Package ‘DChaos’

February 10, 2021

Type Package

Version 0.1-6

Date 2021-02-10

Title Chaotic Time Series Analysis

Author Julio E. Sandubete [aut, cre],
Lorenzo Escot [aut]

Maintainer Julio E. Sandubete <jsandube@ucm.es>

Imports xts, zoo, outliers, nnet, pracma, sandwich

Description Chaos theory has been hailed as a revolution of thoughts and attracting ever increasing attention of many scientists from diverse disciplines. Chaotic systems are nonlinear deterministic dynamic systems which can behave like an erratic and apparently random motion. A relevant field inside chaos theory and nonlinear time series analysis is the detection of a chaotic behaviour from empirical time series data. One of the main features of chaos is the well known initial value sensitivity property. Methods and techniques related to test the hypothesis of chaos try to quantify the initial value sensitive property estimating the Lyapunov exponents. The DChaos package provides different useful tools and efficient algorithms which test robustly the hypothesis of chaos based on the Lyapunov exponent in order to know if the data generating process behind time series behave chaotically or not.

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

NeedsCompilation no

Repository CRAN

Date/Publication 2021-02-10 10:40:36 UTC

R topics documented:

`embedding` *Provides the delayed-coordinate embedding vectors backwards*

Description

This function generates both the uniform and non-uniform embedding vectors backwards using the method of delays from univariate time-series data.

Usage

```
embedding(x, m = 2, lag = 1, timelapse = c("FIXED", "VARIABLE"))
```

Arguments

`x`	a `vector`, a time-series object `ts` or `xts`, a `data.frame`, a `data.table` or a `matrix` depending on the method selected in `timelapse`.
`m`	a non-negative integer denoting the embedding dimension (Default 2).
`lag`	a non-negative integer denoting the reconstruction delay (Default 1).
`timelapse`	a character denoting if the time-series data are sampled at uniform time-frequency e.g., 1-month, 1-day, 1-hour, 30-min, 5-min, 1-min and so on `FIXED` or non-uniform time-frequency which are not equally spaced in time `VARIABLE` (Default `FIXED`).

Value

The uniform or non-uniform delayed-coordinate embedding vectors backwards by columns from an univariate time-series data considering the parameter set selected by the user. If `FIXED` has been selected `data` must be a `vector` or a time-series object `ts` or `xts`. Otherwise `VARIABLE` has to be specified. In this case `data` must be a `data.frame`, a `data.table` or a `matrix` with two columns, the date and the univariate time series as a sequence of numerical values, in that order. The date can have the following three classes: `POSIXt`, `Date` or `Factor`. In the latter case the date should come in the following format `YMD` `H:M:OS3` considering milliseconds e.g., 20190407 00:00:03.347. If you don't consider milliseconds you must put .000 after the seconds.

Note

Note that a key point to create a suitable reconstruction of the state-space is to fix a criteria in order to estimate the embedding parameters. Researchers usually estimate them using heuristic approaches based on prescriptions proposed by e.g., H.D. Abarbanel (1996) or H. Kantz and T. Schreiber (2004). The main drawbacks of these heuristic approaches are the following: they are not intrinsically statistical; their results are not robust; they lead to estimators whose properties are unknown or largely unexplored; they do not take into account the results of any model fit. The alternative proposed by the statistical approach solves those disadvantages. The statistical approach to state-space reconstruction can be viewed as a best subset selection problem within the nonparametric regression context as argued K.-S. Chan and H. Tong (2001). The DChaos package allows the R users to choose between both methods. By default it uses the statistical approach based on model selection procedures instead of heuristic techniques, see `netfit` function.

Author(s)

Julio E. Sandubete, Lorenzo Escot

References

Ruelle, D., Takens, F. 1971 On the nature of turbulence. Communications in Mathematical Physics 20(3):167-192.

Takens, F. 1981 Detecting strange attractors in turbulence. Springer Berlin Heidelberg.

Abarbanel, H.D. 1996 Analysis of observed chaotic data. Springer.

Cha, K.-S., Tong, H. 2001 Chaos: a statistical perspective. Springer-Verlag.

Kantz, H., Schreiber, T. 2004 Nonlinear time series analysis, volume 7. Cambridge university press.

Huke, J.P., Broomhead, D.S. 2007 Embedding theorems for non-uniformly sampled dynamical systems. Nonlinearity 20(9):205-244.

Examples

```
## set.seed(34)
## Simulates time-series data from the Logistic map with chaos
## ts       <- DChaos::logistic.sim(n=1000, a=4)
## show(head(ts, 5))

## Provides the uniform delayed-coordinate embedding vectors (Backward)
## data     <- DChaos::embedding(ts, m=5, lag=2, timelapse="FIXED")
## show(head(data, 5))

## Simulates tick-by-tick data (bid price) for Starbucks company
## ts       <- highfrequency::sbux
## show(head(ts, 5))

## Provides the non-uniform delayed-coordinate embedding vectors (Backward)
## data     <- DChaos::embedding(ts, m=3, lag=4, timelapse="VARIABLE")
## show(head(data, 5))
```

`gauss.sim` *Simulates time-series data from the Gauss map*

Description

This function simulates time-series data from the Gauss map considering the parameter set selected by the user. The initial condition is a random number between 0 and 1. Some initial conditions may lead to an unstable system that will tend to infinity.

