Package 'EnvCpt'

October 12, 2022

Type Package

Title Detection of Structural Changes in Climate and Environment Time Series

Version 1.1.3

Date 2021-03-29

Maintainer Rebecca Killick <r.killick@lancs.ac.uk>

URL https://github.com/rkillick/EnvCpt/

Description
Tools for automatic model selection and diagnostics for Climate and Environmental data. In particular the envcpt() function does automatic model selection between a variety of trend, changepoint and autocorrelation models. The envcpt() function should be your first port of call.

Depends R(>= 3.3), changepoint(>= 2.2.2), grDevices, MASS, methods, stats, utils, zoo

Suggests testthat

License GPL

LazyLoad yes

NeedsCompilation yes

Author Rebecca Killick [aut, cre],
Claudie Beaulieu [aut],
Simon Taylor [aut],
Harjit Hullait [aut]

Repository CRAN

Date/Publication 2021-03-29 12:32:16 UTC

R topics documented:

EnvCpt-package *Detection of Structural Changes in Climate and Environment Time Series*

Description

Tools for automatic model selection and diagnostics for Climate and Environmental data. In particular the `envcpt()` function does automatic model selection between a variety of trend, changepoint and autocorrelation models. The `envcpt()` function should be your first port of call.

Details

Package:	EnvCpt
Type:	Package
Version:	1.1.3
Date:	2021-03-29
License:	GPL
LazyLoad:	yes

Author(s)

Rebecca Killick <r.killick@lancs.ac.uk>, Claudie Beaulieu <c.beaulieu@soton.ac.uk>, Simon Taylor <s.taylor2@lancs.ac.uk>, Harjit Hullait <h.hullait@lancs.ac.uk>.

Maintainer: Rebecca Killick <r.killick@lancs.ac.uk>

References

EnvCpt Algorithm: Beaulieu, C, Killick, R (2018+) Distinguishing trends and shifts from memory in climate data.

PELT Algorithm: Killick R, Fearnhead P, Eckley IA (2012) Optimal detection of changepoints with a linear computational cost, *JASA* **107(500)**, 1590–1598

See Also

`envcpt`,`AIC.envcpt`,`plot.envcpt`

Examples

```
## Not run:
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,5,1))
out=envcpt(x) # run all models with default values
```

```
out[[1]] # first row is twice the negative log-likelihood for each model
         # second row is the number of parameters
AIC(out) # returns AIC for each model.
which.min(AIC(out)) # gives meancpt (model 2) as the best model fit.
out$meancpt # gives the model fit for the meancpt model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # gives meancpt (model 2) as the best model fit too.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

set.seed(10)
x=c(0.01*(1:100),1.5-0.02*((101:250)-101))+rnorm(250,0,0.2)
out=envcpt(x,minseglen=10) # run all models with a minimum of 10 observations between changes
AIC(out) # returns the AIC for each model
which.min(AIC(out)) # gives trendcpt (model 8) as the best model fit.
out$trendcpt # gives the model fit for the trendcpt model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # gives trendcpt (model 8) as the best model fit too.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

set.seed(100)
x=arima.sim(model=list(ar=c(0.7,0.2)),n=500)+0.01*(1:500)
out=envcpt(x,models=c(3:6,9:12)) # runs a subset of models (those with AR components)
AIC(out) # returns the AIC for each model
which.min(AIC(out)) # gives trendar2 (model 10) as the best model fit.
out$trendar2 # gives the model fit for the trendar2 model. Notice that the trend is tiny but does
# produce a significantly better fit than the meanar2 model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # best fit is trendar2 (model 10) again.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

## End(Not run)
```

`AIC.envcpt` *Computes the AIC/BIC measure for output from the envcpt function.*

Description

Uses the likelihood and number of parameters from the output of the `envcpt` function and calculates the AIC/BIC measure for each model.

Usage

```
## S3 method for class 'envcpt'
AIC(object,...,k=2)
## S3 method for class 'envcpt'
BIC(object,...)
```

Arguments

object	A list produced as output from the envcpt function. In essence any list where the first element contains a matrix whose first row is twice the negative log-likelihood and second row is the number of parameters. Columns are different models to compare.
...	Used to pass the length of the data to the BIC function as argument n.
k	numeric, the penalty per parameter to be used: the default k=2 is the classical AIC. The value log(n) is hard coded into BIC.

