Package 'changepoint'

July 2, 2014

Type Package

Title An R package for changepoint analysis

Version 1.1.5

Date 2014-06-25

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

Description Implements various mainstream and specialised changepoint methods for finding single and multiple changepoints within data. Many popular non-parametric and frequentist methods are included. The cpt.mean, cpt.var, cpt.meanvar functions should be your first point of call.

Depends R(>= 3.0), methods, stats, zoo

License GPL

LazyLoad yes

Author Rebecca Killick [aut, cre],Idris Eckley [aut],Kaylea Haynes [aut],Paul Fearnhead [ctb]

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-06-25 17:32:45

R topics documented:

`changepoint-package` *Contains funcions that run various single and multiple changepoint methods*

Description

Implements various mainstream and specialised changepoint methods for finding single and multiple changepoints within data. Many popular non-parametric and frequentist methods are included. Users should start by looking at the documentation for cpt.mean, cpt.var and cpt.meanvar.

Details

Package:	changepoint
Type:	Package
Version:	1.1.5
Date:	2014-06-25
License:	GPL
LazyLoad:	yes

Author(s)

Rebecca Killick <r.killick@lancs.ac.uk>, Kaylea Haynes <k.haynes1@lancs.ac.uk> with contributions from Idris A. Eckley <i.eckley@lancs.ac.uk>, Paul Fearnhead <p.fearnhead@lancs.ac.uk>.

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

References

Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

cpt.mean,cpt.var,cpt.meanvar

Examples

```
# change in variance
x=c(rnorm(100,0,1),rnorm(100,0,10))
ansvar=cpt.var(x)
plot(ansvar)
print(ansvar)

# change in mean
y=c(rnorm(100,0,1),rnorm(100,5,1))
ansmean=cpt.mean(y)
plot(ansmean,cpt.col='blue')
print(ansmean)

# change in mean and variance
z=c(rnorm(100,0,1),rnorm(100,2,10))
ansmeanvar=cpt.meanvar(z)
plot(ansmeanvar,cpt.width=3)
print(ansmeanvar)
```

---

binseg.mean.cusum — *Multiple Changes in Mean using Binary Segmentation method - Cumulative Sums*

Description

Calculates the optimal positioning and number of changepoints for the cumulative sums test statistic using Binary Segmentation method. Note that this is an approximate method.

Usage

```
binseg.mean.cusum(data, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
Q	Numeric value of the maximum number of changepoints you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the decision as to the optimal number of changepoints.

Details

This function is used to find a multiple changes in mean for data where no assumption about the distribution is made. The value returned is the result of finding the optimal location of up to Q changepoints using the cumulative sums test statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using pen as the penalty function.

Value

A list is returned containing the following items

cps	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
op.cpts	The optimal changepoint locations for the penalty supplied.
pen	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

M. Csorgo, L. Horvath (1997) Limit Theorems in Change-Point Analysis, *Wiley*

E. S. Page (1954) Continuous Inspection Schemes, *Biometrika* **41(1/2)**, 100–115

See Also

binseg.mean.norm,cpt.mean,multiple.mean.cusum,single.mean.cusum,segneigh.mean.cusum

Examples

```
# Example of multiple changes in mean at 50,100,150 in simulated data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,1),rnorm(50,10,1),rnorm(50,3,1))
binseg.mean.cusum(x,Q=5,pen=0.8) # returns optimal number as 3 and the locations as c(50,100,150)
binseg.mean.cusum(x,Q=2,pen=0.8) # returns optimal number as 2 as this is the maximum number of
#changepoints it can find.  If you get the maximum number, you need to increase Q until this is not
#the case.

# Example no change in mean
set.seed(10)
x=rnorm(200,0,1)
binseg.mean.cusum(x,Q=5,pen=0.8) # returns optimal number as 0
```

`binseg.mean.norm` *Multiple Changes in Mean using Binary Segmentation method - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using Binary Segmentation method. Note that this is an approximate method.

Usage

```
binseg.mean.norm(data, Q=5, pen=0)
```

Arguments

`data`	A vector containing the data within which you wish to find changepoints.
`Q`	Numeric value of the maximum number of changepoints you wish to search for, default is 5.
`pen`	Numeric value of the linear penalty function. This value is used in the decision as to the optimal number of changepoints.

Details

This function is used to find a multiple changes in mean for data that is assumed to be normally distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using pen as the penalty function.

Value

A list is returned containing the following items

`cps`	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`pen`	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Change in mean likelihood: Hinkley, D. V. (1970) Inference About the Change-Point in a Sequence of Random Variables, *Biometrika* **57**, 1–17

Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

binseg.var.norm,binseg.meanvar.norm,cpt.mean,PELT.mean.norm,multiple.mean.norm,single.mean.norm,segnei

Examples

```
# Example of multiple changes in mean at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,1),rnorm(50,10,1),rnorm(50,3,1))
binseg.mean.norm(x,Q=5,pen=2*log(200)) # returns optimal number as 3 and the locations as
#c(50,100,150)
binseg.mean.norm(x,Q=2,pen=2*log(200)) # returns optimal number as 2 as this is the maximum number
#of changepoints it can find.  If you get the maximum number, you need to increase Q until this is
#not the case.

# Example no change in mean
set.seed(10)
x=rnorm(200,0,1)
binseg.mean.norm(x,Q=5,pen=2*log(200)) # returns optimal number as 0
```

---

binseg.meanvar.exp *Multiple Changes in Mean and Variance using Binary Segmentation method - Exponential Data*

Description

Calculates the optimal positioning and number of changepoints for Exponential data using Binary Segmentation method. Note that this is an approximate method.

Usage

```
binseg.meanvar.exp(data, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
Q	Numeric value of the maximum number of changepoints you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the decision as to the optimal number of changepoints.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be Exponential distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using pen as the penalty function.

Value

A list is returned containing the following items

cps	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
op.cpts	The optimal changepoint locations for the penalty supplied.
pen	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Change in Exponential model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

binseg.meanvar.norm,binseg.meanvar.gamma,cpt.meanvar,PELT.meanvar.exp,multiple.meanvar.exp,single.mear

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated Exponential data
set.seed(1)
x=c(rexp(50,rate=1),rexp(50,rate=5),rexp(50,rate=1),rexp(50,rate=10))
binseg.meanvar.exp(x,Q=5,pen=2*log(200)) # returns optimal number as 3 and the locations as
#c(50,100,150)
binseg.meanvar.exp(x,Q=2,pen=2*log(200)) # returns optimal number as 2 as this is the maximum number
#of changepoints it can find.  If you get the maximum number, you need to increase Q until this is
#not the case.

# Example no change in mean or variance
set.seed(1)
x=rexp(200,rate=1)
binseg.meanvar.exp(x,pen=2*log(200)) # returns optimal number as 0
```

---

binseg.meanvar.gamma *Multiple Changes in Mean and Variance using Binary Segmentation method - Gamma Data (i.e. change in scale parameter)*

Description

Calculates the optimal positioning and number of changepoints for Gamma data using Binary Segmentation method. Note that this is an approximate method.

Usage

```
binseg.meanvar.gamma(data, shape=1, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
shape	Numerical value of the true shape parameter for the data. Either single value or vector of length nrow(data). If data is a matrix and shape is a single value, the same shape parameter is used for each row.
Q	Numeric value of the maximum number of changepoints you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the decision as to the optimal number of changepoints.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be Gamma distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using pen as the penalty function.

Value

A list is returned containing the following items

cps	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
op.cpts	The optimal changepoint locations for the penalty supplied.
pen	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Change in Gamma scale parameter: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

binseg.meanvar.norm,cpt.meanvar,PELT.meanvar.gamma,multiple.meanvar.gamma,single.meanvar.gamma,segneig

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated Gamma data
set.seed(1)
x=c(rgamma(50,shape=1,rate=1),rgamma(50,shape=1,rate=3),rgamma(50,shape=1,rate=1),rgamma(50,shape=1,
rate=10))
binseg.meanvar.gamma(x,shape=1,Q=5,pen=2*log(200)) # returns optimal number as 3 and the locations
#as c(47,104,150,152)
binseg.meanvar.gamma(x,shape=1,Q=2,pen=2*log(200)) # returns optimal number as 2 as this is the
#maximum number of changepoints it can find.  If you get the maximum number, you need to increase Q
#until this is not the case.