Usage

```
gauss.sim(
  alpha = 6.2,
  beta = -0.5,
  s = 0,
  x0 = runif(1, 0, 1),
  n = 1000,
  n.start = 50
)
```

Arguments

`alpha`	a non-negative integer denoting the value of parameter `alpha` (Default 6.2).
`beta`	a non-negative integer denoting the value of parameter `beta` (Default -0.5).
`s`	a non-negative integer denoting the variance value of the error term. If $s = 0$ gives the standard deterministic map (Default 0).
`x0`	a non-negative integer denoting the initial condition (Default random number between 0 and 1).
`n`	a non-negative integer denoting the length (Default 1000).
`n.start`	a non-negative integer denoting the number of observations that will be discarded to ensure that the values are in the attractor (Default 50).

Value

A time-series data object generated from the Gauss map with or without an additive measurement noise term. This dataset could be useful for researchers interested in the field of chaotic dynamic systems and non-linear time series analysis and professors (and students) who teach (learn) courses related to those topics.

Note

This function provides also noisy time-series data from the deterministic gauss map adding an additive measurement noise term if $s > 0$. We have added to each time-series data a normal multinomial error term denoted by $\varepsilon_t \sim N\left(0, s\right)$ with different variance values (s). In this sense we have considered it appropriate to add a measurement noise term because most real-world observed time-series data are usually noise-contaminated signals, characterised by an erratic and persistent volatility in certain periods and there is almost always a source of noise linked to measurement errors in real-world datasets.

Author(s)

Julio E. Sandubete, Lorenzo Escot

References

Hilborn, R.C. 2004 Chaos and nonlinear dynamics: an introduction for scientists and engineers. Oxford, Univ. Press, New York.

Examples

```
## set.seed(34)
## Simulates time-series data from the deterministic gauss map
## with a chaotic behaviour.
## ts <- gauss.sim(alpha=6.2, beta=-0.5, s=0, n=1000)
##
## Simulates time-series data from the deterministic gauss map
## with a non-chaotic behaviour.
## ts <- gauss.sim(alpha=4.9, beta=-0.58, s=0, n=1000)
```

`henon.sim` *Simulates time-series data from the Henon map*

Description

This function simulates time-series data from the Henon map considering the parameter set selected by the user. The initial condition is a random number between -0.5 and 0.5. Some initial conditions may lead to an unstable system that will tend to infinity.

Usage

```
henon.sim(
  a = 1.4,
  b = 0.3,
  s = 0,
  x0 = runif(1, -0.5, 0.5),
  y0 = runif(1, -0.5, 0.5),
  n = 1000,
  n.start = 50
)
```

Arguments

`a`	a non-negative integer denoting the value of parameter `a` (Default 1.4).
`b`	a non-negative integer denoting the value of parameter `b` (Default 0.3).
`s`	a non-negative integer denoting the variance value of the error term. If $s = 0$ gives the standard deterministic map (Default 0).
`x0`	a non-negative integer denoting the initial condition of x-coordinate (Default random number between -0.5 and 0.5).
`y0`	a non-negative integer denoting the initial condition of y-coordinate (Default random number between -0.5 and 0.5).
`n`	a non-negative integer denoting the length (Default 1000).
`n.start`	a non-negative integer denoting the number of observations that will be discarded to ensure that the values are in the attractor (Default 50).

Value

A time-series data object generated from the Henon map with or without an additive measurement noise term. This dataset could be useful for researchers interested in the field of chaotic dynamic systems and non-linear time series analysis and professors (and students) who teach (learn) courses related to those topics.

Note

This function provides also noisy time-series data from the deterministic henon map adding an additive measurement noise term if $s > 0$. We have added to each time-series data a normal multinomial error term denoted by $\varepsilon_t \sim N\left(0, s\right)$ with different variance values (s). In this sense we have considered it appropriate to add a measurement noise term because most real-world observed time-series data are usually noise-contaminated signals, characterised by an erratic and persistent volatility in certain periods and there is almost always a source of noise linked to measurement errors in real-world datasets.

Author(s)

Julio E. Sandubete, Lorenzo Escot

References

Hénon, M. 1976 A two-dimensional mapping with a strange attractor. Communications in Mathematical Physics 50(1):69-77.

Examples

```
## set.seed(34)
## Simulates time-series data from the deterministic henon map
## with a chaotic behaviour.
ts <- henon.sim(a = 1.4, b = 0.3, s = 0, n = 1000)
##
## Simulates time-series data from the deterministic henon map
## with a non-chaotic behaviour.
ts <- henon.sim(a = 1.2, b = 0.1, s = 0, n = 1000)
```

jacobian.net — *Computes the partial derivatives from the best-fitted neural net model*

Description

This function computes analytically the partial derivatives from the best-fitted neural net model.

Usage

```
jacobian.net(
  model,
  data,
  m = 1:4,
  lag = 1:1,
  timelapse = c("FIXED", "VARIABLE"),
  h = 2:10,
  w0maxit = 100,
  wtsmaxit = 1e+06,
  pre.white = TRUE,
  trace = 1,
  seed.t = TRUE,
  seed = 56666459,
  ...
)
```

Arguments

`model`	a neural network model fitted using the `netfit` function.
`data`	a `vector`, a time-series object `ts` or `xts`, a `data.frame`, a `data.table` or a `matrix` depending on the method selected in `timelapse`.
`m`	a non-negative integer denoting a lower and upper bound for the embedding dimension (Default 1:4).
`lag`	a non-negative integer denoting a lower and upper bound for the the reconstruction delay (Default 1:1).
`timelapse`	a character denoting if the time-series data are sampled at uniform time-frequency e.g., 1-month, 1-day, 1-hour, 30-min, 5-min, 1-min and so on `FIXED` or non-uniform time-frequency which are not equally spaced in time `VARIABLE` (Default `FIXED`).
`h`	a non-negative integer denoting a lower and upper bound for the number of neurones (or nodes) in the single hidden layer (Default 2:10).
`w0maxit`	a non-negative integer denoting the maximum iterations to estimate the initial parameter vector of the neural net models (Default 100).
`wtsmaxit`	a non-negative integer denoting the maximum iterations to estimate the weights parameter vector of the neural net models (Default 1e6).