Details

Calculates the AIC defined as -2*<log-likelihood> + 2*<number of parameters> or the BIC as -2*<log-likelihood> + log(n)*<number of parameters>. When comparing models the smaller the AIC/BIC the better the fit.

Value

Vector of AIC/BIC values the same length as the number of columns in the first entry to the input list (length 12 if output from envcpt is used). The column names from the envcpt output are preserved to give clear indication on models.

Author(s)

Simon Taylor and Rebecca Killick

See Also

envcpt

Examples

```
## Not run:
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,5,1))
out=envcpt(x) # run all models with default values
out[[1]] # first row is twice the negative log-likelihood for each model
         # second row is the number of parameters
AIC(out) # returns AIC for each model.
which.min(AIC(out)) # gives meancpt (model 2) as the best model fit.
out$meancpt # gives the model fit for the meancpt model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # gives meancpt (model 2) as the best model fit too.
```

```
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

set.seed(10)
x=c(0.01*(1:100),1.5-0.02*((101:250)-101))+rnorm(250,0,0.2)
out=envcpt(x,minseglen=10) # run all models with a minimum of 10 observations between changes
AIC(out) # returns the AIC for each model
which.min(AIC(out)) # gives trendcpt (model 8) as the best model fit.
out$trendcpt # gives the model fit for the trendcpt model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # gives trendcpt (model 8) as the best model fit too.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

set.seed(100)
x=arima.sim(model=list(ar=c(0.7,0.2)),n=500)+0.01*(1:500)
out=envcpt(x,models=c(3:6,9:12)) # runs a subset of models (those with AR components)
AIC(out) # returns the AIC for each model
which.min(AIC(out)) # gives trendar2 (model 10) as the best model fit.
out$trendar2 # gives the model fit for the trendar2 model. Notice that the trend is tiny but does
# produce a significantly better fit than the meanar2 model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # best fit is trendar2 (model 10) again.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

## End(Not run)
```

`AICweights` *Computes the AIC weights for output from the envcpt function.*

Description

Uses the likelihood and number of parameters from the output of the `envcpt` function and calculates the AICweights for each model fitted relative to the model with the minimum AIC.

Usage

```
## S3 method for class 'envcpt'
AICweights(object)
```

Arguments

`object`	A list produced as output from the `envcpt` function. In essence any list where the first element contains a matrix whose first row is twice the negative log-

likelihood and second row is the number of parameters. Columns are different models to compare.

Details

Calculates the AICweights defined as

$$w_i = \frac{\exp -0.5\Delta_i}{\sum_r \exp -0.5\Delta_r}$$

where the summation over r is across all models considered and Δ_i is the difference between the AIC value for model i and the best model.

Value

Vector of AICweights the same length as the number of columns in the first entry to the input list (length 12 if output from envcpt where all models are considered). The column names from the envcpt output are preserved to give clear indication on models.

Author(s)

Rebecca Killick

See Also

envcpt

Examples

```
## Not run:
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,5,1))
out=envcpt(x) # run all models with default values
out[[1]] # first row is twice the negative log-likelihood for each model
         # second row is the number of parameters
AIC(out) # returns AIC for each model.
which.min(AIC(out)) # gives meancpt (model 2) as the best model fit.
out$meancpt # gives the model fit for the meancpt model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # gives meancpt (model 2) as the best model fit too.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

set.seed(10)
x=c(0.01*(1:100),1.5-0.02*((101:250)-101))+rnorm(250,0,0.2)
out=envcpt(x,minseglen=10) # run all models with a minimum of 10 observations between changes
AIC(out) # returns the AIC for each model
which.min(AIC(out)) # gives trendcpt (model 8) as the best model fit.
out$trendcpt # gives the model fit for the trendcpt model.
AICweights(out) # gives the AIC weights for each model
```

```
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # gives trendcpt (model 8) as the best model fit too.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

set.seed(100)
x=arima.sim(model=list(ar=c(0.7,0.2)),n=500)+0.01*(1:500)
out=envcpt(x,models=c(3:6,9:12)) # runs a subset of models (those with AR components)
AIC(out) # returns the AIC for each model
which.min(AIC(out)) # gives trendar2 (model 10) as the best model fit.
out$trendar2 # gives the model fit for the trendar2 model. Notice that the trend is tiny but does
# produce a significantly better fit than the meanar2 model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # best fit is trendar2 (model 10) again.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

## End(Not run)
```

`envcpt` *Assesses whether an environmental time series contains trend, autocorrelation and/or changes*

Description

Evaluates up to 12 different models (see details) and returns the model fits as well as a summary of the likelihood for each model.

