Package 'cumSeg'

July 17, 2020

Type Package

Title Change Point Detection in Genomic Sequences

Version 1.3

Date 2020-07-18

Author Vito M.R. Muggeo

Maintainer Vito M.R. Muggeo <vito.muggeo@unipa.it>

Description Estimation of number and location of change points in mean-shift (piecewise constant) models. Particularly useful (but not confined) to model genomic sequences of continuous measurements.

Depends lars

License GPL

NeedsCompilation no

Repository CRAN

Date/Publication 2020-07-17 09:10:02 UTC

R topics documented:

cumSeg-package *Change point detection and estimation in genomic sequences*

Description

Estimation of number and location of change points in 'mean-shift' ('piecewise constant' or 'step-function') models. Particularly useful (but not confined) to model genomic sequences of continuous measurements.

Details

Package:	cumSeg
Type:	Package
Version:	1.3
Date:	2020-07-18
License:	GPL
LazyLoad:	yes

Package cumSeg estimates the number and location of change points in 'mean-shift' (also said 'piecewise constant' or 'step-function') models. These models are particularly useful in Biology, Medicine, or Genomics, where it is of interest to know the location of changes in some genomic sequences (e.g. in array comparative genomic hybridization analysis). The algorithm works by first estimating an high number of change points (via the efficient 'segmented' algorithm of Muggeo (2003)) and then by applying the *lars* algorithm of Efron et al. (2004) to select some of them via a generalized BIC criterion. The procedure appears to be (somewhat) robust to some forms of model mis-specifications and, from a computational standpoint, it is substantially independent of the number of change points to be estimated.

Author(s)

Vito M.R. Muggeo <vito.muggeo@unipa.it>

References

Muggeo, V.M.R., Adelfio, G., Efficient change point detection for genomic sequences of continuous measurements, *Bioinformatics* **27**, 161-166.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R. (2004) Least angle regression, *Annals of Statistics* **32**, 407-489.

Muggeo, V.M.R. (2003) Estimating regression models with unknown break-points. *Statistics in Medicine* **22**, 3055-3071.

Examples

```
## Not run:
library(cumSeg)
```

```
data(fibroblast)
#select chromosomes 1.. but the same for chromosomes 3,9,11
z<-na.omit(fibroblast$gm03563[fibroblast$Chromosome==1])
o<-jumpoints(z,k=30,output="3")
plot(z)
plot(o,add=TRUE,y=FALSE,col=4)

## End(Not run)
```

`fibroblast` *Fibroblast Cell Line dataset*

Description

Genomic sequences of 15 fibroblast cell lines.

Usage

```
data(fibroblast)
```

Format

A data frame with 2462 observations on the following 11 variables.

`Chromosome` a numeric vector to identify the chromosome

`Genome.Order` a numeric vector meaning the genome index

`gm05296` cell line GM05296

`gm03563` cell line GM03563

`gm01535` cell line GM01535

`gm07081` cell line GM07081

`gm01750` cell line GM01750

`gm03134` cell line GM03134

`gm13330` cell line GM13330

`gm13031` cell line GM13031

`gm01524` cell line GM01524

Details

Data come from a single experiments on 15 fibroblast cell lines with each array containing over 2000 (mapped) BACs spotted in triplicate. The variable in the dataset is the normalized average of the log base 2 test over reference ratio.

References

Snijders, A. M., Nowak, N., Segraves, R., Blackwood, S., Brown, N., Conroy, J., Hamilton, G., Hindle, A. K., Huey, B., Kimura, K. et al., (2001). Assembly of microarrays for genome-wide measurement of DNA copy number. Nature Genetics 29, 263-264.

Examples

```
## Not run:
data(fibroblast)
#select chromosome 1
z<-na.omit(fibroblast$gm03563[fibroblast$Chromosome==1])
o<-jumpoints(z,k=30,output="3")
plot(z)
plot(o,add=TRUE,y=FALSE,col=4)

## End(Not run)
```

`fit.control` *Auxiliary function for controlling model fitting*

Description

Auxiliary function as user interface for model fitting. Typically only used when calling 'jumpoints'

Usage

```
fit.control(toll = 0.001, it.max = 5, display = FALSE, last = TRUE,
    maxit.glm = 25, h = 1, stop.if.error = FALSE)
```

Arguments

`toll`	positive convergence tolerance.
`it.max`	integer giving the maximal number of iterations.
`display`	logical indicating if the value of the objective function should be printed at each iteration.
`last`	Currently ignored.
`maxit.glm`	Currently ignored.
`h`	Currently ignored.
`stop.if.error`	logical indicating if the algorithm should stop when one or more estimated changepoints do not assume admissible values. Default is `FALSE` which implies automatic changepoint selection.

Value

A list with the arguments as components to be used by 'jumpoints'.

Author(s)

Vito M. R. Muggeo

See Also

`jumpoints`

`jumpoints` *Change point estimation in piecewise constant models*

Description

Estimation of change points and model selection via generalized BIC and other criteria

Usage

```
jumpoints(y, x, k = min(30, round(length(y)/10)), output = "2",
    psi = NULL, round = TRUE, control = fit.control(),
    selection = sel.control(), ...)
```

Arguments

`y`	the observed (genomic) sequence supposed to have a piecewise constant mean function.
`x`	the 'segmented' variable, e.g. the genomic location. If missing simple indices 1,2,... n (length of `y`) are assumed.
`k`	the starting number of changepoints. It should be quite larger than the supposed number of (true) changepoints. This argument is ignored if starting values of the changepoints are specified via `psi`.
`output`	which output should be produced? Possible values are `"1"`, `"2"`, or `"3"`; see Details
`psi`	numeric vector to indicate the starting values for the changepoints. When `psi=NULL` (default), `k` quantiles are assumed.
`round`	logical; should the values of the changepoints be rounded?
`control`	a list returned by `fit.control`.
`selection`	a list returned by `sel.control`.
`...`	additional arguments.