`pre.white`	a logical value denoting if the user wants to use as points to evaluate the partial derivatives the delayed vectors filtered by the neural net model chosen `TRUE` or not `FALSE` (Default `TRUE`).
`trace`	a binary value denoting if the user wants to print the output on the console `1` or not `0` (Default 1).
`seed.t`	a logical value denoting if the user wants to fix the seed `TRUE` or not `FALSE` (Default TRUE).
`seed`	a non-negative integer denoting the value of the seed selected if `seed.t = TRUE` (Default 56666459).
`...`	further arguments passed to or from `nnet` function.

Value

This function returns several objects considering the parameter set selected by the user. Partial derivatives are calculated analytically from the best-fitted neural net model. It also contains some useful information about the best-fitted feed-forward single hidden layer neural net model saved, the best set of weights found, the fitted values, the residuals obtained or the best embedding parameters set chosen. This function allows the R user uses the data previously obtained from the best-fitted neural network estimated by the `netfit` function if `model` is not empty. Otherwise `data` has to be specified.

Note

The main reason for using neural network models is not to look for the best predictive model but to estimate a model that captures the non-linear time dependence well enough and, additionally, allows us to obtain in an analytical way (instead of numerical) the jacobian functional of the unknown underlying generator system. The estimation of this jacobian or partial derivatives will later allow us to contrast our hypothesis of chaos estimating the Lyapunov exponents.

Author(s)

Julio E. Sandubete, Lorenzo Escot

References

Eckmann, J.P., Ruelle, D. 1985 Ergodic theory of chaos and strange attractors. Rev Mod Phys 57:617–656.

Gencay, R., Dechert, W.D. 1992 An algorithm for the n lyapunov exponents of an n-dimensional unknown dynamical system. Physica D 59(1):142–157.

Shintani, M., Linton, O. 2004 Nonparametric neural network estimation of Lyapunov exponents and a direct test for chaos. Journal of Econometrics 120(1):1-33.

Examples

```
## set.seed(34)
## Simulates time-series data from the Logistic map with chaos
## ts        <- DChaos::logistic.sim(n=1000, a=4)
## show(head(ts, 5))
## Computes analytically the partial derivatives from the best-fitted neural net model
## showed in the netfit example
## model    <- DChaos::netfit(ts, m=1:4, lag=1:3, timelapse="FIXED", h=2:10)
## jacobian <- DChaos::jacobian.net(model=model)
## summary(jacobian)

## Partial derivatives are calculated analytically without setting previously any neural net model
## jacobian <- DChaos::jacobian.net(data=ts, m=3:3, lag=1:1, timelapse="FIXED", h=2:10)
## summary(jacobian)
```

`logistic.sim` *Simulates time-series data from the Logistic map*

Description

This function simulates time-series data from the Logistic map considering the parameter set selected by the user. The initial condition is a random number between 0 and 1. Some initial conditions may lead to an unstable system that will tend to infinity.

Usage

```
logistic.sim(a = 4, s = 0, x0 = runif(1, 0, 1), n = 1000, n.start = 50)
```

Arguments

`a`	a non-negative integer denoting the value of parameter `a` (Default 4).
`s`	a non-negative integer denoting the variance value of the error term. If $s = 0$ gives the standard deterministic map (Default 0).
`x0`	a non-negative integer denoting the initial condition (Default random number between 0 and 1).
`n`	a non-negative integer denoting the length (Default 1000).
`n.start`	a non-negative integer denoting the number of observations that will be discarded to ensure that the values are in the attractor (Default 50).

Value

A time-series data object generated from the Logistic map with or without an additive measurement noise term. This dataset could be useful for researchers interested in the field of chaotic dynamic systems and non-linear time series analysis and professors (and students) who teach (learn) courses related to those topics.

Note

This function provides also noisy time-series data from the deterministic logistic map adding an additive measurement noise term if $s > 0$. We have added to each time-series data a normal multinomial error term denoted by $\varepsilon_t \sim N(0, s)$ with different variance values (s). In this sense we have considered it appropriate to add a measurement noise term because most real-world observed time-series data are usually noise-contaminated signals, characterised by an erratic and persistent volatility in certain periods and there is almost always a source of noise linked to measurement errors in real-world datasets.

Author(s)

Julio E. Sandubete, Lorenzo Escot

References

May, R.M. 1976 Simple mathematical models with very complicated dynamics. Nature (261):459-467.

Examples

```
## set.seed(34)
## Simulates time-series data from the deterministic logistic map
## with a chaotic behaviour.
## ts <- logistic.sim(a=4, s=0, n=1000)
##
## Simulates time-series data from the deterministic logistic map
## with a non-chaotic behaviour.
## ts <- logistic.sim(a=3.2, s=0, n=1000)
```

`lyapunov` *Estimates the Lyapunov exponent through several methods*

Description

This is an all-in-one function. It provides, at the same time, the delayed-coordinate embedding vector (`embedding`), estimates the best neural net model (`netfit`), calculates the partial derivatives directly from the chosen neural network model (`jacobian.net`). Finally, this function estimates both the largest Lyapunov exponent through the Norma-2 procedure (`lyapunov.max`) and the Lyapunov exponent spectrum through the QR decomposition procedure (`lyapunov.spec`) taking into account the full sample and three different methods of subsampling by blocks.