# Example no change in mean or variance
set.seed(1)
x=rgamma(200,shape=1,rate=1)
binseg.meanvar.gamma(x,shape=1,pen=2*log(200)) # returns optimal number as 0
```

---

binseg.meanvar.norm *Multiple Changes in Mean and Variance using Binary Segmentation method - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using Binary Segmentation method. Note that this is an approximate method.

Usage

```
binseg.meanvar.norm(data, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
Q	Numeric value of the maximum number of changepoints you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the decision as to the optimal number of changepoints.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be normally distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using pen as the penalty function.

Value

A list is returned containing the following items

cps	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
op.cpts	The optimal changepoint locations for the penalty supplied.
pen	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Change in mean and variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

binseg.var.norm,binseg.mean.norm,cpt.meanvar,PELT.meanvar.norm,multiple.meanvar.norm,single.meanvar.no

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,3),rnorm(50,10,1),rnorm(50,3,10))
binseg.meanvar.norm(x,Q=5, pen=4*log(200)) # returns optimal number as 4 and the locations as
#c(50,100,150,152)
binseg.meanvar.norm(x,Q=3, pen=4*log(200)) # returns optimal number as 2 as this is the maximum
#number of changepoints it can find.  If you get the maximum number, you need to increase Q until
#this is not the case.

# Example no change in mean or variance
set.seed(1)
x=rnorm(200,0,1)
binseg.meanvar.norm(x,pen=4*log(200)) # returns optimal number as 0
```

---

binseg.meanvar.poisson

Multiple Changes in Mean and Variance using Binary Segmentation method - Poisson Data

---

Description

Calculates the optimal positioning and number of changepoints for Poisson data using Binary Segmentation method. Note that this is an approximate method.

Usage

```
binseg.meanvar.poisson(data, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
Q	Numeric value of the maximum number of changepoints you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the decision as to the optimal number of changepoints.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be Poisson distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using pen as the penalty function.

Value

A list is returned containing the following items

cps	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
op.cpts	The optimal changepoint locations for the penalty supplied.
pen	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Change in Poisson model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

cpt.meanvar,PELT.meanvar.poisson,multiple.meanvar.poisson,single.meanvar.poisson,segneigh.meanvar.poi

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated Poisson data
set.seed(1)
x=c(rpois(50,lambda=1),rpois(50,lambda=5),rpois(50,lambda=1),rpois(50,lambda=10))
binseg.meanvar.poisson(x,Q=5,pen=2*log(200)) # returns optimal number as 3 and the locations as
#c(50,100,150)
binseg.meanvar.poisson(x,Q=2,pen=2*log(200)) # returns optimal number as 2 as this is the maximum
#number of changepoints it can find.  If you get the maximum number, you need to increase Q until
#this is not the case.

# Example no change in mean or variance
set.seed(1)
x=rpois(200,lambda=1)
binseg.meanvar.poisson(x,pen=2*log(200)) # returns optimal number as 0
```

---

binseg.var.css *Multiple Changes in Variance using Binary Segmentation method - Cumulative Sums of Squares*

Description

Calculates the optimal positioning and number of changepoints for the cumulative sums of squares test statistic using Binary Segmentation method. Note that this is an approximate method.

Usage

```
binseg.var.css(data, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
Q	Numeric value of the maximum number of changepoints you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the decision as to the optimal number of changepoints.

Details

This function is used to find a multiple changes in variance for data where no assumption about the distribution is made. The value returned is the result of finding the optimal location of up to Q changepoints using the cumulative sums of squares test statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using pen as the penalty function.

Value

A list is returned containing the following items

cps	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
op.cpts	The optimal changepoint locations for the penalty supplied.
pen	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

C. Inclan, G. C. Tiao (1994) Use of Cumulative Sums of Squares for Retrospective Detection of Changes of Variance, *Journal of the American Statistical Association* **89(427)**, 913–923

R. L. Brown, J. Durbin, J. M. Evans (1975) Techniques for Testing the Constancy of Regression Relationships over Time, *Journal of the Royal Statistical Society B* **32(2)**, 149–192

See Also

binseg.var.norm,cpt.var,multiple.var.css,single.var.css,segneigh.var.css

Examples

```
# Example of multiple changes in variance at 50,100,150 in simulated normal data
set.seed(10)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
binseg.var.css(x,Q=5, pen=1.358) # returns optimal number as 4 and the locations as c(50,52,100,149)
binseg.var.css(x,Q=2, pen=1.358) # returns optimal number as 2 as this is the maximum number of
#changepoints it can find.  If you get the maximum number, you need to increase Q until this is not
#the case.
# 1.358 is the asymptotic value of the penalty for 95% confidence

# Example no change in variance
set.seed(1)
x=rnorm(200,0,1)
binseg.var.css(x,Q=5, pen=1.358) # returns optimal number as 0
```

`binseg.var.norm` *Multiple Changes in Variance using Binary Segmentation method - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using Binary Segmentation method. Note that this is an approximate method.

Usage

```
binseg.var.norm(data, Q=5, pen=0,know.mean=FALSE,mu=NA)
```

Arguments

`data`	A vector containing the data within which you wish to find changepoints.
`Q`	Numeric value of the maximum number of changepoints you wish to search for, default is 5.
`pen`	Numeric value of the linear penalty function. This value is used in the decision as to the optimal number of changepoints.
`know.mean`	Logical, if TRUE then the mean is assumed known and mu is taken as its value. If FALSE, and mu=-1000 (default value) then the mean is estimated via maximum likelihood. If FALSE and the value of mu is supplied, mu is not estimated but is counted as an estimated parameter for decisions.
`mu`	Numerical value of the true mean of the data. Either single value or vector of length nrow(data). If data is a matrix and mu is a single value, the same mean is used for each row.

Details

This function is used to find a multiple changes in variance for data that is assumed to be normally distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using pen as the penalty function.

Value

A list is returned containing the following items

`cps`	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`pen`	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Change in variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

binseg.mean.norm,binseg.meanvar.norm,cpt.var,PELT.var.norm,multiple.var.norm,single.var.norm,segneigh.

Examples

```
# Example of multiple changes in variance at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
binseg.var.norm(x,Q=5, pen=2*log(200)) # returns optimal number as 3 and the locations as
#c(50,99,150)
binseg.var.norm(x,Q=2, pen=2*log(200)) # returns optimal number as 2 as this is the maximum number
#of changepoints it can find.  If you get the maximum number, you need to increase Q until this is
#not the case.

# Example no change in variance
set.seed(10)
x=rnorm(200,0,1)
binseg.var.norm(x,Q=5, pen=2*log(200)) # returns optimal number as 0
```

---

coef-methods *~~ Methods for Function coef ~~*

Description

~~ Methods for function coef ~~

Methods

`signature(object = "cpt")` Retrieves parameter estimates (i.e. param.est slot) from an object of class cpt

`signature(object = "cpt.reg")` Retrieves parameter estimates (i.e. param.est slot) from an object of class cpt.reg

cpt-class *Class "cpt"*

Description

A class for changepoint objects.

Objects from the Class

Objects can be created by calls of the form `new("cpt", ...)`.