Details

The algorithm works by suitably transforming the observed responses to fit a continuous piecewise linear model. Starting from `k` changepoints, a large number of changepoints is first estimated. This number will be (typically slightly) lower than `k` since some changepoints will be discarded during the iterative steps when taking non admissible values. If `output="1"`, `jumpoints` returns them which typically will be more than the actual ones. If `output="2"` the appropriate number of changepoints is selected via the criterion specified in argument `selection` via `sel.control` (e.g. BIC, MDL, ..). Finally if `output="3"`, the segmented algorithm is run again to try to improve the changepoint estimates returned by the previous step.

Value

A list including several components depending on the value of `output`

If `output="1"` the most relevant components are

`fitted.values`	the fitted values
`n.psi`	the estimated number of changepoints
`est.means`	the estimated means
`psi`	the estimated changepoints

If `output="2"` the most relevant components are

`fitted.values`	the fitted values
`n.psi`	the estimated number of changepoints
`criterion`	the values of the selection criterion
`psi`	the estimated changepoints
`est.means`	the estimated means
`psi0`	the estimated changepoints at ouput 1 (before applying the selection criterion)
`est.means0`	the estimated means at ouput 1 (before applying the selection criterion)

If `output="3"` the most relevant components are those of output 2 but

`psi0`	the estimated changepoints at ouput 1
`psi1`	the estimated changepoints at ouput 2
`psi`	the estimated changepoints at ouput 3 (after applying again the segmented algorithm).

Author(s)

Vito Muggeo

References

Muggeo, V.M.R., Adelfio, G., Efficient change point detection for genomic sequences of continuous measurements, *Bioinformatics* **27**, 161-166.

See Also

`lars`, `sel.control`, `fit.control`.

Examples

```
## Not run:
n<-100
x<-1:n/n

lp<-I(x>.1) -1*I(x>.15)+.585*I(x>.45)-.585*I(x>.6) -I(x>.9)
e<-rnorm(n,0,.154)
y<-lp+e #data

#fit the model without selecting the changepoints
o1<-jumpoints(y,output="1")
plot(o1, typeL="l")
lines(lp, col=2) #true regression function
legend("topright", c("true","fit with output=1"),bty="n", col=c(2,1), lty=1)

#fit model and select the changepoints
o2<-jumpoints(y,output="2")
par(mfrow=c(1,2))
plot(o2, what="c")
plot(o2, typeL="s")
lines(lp, col=3) #true regression function
legend("topright", c("true","fit with output=2"),bty="n", col=c(3,1), lty=1)

## End(Not run)
```

`plot.aCGHsegmented` *Plot method for the class 'aCGHsegmented'*

Description

Plots fitted piecewise constant lines.

Usage

```
## S3 method for class 'aCGHsegmented'
plot(x, add = FALSE, y = TRUE, psi.lines = TRUE, typeL="l",
    what=c("lines","criterion"), ...)
```

Arguments

`x`	object of class `"aCGHsegmented"` returned by `jumpoints`.
`add`	logical; if `TRUE` the fitted piecewise constant lines are added to an existing plot.
`y`	logical; if `TRUE` the observations are also plotted, otherwise only the fitted lines.
`psi.lines`	logical; if `TRUE` vertical lines corresponding to the estimated changepoints are added.

typeL	argument type for the fitted lines. Possible options are typeL="s" to plot the horizontal and vertical lines of the step-function, and typeL="l" to draw the horizontal lines only.
what	If 'lines' the fitted lines are plotted, otherwise the criterion values versus the number of change points, provided the fitted object x has been called with argument output='2' or output='3'.
...	possible additional graphical arguments, such as col, xlab, and so on.

Details

This fuction takes a fitted object returned by jumpoints and plots the resulting fit, namely the estimated step-function and changepoints.

Value

The function simply plots the fit returned by 'jumpoints'.

Author(s)

Vito Muggeo

See Also

jumpoints

---

print.aCGHsegmented *Print method for the aCGHsegmented class*

Description

Printing the most important feautures of a model returned by jumpoints.

Usage

```
## S3 method for class 'aCGHsegmented'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

x	object of class aCGHsegmented
digits	number of digits to be printed
...	arguments passed to other functions

Author(s)

Vito M.R. Muggeo

See Also

jumpoints, plot.aCGHsegmented

`sel.control` *Auxiliary function for controlling model selection*

Description

Auxiliary function as user interface for model selection. Typically only used when calling 'jumpoints'

Usage

```
sel.control(display = FALSE, type = c("bic", "mdl", "rss"), S = 1,
    Cn = "log(log(n))", alg = c("stepwise", "lasso"), edf.psi = TRUE)
```

Arguments

`display`	logical to be passed to the argument `trace` of `lars`
`type`	the criterion to be used to perform model selection.
`S`	if `type="rss"` the optimal model is selected when the residual sum of squares decreases by the threshold `S`.
`Cn`	if `type="bic"` a character string (as a function of `'n'`) to specify to generalized BIC. If `Cn=1` the standard BIC is used.
`alg`	which procedure should be used to perform model selection? The value of `alg` is passed to the argument `'type'` of `lars`.
`edf.psi`	logical indicating if the number of changepoints should be computed in the model df.

Details

This function specifies how to perform model seletion, namely how many change points should be selected.

Value

A list with the arguments as components to be used by `jumpoints` and in turn by `lars`.

Author(s)

Vito Muggeo

See Also

jumpoints, lars

Index