Usage

```
lyapunov(
  data,
  m = 1:4,
  lag = 1:1,
  timelapse = c("FIXED", "VARIABLE"),
  h = 2:10,
  w0maxit = 100,
  wtsmaxit = 1e+06,
  pre.white = TRUE,
  lyapmethod = c("SLE", "LLE", "ALL"),
  blocking = c("BOOT", "NOVER", "EQS", "FULL", "ALL"),
  B = 1000,
  trace = 1,
  seed.t = TRUE,
  seed = 56666459,
  doplot = TRUE,
  ...
)
```

Arguments

`data`	a `vector`, a time-series object `ts` or `xts`, a `data.frame`, a `data.table` or a `matrix` depending on the method selected in `timelapse`.
`m`	a non-negative integer denoting a lower and upper bound for the embedding dimension (Default 1:4).
`lag`	a non-negative integer denoting a lower and upper bound for the the reconstruction delay (Default 1:1).
`timelapse`	a character denoting if the time-series data are sampled at uniform time-frequency e.g., 1-month, 1-day, 1-hour, 30-min, 5-min, 1-min and so on `FIXED` or non-uniform time-frequency which are not equally spaced in time `VARIABLE` (Default `FIXED`).
`h`	a non-negative integer denoting a lower and upper bound for the number of neurones (or nodes) in the single hidden layer (Default 2:10).
`w0maxit`	a non-negative integer denoting the maximum iterations to estimate the initial parameter vector of the neural net models (Default 100).
`wtsmaxit`	a non-negative integer denoting the maximum iterations to estimate the weights parameter vector of the neural net models (Default 1e6).
`pre.white`	a logical value denoting if the user wants to use as points to evaluate the partial derivatives the delayed vectors filtered by the neural net model chosen `TRUE` or not `FALSE` (Default `TRUE`).
`lyapmethod`	a character denoting the procedure chosen to estimate the Lyapunov exponent. If `LLE` has been selected the function will estimate only the largest Lyapunov exponent through the Norma-2 method. If `SLE` has been selected the function will estimate the Lyapunov exponent spectrum through the QR decomposition. Otherwise `ALL` has to be specified. In this case the function will estimate the Lyapunov exponent considering both procedures (Default `SLE`).
`blocking`	a character denoting the blocking method chosen for figuring out the Lyapunov exponent. Available options are `FULL` if the user considers the full sample, `NOVER` if the user considers the non-overlapping sample, `EQS` if the user considers the equally spaced sample, `BOOT` if the user considers the bootstrap sample or `ALL` if the user considers all of them (Default `BOOT`).

`B`	a non-negative integer denoting the number of bootstrap iterations (Default 1000).
`trace`	a binary value denoting if the user wants to print the output on the console `1` or not `0` (Default 1).
`seed.t`	a logical value denoting if the user wants to fix the seed `TRUE` or not `FALSE` (Default TRUE).
`seed`	a non-negative integer denoting the value of the seed selected if `seed.t = TRUE` (Default 56666459).
`doplot`	a logical value denoting if the user wants to draw a plot `TRUE` or not `FALSE`. If it is `TRUE` the evolution of the Lyapunov exponent values are represented for the whole period considering the blocking method chosen by the user. It shows as many graphs as embedding dimensions have been considered (Default `TRUE`).
`...`	further arguments passed to or from `nnet` function.

Value

This function returns several objects considering the parameter set selected by the user. The largest Lyapunov exponent (Norma-2 procedure) and the Lyapunov exponent spectrum (QR decomposition procedure) by each blocking method are estimated. It also contains some useful information about the estimated jacobian, the best-fitted feed-forward single hidden layer neural net model, the best set of weights found, the fitted values, the residuals obtained, the best embedding parameters set chosen, the sample size or the block length considered by each blocking method. This function provides the standard error, the z test value and the p-value for testing the null hypothesis $H0 : \lambda_k > 0 for k = 1, 2, 3, \ldots, m$. Reject the null hypothesis $H_0$ means lack of chaotic behaviour. That is, the data-generating process does not have a chaotic attractor because of it does not show the property of sensitivity to initial conditions.

Note

We have considered it appropriate to incorporate a function that unifies the whole process to make it easier and more intuitive for the R users. The DChaos package provides several ways to figure out robustly the neural net estimator of the k-th Lyapunov exponent. Particularly, there are 8 functions (one for each procedure and blocking method) which estimate the Lyapunov exponents consistently. Hence the DChaos package allows the R users to choose between two different procedures to obtain the neural net estimator of the k-th Lyapunov exponent and four ways of subsampling by blocks: full sample, non-overlapping sample, equally spaced sample and bootstrap sample. The blocking methods what they do is to split the time-series data into several blocks by estimating a Lyapunov exponent for each subsample. If the R users choose the non-overlapping sample (`blocking = "NOVER"`), the equally spaced sample (`blocking = "EQS"`) or the bootstrap sample (`blocking = "BOOT"`) the mean and median values of the Lyapunov exponent for each block or subsample are saved. By default we recommend using the median value as it is more robust to the presence of outliers. Notice that the parameter `B` will only be considered if the R users choose the bootstrap blocking method.

Author(s)

Julio E. Sandubete, Lorenzo Escot

References

Ellner, S., Gallant, A., McCaffrey, D., Nychka, D. 1991 Convergence rates and data requirements for jacobian-based estimates of lyapunov exponents from data. Physics Letters A 153(6):357-363.

