized forecasting using simplex projection or s-map*

Description

`block_lnlp` uses multiple time series given as input to generate an attractor reconstruction, and then applies the simplex projection or s-map algorithm to make forecasts. This method generalizes the `simplex` and `s-map` routines, and allows for "mixed" embeddings, where multiple time series can be used as different dimensions of an attractor reconstruction.

Usage

```
block_lnlp(block, lib = c(1, NROW(block)), pred = lib,
  norm_type = c("L2 norm", "L1 norm", "P norm"), P = 0.5,
  method = c("simplex", "s-map"), tp = 1, num_neighbors = "e+1",
  columns = NULL, target_column = 1, stats_only = TRUE,
  first_column_time = FALSE, exclusion_radius = NULL, epsilon = NULL,
  theta = NULL, silent = FALSE, save_smap_coefficients = FALSE,
  short_output = FALSE)
```

Arguments

block	either a vector to be used as the time series, or a data.frame or matrix where each column is a time series
lib	a 2-column matrix (or 2-element vector) where each row specifes the first and last *rows* of the time series to use for attractor reconstruction
pred	(same format as lib), but specifying the sections of the time series to forecast.
norm_type	the distance function to use. see 'Details'
P	the exponent for the P norm
method	the prediction method to use. see 'Details'
tp	the prediction horizon (how far ahead to forecast)
num_neighbors	the number of nearest neighbors to use (any of "e+1", "E+1", "e + 1", "E + 1" will peg this parameter to E+1 for each run, any value < 1 will use all possible neighbors.)
columns	either a vector with the columns to use (indices or names), or a list of such columns
target_column	the index (or name) of the column to forecast
stats_only	specify whether to output just the forecast statistics or the raw predictions for each run
first_column_time	indicates whether the first column of the given block is a time column (and therefore excluded when indexing)
exclusion_radius	excludes vectors from the search space of nearest neighbors if their *time index* is within exclusion_radius (NULL turns this option off)
epsilon	excludes vectors from the search space of nearest neighbors if their *distance* is farther away than epsilon (NULL turns this option off)
theta	the nonlinear tuning parameter (theta is only relevant if method == "s-map")
silent	prevents warning messages from being printed to the R console
save_smap_coefficients	specifies whether to include the s_map coefficients with the output (and forces the full output as if stats_only were set to FALSE)
short_output	specifies whether to return a truncated output data.frame whose rows only include the predictions made and not the whole input block

Details

The default parameters are set so that passing a vector as the only argument will use that vector to predict itself one time step ahead. If a matrix or data.frame is given as the only argument, the first column will be predicted, using the remaining columns as the embedding. Rownames will be converted to numeric if possible to be used as the time index, otherwise 1:NROW will be used instead. The default lib and pred are for leave-one-out cross-validation over the whole time series, and returning just the forecast statistics.

norm_type "L2 norm" (default) uses the typical Euclidean distance:

$$distance(a, b) := \sqrt{\sum_i (a_i - b_i)^2}$$

norm_type "L1 norm" uses the Manhattan distance:

$$distance(a, b) := \sum_i |a_i - b_i|$$

norm type "P norm" uses the P norm, generalizing the L1 and L2 norm to use p as the exponent:

$$distance(a, b) := \sum_i (a_i - b_i)^{p\,1/p}$$

method "simplex" (default) uses the simplex projection forecasting algorithm

method "s-map" uses the s-map forecasting algorithm

Value

If stats_only, then a data.frame with components for the parameters and forecast statistics:

cols	embedding
tp	prediction horizon
nn	number of neighbors
num_pred	number of predictions
rho	correlation coefficient between observations and predictions
mae	mean absolute error
rmse	root mean square error
perc	percent correct sign
p_val	p-value that rho is significantly greater than 0 using Fisher's z-transformation
const_rho	same as rho, but for the constant predictor
const_mae	same as mae, but for the constant predictor
const_rmse	same as rmse, but for the constant predictor
const_perc	same as perc, but for the constant predictor
const_p_val	same as p_val, but for the constant predictor

Otherwise, a list where the number of elements is equal to the number of runs (unique parameter combinations). Each element is a list with the following components:

params	data.frame of parameters (embedding, tp, nn)

model_output	data.frame with columns for the time index, observations, and predictions
stats	data.frame of forecast statistics

Examples

```
data("two_species_model")
block <- two_species_model[1:200,]
block_lnlp(block, columns = c("x", "y"), first_column_time = TRUE)
```

---

ccm *Perform convergent cross mapping using simplex projection*

Description

`ccm` uses time delay embedding on one time series to generate an attractor reconstruction, and then applies the simplex projection algorithm to estimate concurrent values of another time series. This method is typically applied, varying the library sizes, to determine if one time series contains the necessary dynamic information to recover the influence of another, causal variable.