`new("cpt", ...)`**:** creates a new object with class cpt

Slots

`data.set`**:** Object of class `"ts"`, a coerced time series of the original data

`cpttype`**:** Object of class `"character"`, the type of changepoint that was identified

`method`**:** Object of class `"character"`, the method that was used to search for changepoints

`test.stat`**:** Object of class `"character"`, the test statistic for the analysis of the data

`pen.type`**:** Object of class `"character"`, the penalty type specified in the analysis

`pen.value`**:** Object of class `"numeric"`, the value of the penalty used in the analysis

`cpts`**:** Object of class `"numeric"`, vector of changepoints identified

`ncpts.max`**:** Object of class `"numeric"`, maximum number of changepoint that can be identified

`param.est`**:** Object of class `"list"`, list where each element is a vector of parameter estimates, if requested

`date`**:** Object of class `"character"`, date and time the changepoint analysis was run

Methods

cpts `signature(object = "cpt")`: retrieves cpts slot

cpttype `signature(object = "cpt")`: retrieves cpttype slot

data.set `signature(object = "cpt")`: retrieves vector version of data.set slot

data.set.ts `signature(object = "cpt")`: retrieves time series version of data.set slot

test.stat `signature(object = "cpt")`: retrieves test.stat slot

ncpts.max `signature(object = "cpt")`: retrieves ncpts.max slot

method `signature(object = "cpt")`: retrieves method slot

param.est `signature(object = "cpt")`: retrieves param.est slot

pen.type `signature(object = "cpt")`: retrieves pen.type slot

pen.value `signature(object = "cpt")`: retrieves pen.value slot

cpts<- `signature(object = "cpt")`: replaces cpts slot

cpttype<- `signature(object = "cpt")`: replaces cpttype slot

data.set<- `signature(object = "cpt")`: replaces data.set slot

test.stat<- `signature(object = "cpt")`: replaces test.stat slot

ncpts.max<- `signature(object = "cpt")`: replaces ncpts.max slot

method<- `signature(object = "cpt")`: replaces method slot

param.est<- `signature(object = "cpt")`: replaces param.est slot

pen.type<- `signature(object = "cpt")`: replaces pen.type slot

pen.value<- `signature(object = "cpt")`: replaces pen.value slot

print `signature(object = "cpt")`: prints details of the cpt object including summary

summary `signature(object = "cpt")`: prints a summary of the cpt object

plot `signature(object = "cpt")`: plots the cpt object with changepoints highlighted

param `signature(object = "cpt")`: calculates the parameter estimates for the cpt object

logLik `signature(object = "cpt")`: returns the overall log-likelihood of the cpt object

Author(s)

Rebecca Killick

See Also

`data.set-methods`,`cpts-methods`,`cpt.reg`,`cpt.mean`,`cpt.var`,`cpt.meanvar`

Examples

```
showClass("cpt") # shows the structure of the cpt class

x=new("cpt") # creates a new object with the cpt class defaults
cpts(x) # retrieves the cpts slot from x
cpts(x)<-c(10,50,100) # replaces the cpts slot from x with c(10,50,100)

# Example of a change in variance at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,0,10))
ans=cpt.var(x)
print(ans) # prints details of the analysis including a summary
summary(ans)
plot(ans) # plots the data with change (vertical line) at 100
logLik(ans) # raw likelihood of the data with changepoints, second value is likelihood + penalty
```

cpt.mean *Identifying Changes in Mean*

Description

Calculates the optimal positioning and (potentially) number of changepoints for data using the user specified method.

Usage

```
cpt.mean(data,penalty="SIC",pen.value=0,method="AMOC",Q=5,test.stat="Normal",class=TRUE,
param.estimates=TRUE)
```

Arguments

data	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
penalty	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
pen.value	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
method	Choice of "AMOC", "PELT", "SegNeigh" or "BinSeg".
Q	The maximum number of changepoints to search for using the "BinSeg" method. The maximum number of segments (number of changepoints + 1) to search for using the "SegNeigh" method.
test.stat	The assumed test statistic / distribution of the data. Currently only "Normal" and "CUSUM" supported.
class	Logical. If TRUE then an object of class cpt is returned.
param.estimates	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find changes in mean for data using the test statistic specfified in the test.stat parameter. The changes are found using the method supplied which can be single changepoint (AMOC) or multiple changepoints using exact (PELT or SegNeigh) or approximate (BinSeg) methods. A changepoint is denoted as the first observation of the new segment / regime.

Value

If `class=TRUE` then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are returned. For `class=FALSE` the structure is as follows.

If data is a vector (single dataset) then a vector/list is returned depending on the value of method. If data is a matrix (multiple datasets) then a list is returned where each element in the list is either a vector or list depending on the value of method.

If method is AMOC then a vector (one dataset) or matrix (multiple datasets) is returned, the columns are:

`cpt`	The most probable location of a changepoint if a change was identified or NA if no changepoint.
`p value`	The p-value of the identified changepoint.

If method is PELT then a vector is returned containing the changepoint locations for the penalty supplied. This always ends with n. If method is SegNeigh then a list is returned with elements:

`cps`	Matrix containing the changepoint positions for 1,...,Q changepoints.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`pen`	Penalty used to find the optimal number of changepoints.
`like`	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

If method is BinSeg then a list is returned with elements:

`cps`	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`pen`	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Change in Normal mean: Hinkley, D. V. (1970) Inference About the Change-Point in a Sequence of Random Variables, *Biometrika* **57**, 1–17

CUSUM Test: M. Csorgo, L. Horvath (1997) Limit Theorems in Change-Point Analysis, *Wiley*

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

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

cpt.var,cpt.meanvar,plot-methods,cpt

Examples

```
# Example of a change in mean at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,10,1))
cpt.mean(x,penalty="SIC",method="AMOC",class=FALSE) # returns 100 to show that the null hypothesis
#was rejected and the change in mean is at 100
ans=cpt.mean(x,penalty="Asymptotic",pen.value=0.01,method="AMOC")
cpts(ans)# returns 100 to show that the null hypothesis was rejected, the change in mean is at 100
#and we are 99% confident of this result
ans=cpt.mean(x,penalty="Manual",pen.value=0.8,method="AMOC",test.stat="CUSUM") # returns 101 as the
#changepoint location

# Example of multiple changes in mean at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,1),rnorm(50,10,1),rnorm(50,3,1))
cpt.mean(x,penalty="Manual",pen.value="2*log(n)",method="BinSeg",Q=5,class=FALSE) # returns optimal
#number of changepoints is 3, locations are 50,100,150.

# Example multiple datasets where the first row has multiple changes in mean and the second row has
#no change in mean
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,1),rnorm(50,10,1),rnorm(50,3,1))
y=rnorm(200,0,1)
z=rbind(x,y)
cpt.mean(z,penalty="Asymptotic",pen.value=0.01,method="SegNeigh",Q=5,class=FALSE) # returns list
#that has two elements, the first has 3 changes in mean and variance at 50,100,150 and the second
#has no changes in variance
ans=cpt.mean(z,penalty="Asymptotic",pen.value=0.01,method="PELT")
cpts(ans[[1]]) # same results as for the SegNeigh method.
cpts(ans[[2]]) # same results as for the SegNeigh method.
```

---

cpt.meanvar *Identifying Changes in Mean and Variance*

Description

Calculates the optimal positioning and (potentially) number of changepoints for data using the user specified method.

Usage

```
cpt.meanvar(data,penalty="SIC",pen.value=0,method="AMOC",Q=5,test.stat="Normal",
class=TRUE,param.estimates=TRUE,shape=1)
```

Arguments

data	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
penalty	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
pen.value	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
method	Choice of "AMOC", "PELT", "SegNeigh" or "BinSeg".
Q	The maximum number of changepoints to search for using the "BinSeg" method. The maximum number of segments (number of changepoints + 1) to search for using the "SegNeigh" method.
test.stat	The assumed test statistic / distribution of the data. Currently only "Normal", "Gamma", "Exponential" and "Poisson" are supported.
class	Logical. If TRUE then an object of class cpt is returned.
param.estimates	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.
shape	Value of the assumed known shape parameter required when test.stat="Gamma".

Details

This function is used to find changes in mean and variance for data using the test statistic specified in the test.stat parameter. The changes are found using the method supplied which can be single changepoint (AMOC) or multiple changepoints using exact (PELT or SegNeigh) or approximate (BinSeg) methods. A changepoint is denoted as the first observation of the new segment / regime.

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot cpts contains the changepoints that are returned. For class=FALSE the structure is as follows.

If data is a vector (single dataset) then a vector/list is returned depending on the value of method. If data is a matrix (multiple datasets) then a list is returned where each element in the list is either a vector or list depending on the value of method.

If method is AMOC then a vector (one dataset) or matrix (multiple datasets) is returned, the columns are:

cpt	The most probable location of a changepoint if a change was identified or NA if no changepoint.

`p value`	The p-value of the identified changepoint.