McCaffrey, D.F., Ellner, S., Gallant, A.R., Nychka, D.W. 1992 Estimating the lyapunov exponent of a chaotic system with nonparametric regression. Journal of the American Statistical Association 87(419):682-695.

Nychka, D., Ellner, S., Gallant, A.R., McCaffrey, D. 1992 Finding chaos in noisy systems. Journal of the Royal Statistical Society 54(2):399-426.

Whang, Y.J., Linton, O. 1999 The asymptotic distribution of nonparametric estimates of the lyapunov exponent for stochastic time series. Journal of Econometrics 91(1):1-42.

Shintani, M., Linton, O. 2004 Nonparametric neural network estimation of Lyapunov exponents and a direct test for chaos. Journal of Econometrics 120(1):1-33.

See Also

`lyapunov.max`, `lyapunov.spec`

Examples

```
## set.seed(34)
## Simulates time-series data from the Logistic map with chaos
## ts        <- DChaos::logistic.sim(n=1000, a=4)
## show(head(ts, 5))

## Provides the Lyapunov exponent spectrum by the QR decomposition procedure considering the
## bootstrap blocking method directly from the Logistic map with chaos simulated.
## exponent <- DChaos::lyapunov(ts, m=3:3, lag=1:1, timelapse="FIXED", h=2:10, w0maxit=100,
##                      wtsmaxit=1e6, pre.white=TRUE, lyapmethod="SLE", blocking="ALL",
##                      B=100, trace=1, seed.t=TRUE, seed=56666459, doplot=FALSE))
## summmary(exponent)
```

`lyapunov.max` *Estimates the largest Lyapunov exponent*

Description

This function estimates the largest Lyapunov exponent through the Norma-2 procedure based on the partial derivatives computed by the `jacobian.net` function.

Usage

```
lyapunov.max(
  data,
  blocking = c("BOOT", "NOVER", "EQS", "FULL"),
  B = 1000,
  doplot = TRUE
)
```

Arguments

`data`	should be a `jacobian` object containing the partial derivatives computed by the `jacobian.net` function.
`blocking`	a character denoting the blocking method chosen for figuring out the largest Lyapunov exponent through the Norma-2 procedure. Available options are `FULL` if the user considers the full sample, `NOVER` if the user considers the non-overlapping sample, `EQS` if the user considers the equally spaced sample or `BOOT` if the user considers the bootstrap sample (Default `BOOT`).
`B`	a non-negative integer denoting the number of bootstrap iterations (Default 1000).
`doplot`	a logical value denoting if the user wants to draw a plot `TRUE` or not `FALSE`. If it is `TRUE` the evolution of the Lyapunov exponent values are represented for the whole period considering the blocking method chosen by the user. It shows as many graphs as embedding dimensions have been considered (Default `TRUE`).

Value

This function returns several objects considering the parameter set selected by the user. The largest Lyapunov exponent considering the Norma-2 procedure by each blocking method are estimated. It also contains some useful information about the estimated jacobian, the best-fitted feed-forward single hidden layer neural net model, the best set of weights found, the fitted values, the residuals obtained, the best embedding parameters set chosen, the sample size or the block length considered by each blocking method. This function provides the standard error, the z test value and the p-value for testing the null hypothesis $H0 : \lambda_k > 0 fork = 1$ (largest). Reject the null hypothesis $H_0$ means lack of chaotic behaviour. That is, the data-generating process does not have a chaotic attractor because of it does not show the property of sensitivity to initial conditions.

Note

The DChaos package provides several ways to figure out robustly the neural net estimator of the k-th Lyapunov exponent. On the one hand if the R users have previously obtained the partial derivatives from the `jacobian.net` function they can apply directly the function `lyapunov.spec` which estimates the Lyapunov exponent spectrum taking into account the QR decomposition procedure. They can also use the function `lyapunov.max` which estimates only the largest Lyapunov exponent considering the Norma-2 procedure. Hence the DChaos package allows the R users to choose between two different procedures to obtain the neural net estimator of the k-th Lyapunov exponent and four ways of subsampling by blocks: full sample, non-overlapping sample, equally spaced sample and bootstrap sample. The blocking methods what they do is to split the time-series data into several blocks by estimating a Lyapunov exponent for each subsample. If the R users choose the non-overlapping sample (`blocking = "NOVER"`), the equally spaced sample (`blocking = "EQS"`) or the bootstrap sample (`blocking = "BOOT"`) the mean and median values of the Lyapunov exponent for each block or subsample are saved. By default we recommend using the median value as it is more robust to the presence of outliers. Notice that the parameter `B` will only be considered if the R users choose the bootstrap blocking method.

Author(s)

Julio E. Sandubete, Lorenzo Escot

References

Ellner, S., Gallant, A., McCaffrey, D., Nychka, D. 1991 Convergence rates and data requirements for jacobian-based estimates of lyapunov exponents from data. Physics Letters A 153(6):357-363.

McCaffrey, D.F., Ellner, S., Gallant, A.R., Nychka, D.W. 1992 Estimating the lyapunov exponent of a chaotic system with nonparametric regression. Journal of the American Statistical Association 87(419):682-695.

Nychka, D., Ellner, S., Gallant, A.R., McCaffrey, D. 1992 Finding chaos in noisy systems. Journal of the Royal Statistical Society 54(2):399-426.

Whang, Y.J., Linton, O. 1999 The asymptotic distribution of nonparametric estimates of the lyapunov exponent for stochastic time series. Journal of Econometrics 91(1):1-42.