Usage

```
ccm(block, lib = c(1, NROW(block)), pred = lib, norm_type = c("L2 norm",
  "L1 norm", "LP norm"), P = 0.5, E = 1, tau = 1, tp = 0,
  num_neighbors = "e+1", lib_sizes = seq(10, 100, by = 10),
  random_libs = TRUE, num_samples = 100, replace = TRUE, lib_column = 1,
  target_column = 2, first_column_time = FALSE, RNGseed = NULL,
  exclusion_radius = NULL, epsilon = NULL, silent = FALSE)
```

Arguments

`block`	either a vector to be used as the time series, or a data.frame or matrix where each column is a time series
`lib`	a 2-column matrix (or 2-element vector) where each row specifes the first and last *rows* of the time series to use for attractor reconstruction
`pred`	(same format as lib), but specifying the sections of the time series to forecast.
`norm_type`	the distance function to use. see 'Details'
`P`	the exponent for the P norm
`E`	the embedding dimensions to use for time delay embedding
`tau`	the lag to use for time delay embedding
`tp`	the prediction horizon (how far ahead to forecast)
`num_neighbors`	the number of nearest neighbors to use (any of "e+1", "E+1", "e + 1", "E + 1" will peg this parameter to E+1 for each run, any value < 1 will use all possible neighbors.)
`lib_sizes`	the vector of library sizes to try

random_libs	indicates whether to use randomly sampled libs
num_samples	is the number of random samples at each lib size (this parameter is ignored if random_libs is FALSE)
replace	indicates whether to sample vectors with replacement
lib_column	the index (or name) of the column to cross map from
target_column	the index (or name) of the column to cross map to
first_column_time	indicates whether the first column of the given block is a time column (and therefore excluded when indexing)
RNGseed	will set a seed for the random number generator, enabling reproducible runs of ccm with randomly generated libraries
exclusion_radius	excludes vectors from the search space of nearest neighbors if their *time index* is within exclusion_radius (NULL turns this option off)
epsilon	excludes vectors from the search space of nearest neighbors if their *distance* is farther away than epsilon (NULL turns this option off)
silent	prevents warning messages from being printed to the R console

Details

The default parameters are set so that passing a matrix as the only argument will use E = 1 (embedding dimension), and leave-one-out cross-validation over the whole time series to compute cross-mapping from the first column to the second column, letting the library size vary from 10 to 100 in increments of 10.

norm_type "L2 norm" (default) uses the typical Euclidean distance:

$$distance(a, b) := \sqrt{\sum_i (a_i - b_i)^2}$$

norm_type "L1 norm" uses the Manhattan distance:

$$distance(a, b) := \sum_i |a_i - b_i|$$

norm type "P norm" uses the LP norm, generalizing the L1 and L2 norm to use p as the exponent:

$$distance(a, b) := \sum_i (a_i - b_i)^{p\,1/p}$$

Value

A data.frame with forecast statistics for the different parameter settings:

L	library length (number of vectors)
num_pred	number of predictions
rho	correlation coefficient between observations and predictions
mae	mean absolute error
rmse	root mean square error

Examples

```
data("sardine_anchovy_sst")
anchovy_xmap_sst <- ccm(sardine_anchovy_sst, E = 3,
  lib_column = "anchovy", target_column = "np_sst",
  lib_sizes = seq(10, 80, by = 10), num_samples = 100)
```

`ccm_means` *Take output from ccm and compute means as a function of library size.*

Description

`ccm_means` is a utility function to summarize output from the `ccm` function

Usage

```
ccm_means(ccm_df, FUN = mean, ...)
```

Arguments

`ccm_df`	a data.frame, usually output from the `ccm` function
`FUN`	a function that aggregates the numerical statistics (by default, uses the mean)
`...`	optional arguments to FUN

Value

A data.frame with forecast statistics aggregated at each unique library size

Examples

```
data("sardine_anchovy_sst")
anchovy_xmap_sst <- ccm(sardine_anchovy_sst, E = 3,
  lib_column = "anchovy", target_column = "np_sst",
  lib_sizes = seq(10, 80, by = 10), num_samples = 100)
a_xmap_t_means <- ccm_means(anchovy_xmap_sst)
```

compute_stats *Compute performance metrics for predictions*

Description

Computes the rho, MAE, RMSE, perc, and p-val performance metrics using the compiled C++ function

Arguments

observed	a vector of the observed values
predicted	a vector of the corresponding predicted values

Value

A data.frame with components for the various performance metrics:

num_pred	number of predictions
rho	correlation coefficient between observations and predictions
mae	mean absolute error
rmse	root mean square error
perc	percent correct sign
p_val	p-value that rho is significantly greater than 0 using Fisher's

Examples

```
compute_stats(rnorm(100), rnorm(100))
```

e054_succession *Succession data at the Cedar Creek LTER*

Description

Experiment 054 is a subset of the long-term observational study of old field succession at the Cedar Creek LTER.