Usage

```
envcpt(data,models=c("mean","meancpt","meanar1","meanar2","meanar1cpt","meanar2cpt",
"trend","trendcpt","trendar1","trendar2","trendar1cpt","trendar2cpt"),minseglen=5,...,
verbose=TRUE)
```

Arguments

`data`	A vector or ts object containing the data to fit the models to.
`models`	A vector containing the subset of models to fit, defaults to all available models. This can either be named (as in the default) or numbered (1:12), see details or defaults for number-model pairings.
`minseglen`	Positive integer giving the minimum segment length (no. of observations between changes) for the changepoint models, default is the minimum allowed by theory (for the largest model).
`...`	Additional arguments to pass to the changepoint functions, if none are specified defaults with PELT multiple changepoint algorithm are used. See `cpt.meanvar` for options.

verbose — If TRUE (default), prints to the console an progress bar indicating progression through the fit of the up to 12 models.

Details

This function is used to automatically fit up to 12 different models all with Normal distribution for errors: 1. A constant mean and variance (using `fitdistr`) 2. A piecewise constant mean and variance (using `cpt.meanvar`) 3. A constant mean with AR (1) errors (using `arima`) 4. A constant mean with AR (2) errors (using `arima`) 5. A piecewise constant mean with AR(1) errors (using `cpt.reg` in this package - not exported) 6. A piecewise constant mean with AR(2) errors (using `cpt.reg` in this package - not exported) 7. A linear trend over time (using `lm`) 8. A piecewise linear trend over time (using `cpt.reg` in this package - not exported) 9. A linear trend over time with AR(1) errors (using `lm`) 10. A linear trend over time with AR(2) errors (using `lm`) 11. A piecewise linear trend over time with AR(1) errors (using `cpt.reg` in this package - not exported) 12. A piecewise linear trend over time with AR(2) errors (using `cpt.reg` in this package - not exported) The default values for each function are used, except for the changepoint functions where multiple changes are identified and thus the PELT algorithm is used (see references or changepoint package for details).

Value

`envcpt` outputs a list of 13 elements. The first element is a 2x8 matrix where the first row contains the likelihood for each model fit and the second row contains the number of parameters fit in each model. The 12 columns are for the 12 different models in the order above (headings given). If any element is NA then either there was an error fitting this type of model (typically the AR models and this implies nonstationarity) or the model was not specified in the `models` argument.

Elements 2-13 of the list are the fits for each individual model, these are the direct output from the respective functions so see the individual functions for formats. The first model fit is in element 2 and the twelth model fit is in element 13, but are named for convenience.

Author(s)

Rebecca Killick

References

EnvCpt Algorithm: Beaulieu, C, Killick, R (2018+) Distinguishing trends and shifts from memory in climate data.

PELT Algorithm: Killick R, Fearnhead P, Eckley IA (2012) Optimal detection of changepoints with a linear computational cost, *JASA* **107(500)**, 1590–1598

MBIC: Zhang, N. R. and Siegmund, D. O. (2007) A Modified Bayes Information Criterion with Applications to the Analysis of Comparative Genomic Hybridization Data. *Biometrics* **63**, 22-32.

See Also

`cpt.meanvar`,`plot-methods`,`cpt`, `plot.envcpt`

Examples

```
## Not run:
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,5,1))
out=envcpt(x) # run all models with default values
out[[1]] # first row is twice the negative log-likelihood for each model
         # second row is the number of parameters
AIC(out) # returns AIC for each model.
which.min(AIC(out)) # gives meancpt (model 2) as the best model fit.
out$meancpt # gives the model fit for the meancpt model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # gives meancpt (model 2) as the best model fit too.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

set.seed(10)
x=c(0.01*(1:100),1.5-0.02*((101:250)-101))+rnorm(250,0,0.2)
out=envcpt(x,minseglen=10) # run all models with a minimum of 10 observations between changes
AIC(out) # returns the AIC for each model
which.min(AIC(out)) # gives trendcpt (model 8) as the best model fit.
out$trendcpt # gives the model fit for the trendcpt model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # gives trendcpt (model 8) as the best model fit too.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

set.seed(100)
x=arima.sim(model=list(ar=c(0.7,0.2)),n=500)+0.01*(1:500)
out=envcpt(x,models=c(3:6,9:12)) # runs a subset of models (those with AR components)
AIC(out) # returns the AIC for each model
which.min(AIC(out)) # gives trendar2 (model 10) as the best model fit.
out$trendar2 # gives the model fit for the trendar2 model. Notice that the trend is tiny but does
# produce a significantly better fit than the meanar2 model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # best fit is trendar2 (model 10) again.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

## End(Not run)
```