Shintani, M., Linton, O. 2004 Nonparametric neural network estimation of Lyapunov exponents and a direct test for chaos. Journal of Econometrics 120(1):1-33.

Examples

```
## set.seed(34)
## Simulates time-series data from the Logistic map with chaos
## ts        <- DChaos::logistic.sim(n=1000, a=4)
## show(head(ts, 5))

## Provides the largest Lyapunov exponent by the Norma-2 procedure considering the
## bootstrap blocking method from the best-fitted neural net model and the partial
## derivatives showed in the jacobian.net example.
## jacobian <- DChaos::jacobian.net(data=ts, m=3:3, lag=1:1, timelapse="FIXED", h=2:10)
## summary(jacobian)
## exponent <- DChaos::lyapunov.max(data=jacobian, blocking="BOOT", B=100, doplot=FALSE)
## summary(exponent)
```

`lyapunov.spec` *Estimates the Lyapunov exponent spectrum*

Description

This function estimates the Lyapunov exponent spectrum through the QR decomposition procedure based on the partial derivatives computed by the `jacobian.net` function.

Usage

```
lyapunov.spec(
  data,
  blocking = c("BOOT", "NOVER", "EQS", "FULL"),
  B = 1000,
  doplot = TRUE
)
```

Arguments

`data`	should be a `jacobian` object containing the partial derivatives computed by the `jacobian.net` function.
`blocking`	a character denoting the blocking method chosen for figuring out the Lyapunov exponent spectrum through the QR decomposition procedure. Available options are `FULL` if the user considers the full sample, `NOVER` if the user considers the non-overlapping sample, `EQS` if the user considers the equally spaced sample or `BOOT` if the user considers the bootstrap sample (Default `BOOT`).
`B`	a non-negative integer denoting the number of bootstrap iterations (Default 1000).
`doplot`	a logical value denoting if the user wants to draw a plot `TRUE` or not `FALSE`. If it is `TRUE` the evolution of the Lyapunov exponent values are represented for the whole period considering the blocking method chosen by the user. It shows as many graphs as embedding dimensions have been considered (Default `TRUE`).

Value

This function returns several objects considering the parameter set selected by the user. The Lyapunov exponent spectrum considering the QR decomposition procedure by each blocking method are estimated. It also contains some useful information about the estimated jacobian, the best-fitted feed-forward single hidden layer neural net model, the best set of weights found, the fitted values, the residuals obtained, the best embedding parameters set chosen, the sample size or the block length considered by each blocking method. This function provides the standard error, the z test value and the p-value for testing the null hypothesis $H0 : \lambda_k > 0 for k = 1, 2, 3, \ldots, m$ (full spectrum). Reject the null hypothesis $H_0$ means lack of chaotic behaviour. That is, the data-generating process does not have a chaotic attractor because of it does not show the property of sensitivity to initial conditions.

Note

The DChaos package provides several ways to figure out robustly the neural net estimator of the k-th Lyapunov exponent. On the one hand if the R users have previously obtained the partial derivatives from the `jacobian.net` function they can apply directly the function `lyapunov.spec` which estimates the Lyapunov exponent spectrum taking into account the QR decomposition procedure. They can also use the function `lyapunov.max` which estimates only the largest Lyapunov exponent considering the Norma-2 procedure. Hence the DChaos package allows the R users to choose between two different procedures to obtain the neural net estimator of the k-th Lyapunov exponent and four ways of subsampling by blocks: full sample, non-overlapping sample, equally spaced sample and bootstrap sample. The blocking methods what they do is to split the time-series data into several blocks by estimating a Lyapunov exponent for each subsample. If the R users choose the non-overlapping sample (`blocking = "NOVER"`), the equally spaced sample (`blocking = "EQS"`) or the bootstrap sample (`blocking = "BOOT"`) the mean and median values of the Lyapunov exponent for each block or subsample are saved. By default we recommend using the median value as it is more robust to the presence of outliers. Notice that the parameter `B` will only be considered if the R users choose the bootstrap blocking method.

Author(s)

Julio E. Sandubete, Lorenzo Escot

References

Ellner, S., Gallant, A., McCaffrey, D., Nychka, D. 1991 Convergence rates and data requirements for jacobian-based estimates of lyapunov exponents from data. Physics Letters A 153(6):357-363.

McCaffrey, D.F., Ellner, S., Gallant, A.R., Nychka, D.W. 1992 Estimating the lyapunov exponent of a chaotic system with nonparametric regression. Journal of the American Statistical Association 87(419):682-695.

Nychka, D., Ellner, S., Gallant, A.R., McCaffrey, D. 1992 Finding chaos in noisy systems. Journal of the Royal Statistical Society 54(2):399-426.

Whang, Y.J., Linton, O. 1999 The asymptotic distribution of nonparametric estimates of the lyapunov exponent for stochastic time series. Journal of Econometrics 91(1):1-42.

Shintani, M., Linton, O. 2004 Nonparametric neural network estimation of Lyapunov exponents and a direct test for chaos. Journal of Econometrics 120(1):1-33.

Examples

```
## set.seed(34)
## Simulates time-series data from the Logistic map with chaos
## ts       <- DChaos::logistic.sim(n=1000, a=4)
## show(head(ts, 5))

## Provides the Lyapunov exponent spectrum by the QR decomposition procedure considering the
## bootstrap blocking method from the best-fitted neural net model and the partial
## derivatives showed in the jacobian.net example.
## jacobian <- DChaos::jacobian.net(data=ts, m=3:3, lag=1:1, timelapse="FIXED", h=2:10)
## summary(jacobian)
## exponent <- DChaos::lyapunov.spec(data=jacobian, blocking="BOOT", B=100, doplot=FALSE)
## summary(exponent)
```

`netfit` *Fits any standard feedforward neural net model from time-series data*

Description

This function fits any standard feedforward neural net model from time-series data.