Author(s)

****

e120_biodiversity *Biodiversity data at the Cedar Creek LTER*

Description

Experiment 120, the "Big Biodiversity" experiment at Cedar Creek LTER. This experiment is the longest running randomized test for the effects of plant diversity on ecosystem functions.

Author(s)

****

make_block *Make a lagged block for multiview*

Description

`make_block` generates a lagged block with the appropriate max_lag and tau, while respecting lib (by inserting NANs, when trying to lag past lib regions)

Usage

```
make_block(block, max_lag = 3, t = NULL, lib = NULL, tau = 1)
```

Arguments

`block`	a data.frame or matrix where each column is a time series
`max_lag`	the total number of lags to include for each variable
`t`	the time index for the block
`lib`	a 2-column matrix (or 2-element vector) where each row specifes the first and last *rows* of the time series to use for attractor reconstruction
`tau`	the lag to use for time delay embedding

Value

A data.frame with the lagged columns and a time column

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

Description

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

Usage

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

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)

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.

See `test_nonlinearity` for context.

Value

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

Examples

```
data("two_species_model")
ts <- two_species_model$x[1:200]
make_surrogate_data(ts, method = "ebisuzaki")
```

multiview *Perform forecasting using multiview embedding*

Description

multiview applies the method described in Ye & Sugihara (2016) for forecasting, wherein multiple attractor reconstructions are tested, and a single nearest neighbor is selected from each of the top "k" reconstructions to produce final forecasts.

Usage

```
multiview(block, lib = c(1, floor(NROW(block)/2)),
  pred = c(floor(NROW(block)/2), NROW(block)), norm_type = c("L2 norm",
  "L1 norm", "P norm"), P = 0.5, E = 3, tau = 1, tp = 1, max_lag = 3,
  num_neighbors = "e+1", k = "sqrt", na.rm = FALSE, target_column = 1,
  stats_only = TRUE, first_column_time = FALSE, exclusion_radius = NULL,
  silent = FALSE, short_output = FALSE)
```

Arguments

block	either a vector to be used as the time series, or a data.frame or matrix where each column is a time series
lib	a 2-column matrix (or 2-element vector) where each row specifes the first and last *rows* of the time series to use for attractor reconstruction
pred	(same format as lib), but specifying the sections of the time series to forecast.
norm_type	the distance function to use. see 'Details'
P	the exponent for the P norm
E	the embedding dimensions to use for time delay embedding
tau	the lag to use for time delay embedding
tp	the prediction horizon (how far ahead to forecast)
max_lag	the maximum number of lags to use for variable combinations
num_neighbors	the number of nearest neighbors to use for the in-sample prediction (any of "e+1", "E+1", "e + 1", "E + 1" will peg this parameter to E+1 for each run, any value < 1 will use all possible neighbors.)
k	the number of embeddings to use (any of "sqrt", "SQRT" will use k = floor(sqrt(m)))
na.rm	logical. Should missing values (including NaN be omitted from the calculations?)
target_column	the index (or name) of the column to forecast
stats_only	specify whether to output just the forecast statistics or the raw predictions for each run
first_column_time	indicates whether the first column of the given block is a time column (and therefore excluded when indexing)

`exclusion_radius`
: excludes vectors from the search space of nearest neighbors if their *time index* is within exclusion_radius (NULL turns this option off)

`silent`
: prevents warning messages from being printed to the R console

`short_output`
: specifies whether to return a truncated output data.frame whose rows only include the predictions made and not the whole input block

Details

uses multiple time series given as input to generate an attractor reconstruction, and then applies the simplex projection or s-map algorithm to make forecasts. This method generalizes the `simplex` and `s-map` routines, and allows for "mixed" embeddings, where multiple time series can be used as different dimensions of an attractor reconstruction.