If method is PELT then a vector is returned containing the changepoint locations for the penalty supplied. This always ends with n. If method is SegNeigh then a list is returned with elements:

`cps`	Matrix containing the changepoint positions for 1,...,Q changepoints.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`pen`	Penalty used to find the optimal number of changepoints.
`like`	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

If method is BinSeg then a list is returned with elements:

`cps`	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`pen`	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Change in Normal mean and variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

Change in Gamma shape parameter: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

Change in Exponential model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

Change in Poisson model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

`cpt.var`,`cpt.mean`,`plot-methods`,`cpt`

Examples

```
# Example of a change in scale parameter (mean and variance) at 100 in simulated gamma data
set.seed(1)
x=c(rgamma(100,shape=1,rate=1),rgamma(100,shape=1,rate=5))
cpt.meanvar(x,penalty="SIC",method="AMOC",test.stat="Gamma",class=FALSE,shape=1) # returns 97 to
#show that the null hypothesis was rejected and the change in scale parameter is at 97
ans=cpt.meanvar(x,penalty="AIC",method="AMOC",test.stat="Gamma",shape=1)
cpts(ans) # returns 97 to show that the null hypothesis was rejected, the change in scale parameter
#is at 97

# Example of multiple changes in mean and variance at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,3),rnorm(50,10,1),rnorm(50,3,10))
cpt.meanvar(x,penalty="Manual",pen.value="4*log(n)",method="BinSeg",Q=5,class=FALSE) # returns
#optimal number of changepoints is 4, locations are 50,100,150,152.

# Example multiple datasets where the first row has multiple changes in mean and variance and the
#second row has no change in mean or variance
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,3),rnorm(50,10,1),rnorm(50,3,10))
y=rnorm(200,0,1)
z=rbind(x,y)
cpt.meanvar(z,penalty="Asymptotic",pen.value=0.01,method="SegNeigh",Q=5,class=FALSE) # returns list
#that has two elements, the first has 3 changes in mean and variance at 50,100,150 and the second
#has no changes in mean or variance
ans=cpt.meanvar(z,penalty="Asymptotic",pen.value=0.01,method="PELT")
cpts(ans[[1]]) # same results as for the SegNeigh method.
cpts(ans[[2]]) # same results as for the SegNeigh method.
```

---

cpt.reg-class *Class "cpt.reg"*

Description

A class for changepoint objects, specifically change in regression.

Objects from the Class

Objects can be created by calls of the form `new("cpt", ...)`.

`new("cpt", ...)`**:** creates a new object with class cpt

Slots

`data.set`**:** Object of class `"numeric"`, the original vector of data

`cpttype`**:** Object of class `"character"`, the type of changepoint that was identified

`method`**:** Object of class `"character"`, the method that was used to search for changepoints, default change in regression

`test.stat`**:** Object of class `"character"`, the test statistic used to analyse the data

`pen.type`**:** Object of class `"character"`, the penalty type specified in the analysis

`pen.value`**:** Object of class `"numeric"`, the value of the penalty used in the analysis

`cpts`**:** Object of class `"numeric"`, vector of changepoints identified

`ncpts.max`**:** Object of class `"numeric"`, maximum number of changepoint that can be identified

`param.est`**:** Object of class `"list"`, list where each element is a vector of parameter estimates, if requested

`date`**:** Object of class `"character"`, date and time the changepoint analysis was run

Methods

cpts `signature(object = "cpt.reg")`: retrieves cpts slot

cpttype `signature(object = "cpt.reg")`: retrieves cpttype slot

data.set `signature(object = "cpt.reg")`: retrieves data.set slot

test.stat `signature(object = "cpt.reg")`: retrieves test.stat slot

ncpts.max `signature(object = "cpt.reg")`: retrieves ncpts.max slot

method `signature(object = "cpt.reg")`: retrieves method slot

param.est `signature(object = "cpt.reg")`: retrieves param.est slot

pen.type `signature(object = "cpt.reg")`: retrieves pen.type slot

pen.value `signature(object = "cpt.reg")`: retrieves pen.value slot

cpts<- `signature(object = "cpt.reg")`: replaces cpts slot

cpttype<- `signature(object = "cpt.reg")`: replaces cpttype slot

data.set<- `signature(object = "cpt.reg")`: replaces data.set slot

test.stat<- `signature(object = "cpt.reg")`: replaces test.stat slot

ncpts.max<- `signature(object = "cpt.reg")`: replaces ncpts.max slot

method<- `signature(object = "cpt.reg")`: replaces method slot

param.est<- `signature(object = "cpt.reg")`: replaces param.est slot

pen.type<- `signature(object = "cpt.reg")`: replaces pen.type slot

pen.value<- `signature(object = "cpt.reg")`: replaces pen.value slot

print `signature(object = "cpt.reg")`: prints details of the cpt object including summary

summary `signature(object = "cpt.reg")`: prints a summary of the cpt object

param `signature(object = "cpt.reg")`: calculates the parameter estimates for the cpt object

Author(s)

Rebecca Killick

See Also

`plot-methods`,`cpts-methods`,`cpt`

Examples

```
showClass("cpt.reg")

x=new("cpt.reg") # creates a new object with the cpt.reg class defaults
data.set(x)  # retrieves the data.set slot from x
data.set(x)<-matrix(1:10,nrow=5,ncol=2) # replaces the data.set slot from x with a matrix
```

---

cpt.var *Identifying Changes in Variance*

Description

Calculates the optimal positioning and (potentially) number of changepoints for data using the user specified method.

Usage

```
cpt.var(data,penalty="SIC",pen.value=0,know.mean=FALSE,mu=NA,method="AMOC",Q=5,
test.stat="Normal",class=TRUE,param.estimates=TRUE)
```

Arguments

data	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
penalty	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
pen.value	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
know.mean	Only required for test.stat="Normal". Logical, if TRUE then the mean is assumed known and mu is taken as its value. If FALSE, and mu=NA (default value) then the mean is estimated via maximum likelihood. If FALSE and the value of mu is supplied, mu is not estimated but is counted as an estimated parameter for decisions.
mu	Only required for test.stat="Normal". Numerical value of the true mean of the data. Either single value or vector of length nrow(data). If data is a matrix and mu is a single value, the same mean is used for each row.
method	Choice of "AMOC", "PELT", "SegNeigh" or "BinSeg".

Q	The maximum number of changepoints to search for using the "BinSeg" method. The maximum number of segments (number of changepoints + 1) to search for using the "SegNeigh" method.
test.stat	The assumed test statistic / distribution of the data. Currently only "Normal" and "CSS" supported.
class	Logical. If TRUE then an object of class cpt is returned.
param.estimates	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find changes in variance for data using the test statistic specified in the test.stat parameter. The changes are found using the method supplied which can be single changepoint (AMOC) or multiple changepoints using exact (PELT or SegNeigh) or approximate (BinSeg) methods. A changepoint is denoted as the first observation of the new segment / regime. Note that for the test.stat="CSS" option the preset penalties are log(.) to allow comparison with test.stat="Normal".

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot cpts contains the changepoints that are returned. For class=FALSE the structure is as follows.

If data is a vector (single dataset) then a vector/list is returned depending on the value of method. If data is a matrix (multiple datasets) then a list is returned where each element in the list is either a vector or list depending on the value of method.

If method is AMOC then a vector (one dataset) or matrix (multiple datasets) is returned, the columns are:

cpt	The most probable location of a changepoint if a change was identified or NA if no changepoint.
p value	The p-value of the identified changepoint.