Usage

```
netfit(
  serie,
  m = 1:4,
  lag = 1:1,
  timelapse = c("FIXED", "VARIABLE"),
  h = 2:10,
  w0maxit = 100,
  wtsmaxit = 1e+06,
  pre.white = TRUE,
  trace = 1,
  seed.t = TRUE,
  seed = 56666459,
  ...
)
```

Arguments

`serie`	a `vector`, a time-series object `ts` or `xts`, a `data.frame`, a `data.table` or a `matrix` depending on the method selected in `timelapse`.
`m`	a non-negative integer denoting a lower and upper bound for the embedding dimension (Default 1:4).
`lag`	a non-negative integer denoting a lower and upper bound for the the reconstruction delay (Default 1:1).
`timelapse`	a character denoting if the time-series data are sampled at uniform time-frequency e.g., 1-month, 1-day, 1-hour, 30-min, 5-min, 1-min and so on `FIXED` or non-uniform time-frequency which are not equally spaced in time `VARIABLE` (Default `FIXED`).
`h`	a non-negative integer denoting a lower and upper bound for the number of neurones (or nodes) in the single hidden layer (Default 2:10).
`w0maxit`	a non-negative integer denoting the maximum iterations to estimate the initial parameter vector of the neural net models (Default 100).
`wtsmaxit`	a non-negative integer denoting the maximum iterations to estimate the weights parameter vector of the neural net models (Default 1e6).
`pre.white`	a logical value denoting if the user wants to use as points to evaluate the partial derivatives the delayed vectors filtered by the neural net model chosen `TRUE` or not `FALSE` (Default `TRUE`).
`trace`	a binary value denoting if the user wants to print the output on the console `1` or not `0` (Default 1).
`seed.t`	a logical value denoting if the user wants to fix the seed `TRUE` or not `FALSE` (Default TRUE).
`seed`	a non-negative integer denoting the value of the seed selected if `seed.t = TRUE` (Default 56666459).
`...`	further arguments passed to or from `nnet` function.

Value

This function returns several objects considering the parameter set selected by the user. The best-fitted feed-forward single hidden layer neural net model is saved. It also contains some useful information about the best set of weights found, the fitted values, the residuals obtained or the best embedding parameters set chosen. The best 10 models are displayed on the console. The first column is the neural net number, the second column is the embedding dimension, the third column is the lag or reconstruction delay considered, the fourth column is the number of neurones (or nodes) in the single hidden layer and the fifth column is the Bayesian Information Criterion (BIC) value corresponding to that neural net. Notice that the neural net models are sorted from lowest to highest BIC values.

Note

The process of adjustment to a neural net model often suffers from being trapped in local optima and different initialization strategies should be taken into account. For this reason the function `w0.net` have been implemented. This function estimates previously the initial parameter vector of the neural net model being able to set the maximum number of iterations that the user wants to obtain setting `w0maxit`. In addition, by default the neural network estimation is initialized with a fixed seed denoted by `seed.t=TRUE` with a value equal to `seed=56666459`. The R user can let the seed be fixed either randomly by `seed.t=FALSE` or even fix other value of the seed to be able to replicate the results obtained.

Author(s)

Julio E. Sandubete, Lorenzo Escot

References

Ripley, B.D. 1996 Pattern Recognition and Neural Networks. Cambridge.

Venables, W.N., Ripley, B.D. 2002 Modern Applied Statistics with S. Fourth edition. Springer.

Hornik, K., Stinchcombe, M., White, H. 1989 Multilayer feedforward networks are universal approximators. Neural Networks 2(5):359-366.

Examples

```
## set.seed(34)
## Simulates time-series data from the Logistic map with chaos
## ts        <- DChaos::logistic.sim(n=1000, a=4)
## show(head(ts, 5))

## Provides the best-fitted neural network models for certain parameter set
## model    <- DChaos::netfit(ts, m=1:4, lag=1:3, timelapse="FIXED", h=2:10)
## summary(model)
```

`rossler.sim` *Simulates time-series data from the Rossler system*

Description

This function simulates time-series data from the Rossler system considering the parameter set selected by the user. The initial condition is a random number from the normal distribution with mean equal to 0 and variance equal to 1. Some initial conditions may lead to an unstable system that will tend to infinity.

Usage

```
rossler.sim(
  a = 0.2,
  b = 0.2,
  c = 5.7,
  s = 0,
  x0 = rnorm(1),
  y0 = rnorm(1),
  z0 = rnorm(1),
  time = seq(0, 100, 0.01),
  n.start = 50
)
```

Arguments

`a`	a non-negative integer denoting the value of parameter `a` (Default 0.2).
`b`	a non-negative integer denoting the value of parameter `b` (Default 0.2).
`c`	a non-negative integer denoting the value of parameter `c` (Default 5.7).
`s`	a non-negative integer denoting the variance value of the error term. If $s = 0$ gives the standard deterministic map (Default 0).
`x0`	a non-negative integer denoting the initial condition of x-coordinate (Default random number from the normal distribution).
`y0`	a non-negative integer denoting the initial condition of y-coordinate (Default random number from the normal distribution).
`z0`	a non-negative integer denoting the initial condition of z-coordinate (Default random number from the normal distribution).
`time`	a numeric vector denoting the time-lapse and the time-step (Default time-lapse equal to 10000 with a time-step of 0.01 seconds)
`n.start`	a non-negative integer denoting the number of observations that will be discarded to ensure that the values are in the attractor (Default 50).