The default parameters are set so that, given a matrix of time series, forecasts will be produced for the first column. By default, all possible combinations of the columns are used for the attractor construction, the k = sqrt(m) heuristic will be used, forecasts will be one time step ahead. Rownames will be converted to numeric if possible to be used as the time index, otherwise 1:NROW will be used instead. The default lib and pred are to use the first half of the data for the "library" and to predict over the second half of the data. Unless otherwise set, the output will be just the forecast statistics.

norm_type "L2 norm" (default) uses the typical Euclidean distance:

$$distance(a, b) := \sqrt{\sum_i (a_i - b_i)^2}$$

norm_type "L1 norm" uses the Manhattan distance:

$$distance(a, b) := \sum_i |a_i - b_i|$$

norm type "P norm" uses the P norm, generalizing the L1 and L2 norm to use p as the exponent:

$$distance(a, b) := \sum_i (a_i - b_i)^{p\,1/p}$$

Value

If stats_only, then a data.frame with components for the parameters and forecast statistics:

E	embedding dimension
tau	time lag
tp	prediction horizon
nn	number of neighbors
k	number of embeddings used
num_pred	number of predictions
rho	correlation coefficient between observations and predictions
mae	mean absolute error
rmse	root mean square error
perc	percent correct sign

p_val	p-value that rho is significantly greater than 0 using Fisher's z-transformation
const_rho	same as rho, but for the constant predictor
const_mae	same as mae, but for the constant predictor
const_rmse	same as rmse, but for the constant predictor
const_perc	same as perc, but for the constant predictor
const_p_val	same as p_val, but for the constant predictor

Otherwise, a list where the number of elements is equal to the number of runs (unique parameter combinations). Each element is a list with the following components:

params	data.frame of parameters (E, tau, tp, nn, k)
lib_stats	data.frame of in-sample forecast statistics
model_output	data.frame with columns for the time index, observations, and predictions
pred_stats	data.frame of forecast statistics

Examples

```
data("block_3sp")
block <- block_3sp[, c(2, 5, 8)]
multiview(block, k = c(1, 3, "sqrt"))
```

---

paramecium_didinium *Time series for the Paramecium-Didinium laboratory experiment*

Description

Time series of Paramecium and Didinium abundances (#/mL) from an experiment by Veilleux (1979)

Author(s)

Veilleux

---

rEDM *Applications of empirical dynamic modeling from time series.*

Description

The rEDM package provides an interface from R to C++ compiled objects that use time delay embedding to perform state-space reconstruction and nonlinear forecasting.

Author(s)

Hao Ye

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

Author(s)

****

simplex *Perform univariate forecasting*

Description

`simplex` uses time delay embedding on a single time series to generate an attractor reconstruction, and then applies the simplex projection algorithm to make forecasts.

`s_map` is similar to `simplex`, but uses the S-map algorithm to make forecasts.

Usage

```
simplex(time_series, lib = c(1, NROW(time_series)), pred = lib,
  norm_type = c("L2 norm", "L1 norm", "P norm"), P = 0.5, E = 1:10,
  tau = 1, tp = 1, num_neighbors = "e+1", stats_only = TRUE,
  exclusion_radius = NULL, epsilon = NULL, silent = FALSE)

s_map(time_series, lib = c(1, NROW(time_series)), pred = lib,
  norm_type = c("L2 norm", "L1 norm", "P norm"), P = 0.5, E = 1,
  tau = 1, tp = 1, num_neighbors = 0, theta = c(0, 1e-04, 3e-04, 0.001,
  0.003, 0.01, 0.03, 0.1, 0.3, 0.5, 0.75, 1, 1.5, 2, 3, 4, 6, 8),
  stats_only = TRUE, exclusion_radius = NULL, epsilon = NULL,
  silent = FALSE, save_smap_coefficients = FALSE)
```

Arguments

`time_series`	either a vector to be used as the time series, or a data.frame or matrix with at least 2 columns (in which case the first column will be used as the time index, and the second column as the time series)
`lib`	a 2-column matrix (or 2-element vector) where each row specifes the first and last *rows* of the time series to use for attractor reconstruction
`pred`	(same format as lib), but specifying the sections of the time series to forecast.
`norm_type`	the distance function to use. see 'Details'

P	the exponent for the P norm
E	the embedding dimensions to use for time delay embedding
tau	the lag to use for time delay embedding
tp	the prediction horizon (how far ahead to forecast)
num_neighbors	the number of nearest neighbors to use (any of "e+1", "E+1", "e + 1", "E + 1" will peg this parameter to E+1 for each run, any value < 1 will use all possible neighbors.)
stats_only	specify whether to output just the forecast statistics or the raw predictions for each run
exclusion_radius	excludes vectors from the search space of nearest neighbors if their *time index* is within exclusion_radius (NULL turns this option off)
epsilon	excludes vectors from the search space of nearest neighbors if their *distance* is farther away than epsilon (NULL turns this option off)
silent	prevents warning messages from being printed to the R console
theta	the nonlinear tuning parameter (note that theta = 0 is equivalent to an autoregressive model of order E.)
save_smap_coefficients	specifies whether to include the s_map coefficients with the output (and forces the full output as if stats_only were set to FALSE)