`plot.envcpt` *Plots optionally either ("fit") the data and fits from each of the 8 models or ("aic") the aic for each of the 8 models.*

Description

Uses the output of the envcpt function and plots optionally ("fit") the original data and the fit from each of the 8 models or ("aic") the aic for each of the 8 models.

Usage

```
## S3 method for class 'envcpt'
plot(x,type=c('fit','bic','aic'),lwd=3,colors=rainbow(12),...,data=NA)
```

Arguments

x	A list produced as output from the envcpt function. In essence a list with the named elements: "mean","meancpt","meanar1","meanar2","meanar1cpt","meanar2cpt","trend","trendcpt where each element is the output from the appropriate function call.
type	character vector. fit returns a plot of the data and fitted models. aic returns a bar chart of the aic values for each of the 8 models
lwd	Line width graphical parameter, see par for further details.
colors	colors for the individual models to be passed to the plotting functions. Note that this must be of length 12 regardless of how many models were evaluated. Color validity is checked using col2rgb.
...	Extra graphical parameters, passed to the original plot and the individual calls to lines or barchar.
data	This argument is only required when x doesn't contain any fits that provide the data which is required for the "fit" type option. Default is NA as most of the time this is likely not required.

Details

If type="fit", the function plots the data at the bottom and stacks the different fits for the 8 models from the envcpt function on top. No scale is given as all data and fits are scaled to be in (0,1). This is designed as an initial visualization tool for the fits only. If type="aic" the function uses the AIC.envcpt function to calculate the AIC values for the envcpt output x. Then barcharts the AIC values in the same order as the type="fit" option. The minimum AIC is the preferred model and this is highlight by a solid block. This is designed as an initial visualization tool for the AIC values only. If type="bic" the function uses the BIC.envcpt function to calculate the BIC values for the envcpt output x. Then barcharts the BIC values in the same order as the type="fit" option. The minimum BIC is the preferred model and this is highlight by a solid block. This is designed as an initial visualization tool for the BIC values only.

For ease of use, if "colours" is specified instead of (or in addition to) colors then this will take precedence.

Value

Returns the printed graphic to the active device.

Author(s)

Rebecca Killick & Claudie Beaulieu.

See Also

envcpt

Examples

```
## Not run:
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,5,1))
out=envcpt(x) # run all models with default values
out[[1]] # first row is twice the negative log-likelihood for each model
         # second row is the number of parameters
AIC(out) # returns AIC for each model.
which.min(AIC(out)) # gives meancpt (model 2) as the best model fit.
out$meancpt # gives the model fit for the meancpt model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # gives meancpt (model 2) as the best model fit too.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

set.seed(10)
x=c(0.01*(1:100),1.5-0.02*((101:250)-101))+rnorm(250,0,0.2)
out=envcpt(x,minseglen=10) # run all models with a minimum of 10 observations between changes
AIC(out) # returns the AIC for each model
which.min(AIC(out)) # gives trendcpt (model 8) as the best model fit.
out$trendcpt # gives the model fit for the trendcpt model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # gives trendcpt (model 8) as the best model fit too.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

set.seed(100)
x=arima.sim(model=list(ar=c(0.7,0.2)),n=500)+0.01*(1:500)
out=envcpt(x,models=c(3:6,9:12)) # runs a subset of models (those with AR components)
AIC(out) # returns the AIC for each model
which.min(AIC(out)) # gives trendar2 (model 10) as the best model fit.
out$trendar2 # gives the model fit for the trendar2 model. Notice that the trend is tiny but does
# produce a significantly better fit than the meanar2 model.
AICweights(out) # gives the AIC weights for each model
BIC(out) # returns the BIC for each model.
which.min(BIC(out)) # best fit is trendar2 (model 10) again.
plot(out,type='fit') # plots the fits
plot(out,type="aic") # plots the aic values
plot(out,type="bic") # plots the bic values

## End(Not run)
```

Index