Value

A time-series data object generated from the Rossler system with or without an additive measurement noise term. This dataset could be useful for researchers interested in the field of chaotic dynamic systems and non-linear time series analysis and professors (and students) who teach (learn) courses related to those topics.

Note

This function provides also noisy time-series data from the deterministic rossler system adding an additive measurement noise term if $s > 0$. We have added to each time-series data a normal multinomial error term denoted by $\varepsilon_t \sim N\left(0, s\right)$ with different variance values (s). In this sense we have considered it appropriate to add a measurement noise term because most real-world observed time-series data are usually noise-contaminated signals, characterised by an erratic and persistent volatility in certain periods and there is almost always a source of noise linked to measurement errors in real-world datasets. It has been implemented the classical Runge–Kutta method (RK4) in order to generate time-series data from continuous-time dynamical system as the Rossler system.

Author(s)

Julio E. Sandubete, Lorenzo Escot

References

Rössler, O. 1976 An equation for continuous chaos. Physics Letters A 57(5):397-398.

Examples

```
## set.seed(34)
## Simulates time-series data from the deterministic rossler system
## with a chaotic behaviour.
ts <- rossler.sim(a = 0.2, b = 0.2, c = 5.7, s = 0, time = seq(0, 100, 0.1))
##
## Simulates time-series data from the deterministic rossler system
## with a non-chaotic behaviour.
ts <- rossler.sim(a = 0.1, b = 0.1, c = 7, s = 0, time = seq(0, 100, 0.1))
```

`summary.lyapunov` *Summary method for a lyapunov object*

Description

summary method for class "lyapunov".

Usage

```
## S3 method for class 'lyapunov'
summary(object, ...)
```

Arguments

`object`	an object of class "lyapunov" provided by `lyapunov.max`, `lyapunov.spec` or `lyapunov` functions.
`...`	further arguments passed to or from other methods.

Value

This function `summary.lyapunov` computes and returns a list of summary statistics of the results given in a `lyapunov` object using the components (list elements) from its argument.

Author(s)

Julio E. Sandubete, Lorenzo Escot

Examples

```
## set.seed(34)
## Simulates time-series data from the Logistic map with chaos
## ts       <- DChaos::logistic.sim(n=1000, a=4)
## show(head(ts, 5))

## Summary method for a lyapunov object (only 1 method)
## jacobian <- DChaos::jacobian.net(data=ts, m=3:3, lag=1:1, timelapse="FIXED", h=2:10)
## exponent <- DChaos::lyapunov.spec(data=jacobian, blocking="BOOT", B=100, doplot=FALSE)
## summary(exponent)

## Summary method for a lyapunov object (> 1 method)
## exponent <- DChaos::lyapunov(ts, m=3:3, lag=1:1, timelapse="FIXED", h=2:10, w0maxit=100,
##                     wtsmaxit=1e6, pre.white=TRUE, lyapmethod="SLE", blocking="ALL",
##                     B=100, trace=1, seed.t=TRUE, seed=56666459, doplot=FALSE))
## summmary(exponent)
```

`w0.net` *Estimates the initial parameter vector of the neural net model*

Description

This function estimates the initial parameter vector of the neural net model.

Usage

```
w0.net(
  x,
  y,
  m = 2,
  h = 2,
  rangx = 1/max(abs(x)),
  w0maxit = 100,
  seed.t = TRUE,
  seed = 56666459,
  ...
)
```

Arguments

x	a `matrix` or a `data.frame` denoting the explanatory variables.
y	a `vector`, a `matrix` or a `data.frame` denoting the response variable.
m	a non-negative integer denoting the embedding dimension (Default 2).
h	a non-negative integer denoting the number of neurones (or nodes) in the single hidden layer (Default 2).
rangx	a non-negative integer denoting the range of the explanatory variables (Default 1/max(abs(x)).

`w0maxit`	a non-negative integer denoting the maximum iterations to estimate the initial parameter vector of the neural net models (Default 100).
`seed.t`	a logical value denoting if the user wants to fix the seed `TRUE` or not `FALSE` (Default TRUE).
`seed`	a non-negative integer denoting the value of the seed selected if `seed.t = TRUE` (Default 56666459).
`...`	further arguments passed to or from `nnet` function.

Value

The optimal initial parameter vector of the neural net model considering the argument set selected by the user.

Note

The process of adjustment to a neural network often suffers from being trapped in local optima and different initialization strategies should be taken into account. For this reason the function `w0.net` have been implemented. This function estimates previously the initial parameter vector of the neural net model being able to set the maximum number of iterations that the user wants to obtain setting `w0maxit`. In addition, by default the neural network estimation is initialized with a fixed seed denoted by `seed.t=TRUE` with a value equal to `seed=56666459`. The R user can let the seed be fixed either randomly by `seed.t=FALSE` or even fix other value of the seed to be able to replicate the results obtained.

Author(s)

Julio E. Sandubete, Lorenzo Escot

References

Ripley, B.D. 1996 Pattern Recognition and Neural Networks. Cambridge.

Venables, W.N., Ripley, B.D. 2002 Modern Applied Statistics with S. Fourth edition. Springer.

Hornik, K., Stinchcombe, M., White, H. 1989 Multilayer feedforward networks are universal approximators. Neural Networks 2(5):359-366.

Index