If method is PELT then a vector is returned containing the changepoint locations for the penalty supplied. This always ends with n. If method is SegNeigh then a list is returned with elements:

cps	Matrix containing the changepoint positions for 1,...,Q changepoints.
op.cpts	The optimal changepoint locations for the penalty supplied.
pen	Penalty used to find the optimal number of changepoints.
like	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

If method is BinSeg then a list is returned with elements:

cps	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
op.cpts	The optimal changepoint locations for the penalty supplied.
pen	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Normal: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

CSS: C. Inclan, G. C. Tiao (1994) Use of Cumulative Sums of Squares for Retrospective Detection of Changes of Variance, *Journal of the American Statistical Association* **89(427)**, 913–923

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

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

cpt.mean,cpt.meanvar,plot-methods,cpt

Examples

```
# Example of a change in variance at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,0,10))
cpt.var(x,penalty="SIC",method="AMOC",class=FALSE) # returns 100 to show that the null hypothesis
#was rejected and the change in variance is at 100
ans=cpt.var(x,penalty="Asymptotic",pen.value=0.01,method="AMOC")
cpts(ans)# returns 100 to show that the null hypothesis was rejected, the change in variance is at
#100 and we are 99% confident of this result

# Example of multiple changes in variance at 50,100,150 in simulated data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
cpt.var(x,penalty="Manual",pen.value="log(2*log(n))",method="BinSeg",test.stat="CSS",Q=5,
class=FALSE) # returns optimal number of changepoints is 4, locations are 50,53,99,150.

# Example multiple datasets where the first row has multiple changes in variance and the second row
#has no change in variance
set.seed(10)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
y=rnorm(200,0,1)
z=rbind(x,y)
cpt.var(z,penalty="Asymptotic",pen.value=0.01,method="SegNeigh",Q=5,class=FALSE) # returns list that
#has two elements, the first has 3 changes in variance at 50,100,149 and the second has no changes
#in variance
ans=cpt.var(z,penalty="Asymptotic",pen.value=0.01,method="PELT")
cpts(ans[[1]]) # same results as for the SegNeigh method.
cpts(ans[[2]]) # same results as for the SegNeigh method.
```

cpts *Generic Function - cpts*

Description

Generic function

Usage

```
cpts(object)
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)

Details

Generic function.

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

cpts-methods

Examples

```
x=new("cpt") # new cpt object
cpts(x) # retrieves the cpts slot from x
```

cpts-methods *~~ Methods for Function cpts ~~*

Description

~~ Methods for function `cpts` ~~

Methods

`signature(object = "cpt")` Retrieves cpts slot from an object of class cpt, from version 1.0 this no longer prints the length of the dataset.

`signature(object = "cpt.reg")` Retrieves cpts slot from an object of class cpt.reg, from version 1.0 this no longer prints the length of the dataset.

cpts.ts *Generic Function - cpts.ts*

Description

Generic function

Usage

```
cpts.ts(object)
```

Arguments

`object`	Depending on the class of `object` depends on the method used (and if one exists)

Details

Generic function.

Value

Depends on the class of `object`, see individual methods

Author(s)

Rebecca Killick

See Also

`cpts.ts-methods`

Examples

```
x=new("cpt") # new cpt object
cpts.ts(x) # retrieves the cpts slot from x but unlike cpts, displays the "date" index rather than
#the position within the vector
```

`cpts.ts-methods` *~~ Methods for Function cpts.ts ~~*

Description

~~ Methods for function `cpts.ts` ~~

Methods

`signature(object = "cpt")` Retrieves cpts slot from an object of class cpt, from version 1.0 this no longer prints the length of the dataset. Contrary to the `cpts` function, cpts.ts displays the changepoints using the "date" index according to the start and frequency of the time series data.set(object)

`signature(object = "cpt.reg")` Retrieves cpts slot from an object of class cpt.reg, from version 1.0 this no longer prints the length of the dataset. Contrary to the `cpts` function, cpts.ts displays the changepoints using the "date" index according to the start and frequency of the time series data.set(object)

`cpts<-` *Generic Function - cpts<-*

Description

Generic function

Usage

```
cpts(object)<-value
```

Arguments

`object`	Depending on the class of `object` depends on the method used (and if one exists)
`value`	Replacement value

Details

Generic function.

Value

Depends on the class of `object`, see individual methods

Author(s)

Rebecca Killick

See Also

`cpts<--methods`

Examples

```
x=new("cpt") # new cpt object
cpts(x)<-10 # replaces the vector of changepoint in object x with 10
```

---

cpts<--methods *~~ Methods for Function cpts<- ~~*

Description

~~ Methods for function `cpts<-` ~~

Methods

`signature(x = "cpt")` Assigns the value following <- to the cpts slot in x, from version 1.0 this no longer requires the last entry to be the length of the dataset.

`signature(x = "cpt.reg")` Assigns the value following <- to the cpts slot in x, from version 1.0 this no longer requires the last entry to be the length of the dataset.

---

cpttype *Generic Function - cpttype*

Description

Generic function

Usage

```
cpttype(object)
```

Arguments

`object`	Depending on the class of `object` depends on the method used (and if one exists)

Details

Generic Function

Value

Depends on the class of `object`, see individual methods

Author(s)

Rebecca Killick

See Also

`cpttype-methods`

Examples

```
x=new("cpt") # new cpt object
cpttype(x) # retrieves the cpttype slot from x
```

`cpttype-methods` *~~ Methods for Function cpttype ~~*

Description

~~ Methods for function `cpttype` ~~

Methods

`signature(object = "cpt")` Retrieves cpttype slot from an object of class cpt

`signature(object = "cpt.reg")` Retrieves cpttype slot from an object of class cpt.reg

`cpttype<-` *Generic Function - cpttype<-*

Description

Generic function

Usage

```
cpttype(object)<-value
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)
value	Replacement value

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

cpttype<--methods

Examples

```
x=new("cpt") # new cpt object
cpttype(x)<-"mean" # replaces the existing cpttype in object x with "mean"
```

---

cpttype<--methods ~~ *Methods for Function cpttype<-* ~~

Description

~~ Methods for function cpttype<- ~~

Methods

signature(x = "cpt") Assigns the value following <- to the cpttype slot in x

signature(x = "cpt.reg") Assigns the value following <- to the cpttype slot in x

data.set *Generic Function - data.set*

Description

Generic function

Usage

```
data.set(object)
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

data.set-methods

Examples

```
x=new("cpt") # new cpt object
data.set(x) # retrieves the data.set slot from x
```

data.set-methods *~~ Methods for Function data.set ~~*

Description

~~ Methods for function data.set ~~

Methods

signature(object = "cpt") Retrieves the data.set slot from objects with class cpt

signature(object = "cpt.reg") Retrieves the data.set slot from objects with class cpt.reg

`data.set.ts` *Generic Function - data.set.ts*

Description

Generic function

Usage

```
data.set.ts(object)
```

Arguments

`object`	Depending on the class of `object` depends on the method used (and if one exists)

Details

Generic Function

Value

Depends on the class of `object`, see individual methods

Author(s)

Rebecca Killick

See Also

`data.set.ts-methods`

Examples

```
x=new("cpt") # new cpt object
data.set.ts(x) # retrieves the data.set slot from x.  This is a ts object.
```

data.set.ts-methods *~~ Methods for Function data.set.ts ~~*

Description

~~ Methods for function data.set.ts ~~

Methods

signature(object = "cpt") Retrieves the data.set slot from objects with class cpt. This returns a ts class object.

signature(object = "cpt.reg") Retrieves the data.set slot from objects with class cpt.reg. This returns a ts class object.

data.set<- *Generic Function - data.set<-*

Description

Generic function

Usage

```
data.set(object)<-value
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)
value	Replacement value

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

data.set<--methods

Examples

```
x=new("cpt") # new cpt object
data.set(x)<-c(1,2,3,4,5) # replaces the existing data.set slot in x with c(1,2,3,4,5)
```

---

data.set<--methods *~~ Methods for Function data.set<- ~~*

Description

~~ Methods for function `data.set<-` ~~

Methods

`signature(x = "cpt")` Assigns the value following <- to the data.set slot in x

`signature(x = "cpt.reg")` Assigns the value following <- to the data.set slot in x

---

decision *Likelihood Ratio Decision Function*

Description

Uses the function parameters to decide if a proposed changepoint is a true changepoint or due to random variability. Test is conducted using the user specified penalty.

Usage

```
decision(tau,null,alt=NA,penalty="SIC",n=0,diffparam=1,pen.value=0)
```

Arguments

`tau`	A numeric value or vector specifying the proposed changepoint location(s).
`null`	The value of the null likelihood to be used in the liklihood ratio calculation. If tau is a vector, so is null. If the test statistic is already known, replace the null argument with the test statistic.
`alt`	The value of the alternative likelihood (at tau) to be used in the likelihood ratio calculation. If tau is a vector, so is alt. If the test statistic is already known, then it is used in replacement of the null argument and the alternative should not be specified (default NA to account for this)
`penalty`	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
`n`	The length of the original data, required to give sensible "no changepoint".