Details

simplex is typically applied, and the embedding dimension varied, to find an optimal embedding dimension for the data. Thus, the default parameters are set so that passing a time series as the only argument will run over E = 1:10 (embedding dimension), using leave-one-out cross-validation over the whole time series, and returning just the forecast statistics.

s_map is typically applied, with fixed embedding dimension, and theta varied, to test for nonlinear dynamics in the data. Thus, the default parameters are set so that passing a time series as the only argument will run over a default list of thetas (0, 0.0001, 0.0003, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 0.5, 0.75, 1.0, 1.5, 2, 3, 4, 6, and 8), using E = 1, leave-one-out cross-validation over the whole time series, and returning just the forecast statistics.

norm_type "L2 norm" (default) uses the typical Euclidean distance:

$$distance(a, b) := \sqrt{\sum_i (a_i - b_i)^2}$$

norm_type "L1 norm" uses the Manhattan distance:

$$distance(a, b) := \sum_i |a_i - b_i|$$

norm type "P norm" uses the LP norm, generalizing the L1 and L2 norm to use p as the exponent:

$$distance(a, b) := \sum_i (a_i - b_i)^{p\,1/p}$$

Value

For `simplex`, if `stats_only = TRUE`, then a data.frame with components for the parameters and forecast statistics:

E	embedding dimension
tau	time lag
tp	prediction horizon
nn	number of neighbors
num_pred	number of predictions
rho	correlation coefficient between observations and predictions
mae	mean absolute error
rmse	root mean square error
perc	percent correct sign
p_val	p-value that rho is significantly greater than 0 using Fisher's z-transformation
const_rho	same as rho, but for the constant predictor
const_mae	same as mae, but for the constant predictor
const_rmse	same as rmse, but for the constant predictor
const_perc	same as perc, but for the constant predictor
const_p_val	same as p_val, but for the constant predictor

Otherwise, a list where the number of elements is equal to the number of runs (unique parameter combinations). Each element is a list with the following components:

params	data.frame of parameters (E, tau, tp, nn)
model_output	data.frame with columns for the time index, observations, and predictions
stats	data.frame of forecast statistics

For `s_map`, the same as for `simplex`, but with an additional column for the value of theta. If `stats_only = FALSE` and `save_smap_coefficients = TRUE`, then a matrix of S-map coefficients will appear in the full output.

Examples

```
data("two_species_model")
ts <- two_species_model$x[1:200]
simplex(ts, lib = c(1, 100), pred = c(101, 200))

data("two_species_model")
ts <- two_species_model$x[1:200]
#' simplex(ts, stats_only = FALSE)
data("two_species_model")
ts <- two_species_model$x[1:200]
s_map(ts, E = 2)

data("two_species_model")
ts <- two_species_model$x[1:200]
s_map(ts, E = 2, theta = 1, save_smap_coefficients = TRUE)
```

sockeye_returns — *Time series for sockeye salmon returns.*

Description

Time series of sockeye salmon returns from the Fraser River in British Columbia, Canada.

Author(s)

****

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

Description

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

Author(s)

Hao Ye

test_nonlinearity — *Randomization test for nonlinearity using S-maps and surrogate data*

Description

`test_nonlinearity` tests for nonlinearity using S-maps by comparing improvements in forecast skill (delta rho and delta mae) between linear and nonlinear models with a null distribution from surrogate data.

Usage

```
test_nonlinearity(ts, method = "ebisuzaki", num_surr = 200, T_period = 1,
  E = 1, ...)
```

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)
`E`	the embedding dimension for s_map
`...`	optional arguments to s_map

Value

A data.frame containing the following components:

delta_rho	the value of the delta rho statistic
delta_mae	the value of the delat mae statistic
num_surr	the size of the null distribution
delta_rho_p_value	the p-value for delta rho
delta_mae_p_value	the p-value for delta mae

---

thrips_block *Apple-blossom Thrips time series*

Description

Seasonal outbreaks of Thrips imaginis.

Author(s)

****

---

two_species_model *Time series for a two-species coupled model.*

Description

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

Author(s)

Hao Ye

Index