`diffparam` The difference in the number of parameters in the null and alternative hypotheses, required for the SIC, BIC, AIC, Hanna-Quinn and possibly Manual penalties.

`pen.value` The value of the penalty when using the Manual or Asymptotic option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.

Details

This function is used to test whether tau is a true changepoint or not. This test uses the likelihood ratio as the test statistic and performs the test where the null hypothesis is no change point and the alternative hypothesis is a single changepoint at tau. The test is (null-alt)>=penalty, if TRUE then the changepoint is deemed a true changepoint, if FALSE then n (length of data) is returned.

If the test statistic is already known then it replaces the null value and the alternative is not required (default NA). In this case the test is null>=penalty, if TRUE then the changepoint is deemed a true changepoint, if FALSE then n (length of data) is returned.

In reality this function should not be used unless you are performing a changepoint test using output from other functions. This function is used in the "see also" functions that perform various changepoint tests, ideally these should be used.

Value

A list is returned with two elements, cpt and pen.

`cpt` If tau is a single value then a single value is returned: Either the value of the true changepoint location or n (length of data) if no changepoint is found.

If tau is a vector of length m then a vector of length m is returned:Each element is either the value of the true changepoint location or n (length of data) if no changepoint is found. The first element is for the first value of tau and the final element is for the final value of tau.

`pen` The numeric value of the penalty used for the test(s).

Author(s)

Rebecca Killick

References

SIC/BIC: Schwarz, G. (1978) Estimating the Dimension of a Model, *The Annals of Statistics* **6(2)**, 461–464

AIC: Akaike, H. (1974) A new look at the statistical model identification, *Automatic Control, IEEE Transactions on* **19(6)**, 716–723

Hannan-Quinn: Hannan, E. J. and B. G. Quinn (1979) The Determination of the Order of an Autoregression, *Journal of the Royal Statistical Society, B* **41**, 190–195

See Also

cpt.mean,cpt.var,cpt.meanvar,single.mean.norm,single.var.norm,single.meanvar.norm

Examples

```
# Example of finding a change
out=c(100,765.1905,435.6529) # tau, null, alt
decision(out[1],out[2],out[3],penalty="SIC",n=200,diffparam=1) # returns 100 as a true changepoint

# Example of no change found
out=c(53,-22.47768,-24.39894) # tau, null, alt
decision(out[1],out[2],out[3],penalty="Manual",n=200,diffparam=1,pen.value="2*log(n)")
```

---

distribution *Generic Function - distribution*

Description

Generic function

Usage

```
distribution(object)
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

distribution-methods

Examples

```
x=new("cpt") # new cpt object
distribution(x) # retrieves the distribution slot from x
```

distribution-methods ~~ *Methods for Function distribution* ~~

Description

~~ Methods for function `distribution` ~~

Methods

`signature(object = "cpt")` Retrieves distribution slot from an object of class cpt

`signature(object = "cpt.reg")` Retrieves distribution slot from an object of class cpt.reg

distribution<- *Generic Function - distribution<-*

Description

Generic function

Usage

```
distribution(object)<-value
```

Arguments

`object`	Depending on the class of `object` depends on the method used (and if one exists)
`value`	Replacement value

Details

Generic Function

Value

Depends on the class of `object`, see individual methods

Author(s)

Rebecca Killick

See Also

`distribution<--methods`

Examples

```
x=new("cpt") # new cpt object
distribution(x)<-"normal" # replaces the current distribution slot of x with "normal"
```

`distribution<--methods` *~~ Methods for Function distribution<- ~~*

Description

~~ Methods for function `distribution<-` ~~

Methods

`signature(x = "cpt")` Assigns the value following <- to the distribution slot in x

`signature(x = "cpt.reg")` Assigns the value following <- to the distribution slot in x

---

`ftse100` *FTSE 100 Daily Returns: 2nd April 1984 – 13th September 2012*

Description

This dataset gives the daily returns (c_t+1/c_t -1) of the UK FTSE 100 index from 2nd April 1984 until the 13th September 2012.

Usage

```
ftse100
```

Format

A matrix of dimension 7187 x 2 where the first column is the Date and the second column is the Daily Return.

Source

Yahoo! Finance

HC1 *G+C Content in Human Chromosome 1*

Description

This dataset gives the G+C content in 3kb windows along the Human Chromosome from 10Mb to 33Mb (no missing data).

Usage

```
HC1
```

Format

A vector of length 23553.

Source

http://www.ncbi.nlm.nih.gov/mapview/map_search.cgi?taxid=9606&build=previous

Lai2005fig3 *Normalized glioblastoma profile for chromosome 13*

Description

This dataset is taken from Lai W, Johnson MJ, Kucherlapati R, Park PJ, Bioinformatics , 2005. The paper states that the original source of the data is from Bredel et al. (2005). The data is Chromosome 13 in GBM31.

Usage

```
Lai2005fig3
```

Format

A matrix of dimensions 797 x 5. The columns are Spot, CH, POS.start, POS.end, GBM31.

Source

http://compbio.med.harvard.edu/Supplements/Bioinformatics05b/Profiles/Chrom_13_GBM31.xls

Lai2005fig4 — *Normalized glioblastoma profile for an excerpt of chromosome 7, the EGFR locus.*

Description

This dataset is taken from Lai W, Johnson MJ, Kucherlapati R, Park PJ, Bioinformatics , 2005. The paper states that the original source of the data is from Bredel et al. (2005). The data is an excerpt of chromosome 7 in GBM29 from 40 to 65 Mb.

Usage

```
Lai2005fig4
```

Format

A matrix of dimensions 193 x 5. The columns are Spot, CH, POS.start, POS.end, GBM31.

Source

http://compbio.med.harvard.edu/Supplements/Bioinformatics05b/Profiles/Chrom_7_from40_to65Mb_GBM29.xls

likelihood — *Generic Function - likelihood*

Description

Generic function to calculate the likelihood

Usage

```
likelihood(object)
```

Arguments

object	Depending on the class of object depends on the method used to calculate the likelihood (and if one exists)

Details

Generic Function to calculate the likelihood.

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

likelihood-methods, cpt.mean,cpt.var,cpt.meanvar

Examples

```
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
out=cpt.var(x,penalty="Manual",pen.value="2*log(n)",method="BinSeg",Q=5)
likelihood(out) # returns the raw scaled negative likelihood (783.9144) and the scaled negative
#likelihood + penalty (805.1076)
```

---

likelihood-methods *~~ Methods for Function likelihood ~~*

Description

~~ Methods for function likelihood ~~

Methods

signature(object = "cpt") Returns the likelihood of the data with the fitted changepoints, two values are returned, the raw scaled negative likelihood and the scaled negative likelihood + penalty. Only valid for cpttype="mean","variance" or "mean and variance".

---

logLik-methods *~~ Methods for Function logLik ~~*

Description

~~ Methods for function logLik ~~

Methods

signature(object = "cpt") Returns the log-likelihood of the data with the fitted changepoints, two values are returned, the raw scaled negative log-likelihood and the scaled negative log-likelihood + penalty. Only valid for cpttype="mean","variance" or "mean and variance".

Examples

```
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
out=cpt.var(x,penalty="Manual",pen.value="2*log(n)",method="BinSeg",Q=5)
likelihood(out) # returns the raw scaled negative likelihood (783.9144) and the scaled negative
#likelihood + penalty (805.1076)
```

method *Generic Function - method*

Description

Generic function

Usage

```
method(object)
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

method-methods

Examples

```
x=new("cpt") # new cpt object
method(x) # retrieves the method slot from x
```

method-methods *~~ Methods for Function method ~~*

Description

~~ Methods for function method ~~

Methods

signature(object = "cpt") Retrieves method slot from an object of class cpt

signature(object = "cpt.reg") Retrieves method slot from an object of class cpt.reg

method<- *Generic Function - method<-*

Description

Generic function

Usage

```
method(object)<-value
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)
value	Replacement value

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

method<--methods

Examples

```
x=new("cpt") # new cpt object
method(x)<-"mean" # replaces the existing method slot in x with "mean"
```

method<--methods *~~ Methods for Function method<- ~~*

Description

~~ Methods for function `method<-` ~~

Methods

`signature(x = "cpt")` Assigns the value following <- to the method slot in x

`signature(x = "cpt.reg")` Assigns the value following <- to the method slot in x

multiple.mean.cusum *Multiple Changes in Mean - Cumulative Sums*

Description

Calculates the optimal positioning and number of changepoints for the cumulative sums test statistic using the user specified method.

Usage

```
multiple.mean.cusum(data,mul.method="BinSeg",penalty="Asymptotic",pen.value=0.05,Q=5,
class=TRUE,param.estimates=TRUE)
```

Arguments

`data`	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`mul.method`	Choice of "SegNeigh" or "BinSeg".
`penalty`	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
`pen.value`	The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
`Q`	The maximum number of changepoints to search for using the "BinSeg" method. The maximum number of segments (number of changepoints + 1) to search for using the "SegNeigh" method.
`class`	Logical. If TRUE then an object of class `cpt` is returned.
`param.estimates`	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find multiple changes in mean for data where no assumption about the distribution is made. The changes are found using the method supplied which can be exact (SegNeigh) or approximate (BinSeg). Note that the programmed penalty values are not designed to be used with the CUSUM method, it is advised to use Asymptotic or Manual penalties.

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are solely returned if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a vector/list is returned depending on the value of mul.method. If data is a matrix (multiple datasets) then a list is returned where each element in the list is either a vector or list depending on the value of mul.method.

If mul.method is SegNeigh then a list is returned with elements:

`cps`	Matrix containing the changepoint positions for 1,...,Q changepoints.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`pen`	Penalty used to find the optimal number of changepoints.

If mul.method is BinSeg then a list is returned with elements:

`cps`	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`pen`	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

M. Csorgo, L. Horvath (1997) Limit Theorems in Change-Point Analysis, *Wiley*

E. S. Page (1954) Continuous Inspection Schemes, *Biometrika* **41(1/2)**, 100–115

See Also

`multiple.var.css`,`cpt.mean`,`binseg.mean.cusum`,`single.mean.cusum`,`segneigh.mean.cusum`,`cpt`

Examples

```
# Example of multiple changes in mean at 50,100,150 in simulated data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,1),rnorm(50,10,1),rnorm(50,3,1))
multiple.mean.cusum(x,mul.method="BinSeg",penalty="Manual",pen.value=0.8,Q=5,class=FALSE) # returns
#optimal number of changepoints is 3, locations are 50,100,150.

# Example multiple datasets where the first row has multiple changes in mean and the second row has
#no change in mean
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,1),rnorm(50,10,1),rnorm(50,3,1))
y=rnorm(200,0,1)
z=rbind(x,y)
multiple.mean.cusum(z,mul.method="SegNeigh",penalty="Manual",pen.value=1,Q=5,class=FALSE) # returns
#list that has two elements, the first has 3 changes in mean at 50,101,150 and the second has no
#changes in mean
ans=multiple.mean.cusum(z,mul.method="BinSeg",penalty="Manual",pen.value=0.8)
cpts(ans[[1]]) # same results as for the SegNeigh method.
cpts(ans[[2]]) # same results as for the SegNeigh method.
```

`multiple.mean.norm` *Multiple Changes in Mean - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using the user specified method.

Usage

```
multiple.mean.norm(data,mul.method="PELT",penalty="SIC",pen.value=0,Q=5,class=TRUE,
param.estimates=TRUE)
```

Arguments

`data`	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`mul.method`	Choice of "PELT", "SegNeigh" or "BinSeg".
`penalty`	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
`pen.value`	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length

of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.

`Q` The maximum number of changepoints to search for using the "BinSeg" method. The maximum number of segments (number of changepoints + 1) to search for using the "SegNeigh" method.

`class` Logical. If TRUE then an object of class `cpt` is returned.

`param.estimates`
Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find multiple changes in mean for data that is assumed to be normally distributed. The changes are found using the method supplied which can be exact (PELT or SegNeigh) or approximate (BinSeg).

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are solely returned if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a vector/list is returned depending on the value of mul.method. If data is a matrix (multiple datasets) then a list is returned where each element in the list is either a vector or list depending on the value of mul.method.

If mul.method is PELT then a vector is returned:

`cpt` Vector containing the changepoint locations for the penalty supplied. This always ends with n.

If mul.method is SegNeigh then a list is returned with elements:

`cps` Matrix containing the changepoint positions for 1,...,Q changepoints.

`op.cpts` The optimal changepoint locations for the penalty supplied.

`like` Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

If mul.method is BinSeg then a list is returned with elements:

`cps` 2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.

`op.cpts` The optimal changepoint locations for the penalty supplied.

`pen` Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Change in Normal mean: Hinkley, D. V. (1970) Inference About the Change-Point in a Sequence of Random Variables, *Biometrika* **57**, 1–17

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

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

multiple.var.norm,multiple.meanvar.norm,cpt.mean,PELT.mean.norm,binseg.mean.norm,single.mean.norm,segn

Examples

```
# Example of multiple changes in mean at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,1),rnorm(50,10,1),rnorm(50,3,1))
multiple.mean.norm(x,mul.method="BinSeg",penalty="Manual",pen.value="2*log(n)",Q=5,class=FALSE)
# returns optimal number of changepoints is 3, locations are 50,100,150.

# Example multiple datasets where the first row has multiple changes in mean and the second row has
#no change in mean
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,1),rnorm(50,10,1),rnorm(50,3,1))
y=rnorm(200,0,1)
z=rbind(x,y)
multiple.mean.norm(z,mul.method="SegNeigh",penalty="Asymptotic",pen.value=0.01,Q=5,class=FALSE)
# returns list that has two elements, the first has 3 changes in mean at 50,100,150 and the second
#has no changes in mean
ans=multiple.mean.norm(z,mul.method="PELT",penalty="Asymptotic",pen.value=0.01)
cpts(ans[[1]]) # same results as for the SegNeigh method.
cpts(ans[[2]]) # same results as for the SegNeigh method.
```

---

multiple.meanvar.exp *Multiple Changes in Mean and Variance - Exponential Data*

Description

Calculates the optimal positioning and number of changepoints for Exponential data using the user specified method.

Usage

```
multiple.meanvar.exp(data,mul.method="PELT",penalty="SIC",pen.value=0,Q=5,class=TRUE,
param.estimates=TRUE)
```

Arguments

data	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
mul.method	Choice of "PELT", "SegNeigh" or "BinSeg".
penalty	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
pen.value	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
Q	The maximum number of changepoints to search for using the "BinSeg" method. The maximum number of segments (number of changepoints + 1) to search for using the "SegNeigh" method.
class	Logical. If TRUE then an object of class cpt is returned.
param.estimates	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find multiple changes in mean and variance for data that is assumed to be Exponential distributed. The changes are found using the method supplied which can be exact (PELT or SegNeigh) or approximate (BinSeg).

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot cpts contains the changepoints that are solely returned if class=FALSE. The structure of cpts is as follows.

If data is a vector (single dataset) then a vector/list is returned depending on the value of mul.method. If data is a matrix (multiple datasets) then a list is returned where each element in the list is either a vector or list depending on the value of mul.method.

If mul.method is PELT then a vector is returned:

cpt	Vector containing the changepoint locations for the penalty supplied. This always ends with n.

If mul.method is SegNeigh then a list is returned with elements:

cps	Matrix containing the changepoint positions for 1,...,Q changepoints.
op.cpts	The optimal changepoint locations for the penalty supplied.
like	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

If mul.method is BinSeg then a list is returned with elements:

cps	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
op.cpts	The optimal changepoint locations for the penalty supplied.
pen	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Change in Exponential model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

multiple.meanvar.norm,multiple.meanvar.gamma,cpt.meanvar,PELT.meanvar.exp,binseg.meanvar.exp,single.me

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated Exponential data
set.seed(1)
x=c(rexp(50,rate=1),rexp(50,rate=5),rexp(50,rate=1),rexp(50,rate=10))
multiple.meanvar.exp(x,mul.method="BinSeg",penalty="Manual",pen.value="2*log(n)",Q=5,class=FALSE)
# returns optimal number of changepoints is 3, locations are 50,100,150.

# Example multiple datasets where the first row has multiple changes in mean and variance and the
#second row has no change in mean or variance
set.seed(1)
x=c(rexp(50,rate=1),rexp(50,rate=5),rexp(50,rate=1),rexp(50,rate=10))
y=rexp(200,rate=1)
z=rbind(x,y)
multiple.meanvar.exp(z,mul.method="SegNeigh",penalty="Manual",pen.value=2*log(200),Q=5,class=FALSE)
# returns list that has two elements, the first has 3 changes in mean and variance at 50,100,150 and
#the second has no changes in mean or variance
ans=multiple.meanvar.exp(z,mul.method="PELT",penalty="Manual",pen.value=2*log(200))
cpts(ans[[1]]) # same results as for the SegNeigh method.
cpts(ans[[2]]) # same results as for the SegNeigh method.
```

`multiple.meanvar.gamma` *Multiple Changes in Mean and Variance - Gamma Data (i.e. change in scale parameter)*

Description

Calculates the optimal positioning and number of changepoints for Gamma data using the user specified method.

Usage

```
multiple.meanvar.gamma(data,shape=1,mul.method="PELT",penalty="SIC",pen.value=0,Q=5,
class=TRUE,param.estimates=TRUE)
```

Arguments

`data`	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`shape`	Numerical value of the true shape parameter for the data. Either single value or vector of length nrow(data). If data is a matrix and shape is a single value, the same shape parameter is used for each row.
`mul.method`	Choice of "PELT", "SegNeigh" or "BinSeg".
`penalty`	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
`pen.value`	The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
`Q`	The maximum number of changepoints to search for using the "BinSeg" method. The maximum number of segments (number of changepoints + 1) to search for using the "SegNeigh" method.
`class`	Logical. If TRUE then an object of class `cpt` is returned.
`param.estimates`	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find multiple changes in mean and variance for data that is assumed to be Gamma distributed. The changes are found using the method supplied which can be exact (PELT or SegNeigh) or approximate (BinSeg).

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are solely returned if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a vector/list is returned depending on the value of mul.method. If data is a matrix (multiple datasets) then a list is returned where each element in the list is either a vector or list depending on the value of mul.method.

If mul.method is PELT then a vector is returned:

`cpt`	Vector containing the changepoint locations for the penalty supplied. This always ends with n.

If mul.method is SegNeigh then a list is returned with elements:

`cps`	Matrix containing the changepoint positions for 1,...,Q changepoints.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`like`	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

If mul.method is BinSeg then a list is returned with elements:

`cps`	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`pen`	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Change in Gamma shape parameter: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

`multiple.meanvar.norm`,`cpt.meanvar`,`PELT.meanvar.gamma`,`binseg.meanvar.gamma`,`single.meanvar.gamma`,`segneig`

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated Gamma data
set.seed(1)
x=c(rgamma(50,shape=1,rate=1),rgamma(50,shape=1,rate=3),rgamma(50,shape=1,rate=1),rgamma(50,shape=1,
rate=10))
multiple.meanvar.gamma(x,shape=1,mul.method="BinSeg",penalty="Manual",pen.value="2*log(n)",Q=5,
class=FALSE) # returns optimal number of changepoints is 3, locations are 47,104,150.

# Example multiple datasets where the first row has multiple changes in mean and variance and the
#second row has no change in mean or variance
set.seed(1)
x=c(rgamma(50,shape=1,rate=1),rgamma(50,shape=1,rate=3),rgamma(50,shape=1,rate=1),rgamma(50,shape=1,
rate=10))
y=rgamma(200,shape=1,rate=1)
z=rbind(x,y)
multiple.meanvar.gamma(z,shape=1,mul.method="SegNeigh",penalty="SIC",Q=5,class=FALSE) # returns list
#that has two elements, the first has 4 changes in mean and variance at 47,102,151,172 and the
#second has no changes in mean or variance
ans=multiple.meanvar.gamma(z,shape=1,mul.method="PELT",penalty="SIC")
cpts(ans[[1]]) # same results as for the SegNeigh method.
cpts(ans[[2]]) # same results as for the SegNeigh method.
```

`multiple.meanvar.norm` *Multiple Changes in Mean and Variance - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using the user specified method.

Usage

```
multiple.meanvar.norm(data,mul.method="PELT",penalty="SIC",pen.value=0,Q=5,class=TRUE,
param.estimates=TRUE)
```

Arguments

`data`	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`mul.method`	Choice of "PELT", "SegNeigh" or "BinSeg".
`penalty`	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.

`pen.value` The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.

`Q` The maximum number of changepoints to search for using the "BinSeg" method. The maximum number of segments (number of changepoints + 1) to search for using the "SegNeigh" method.

`class` Logical. If TRUE then an object of class `cpt` is returned.

`param.estimates` Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find multiple changes in mean and variance for data that is assumed to be normally distributed. The changes are found using the method supplied which can be exact (PELT or SegNeigh) or approximate (BinSeg).

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are solely returned if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a vector/list is returned depending on the value of mul.method. If data is a matrix (multiple datasets) then a list is returned where each element in the list is either a vector or list depending on the value of mul.method.

If mul.method is PELT then a vector is returned:

`cpt` Vector containing the changepoint locations for the penalty supplied. This always ends with n.

If mul.method is SegNeigh then a list is returned with elements:

`cps` Matrix containing the changepoint positions for 1,...,Q changepoints.

`op.cpts` The optimal changepoint locations for the penalty supplied.

`like` Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

If mul.method is BinSeg then a list is returned with elements:

`cps` 2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.

`op.cpts` The optimal changepoint locations for the penalty supplied.

`pen` Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Change in Normal mean and variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

multiple.var.norm,multiple.mean.norm,cpt.meanvar,PELT.meanvar.norm,binseg.meanvar.norm,single.meanvar.

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,3),rnorm(50,10,1),rnorm(50,3,10))
multiple.meanvar.norm(x,mul.method="BinSeg",penalty="Manual",pen.value="4*log(n)",Q=5,class=FALSE)
# returns optimal number of changepoints is 4, locations are 50,100,150,152.

# Example multiple datasets where the first row has multiple changes in mean and variance and the
#second row has no change in mean or variance
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,3),rnorm(50,10,1),rnorm(50,3,10))
y=rnorm(200,0,1)
z=rbind(x,y)
multiple.meanvar.norm(z,mul.method="SegNeigh",penalty="Asymptotic",pen.value=0.01,Q=5,class=FALSE)
# returns list that has two elements, the first has 3 changes in mean and variance at 50,100,150 and
#the second has no changes in mean or variance
ans=multiple.meanvar.norm(z,mul.method="PELT",penalty="Asymptotic",pen.value=0.01)
cpts(ans[[1]]) # same results as for the SegNeigh method.
cpts(ans[[2]]) # same results as for the SegNeigh method.
```

---

multiple.meanvar.poisson

Multiple Changes in Mean and Variance - Poisson Data

Description

Calculates the optimal positioning and number of changepoints for Poisson data using the user specified method.

Usage

```
multiple.meanvar.poisson(data,mul.method="PELT",penalty="SIC",pen.value=0,Q=5,class=TRUE,
param.estimates=TRUE)
```

Arguments

data	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
mul.method	Choice of "PELT", "SegNeigh" or "BinSeg".
penalty	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
pen.value	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
Q	The maximum number of changepoints to search for using the "BinSeg" method. The maximum number of segments (number of changepoints + 1) to search for using the "SegNeigh" method.
class	Logical. If TRUE then an object of class cpt is returned.
param.estimates	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find multiple changes in mean and variance for data that is assumed to be Poisson distributed. The changes are found using the method supplied which can be exact (PELT or SegNeigh) or approximate (BinSeg).

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot cpts contains the changepoints that are solely returned if class=FALSE. The structure of cpts is as follows.

If data is a vector (single dataset) then a vector/list is returned depending on the value of mul.method. If data is a matrix (multiple datasets) then a list is returned where each element in the list is either a vector or list depending on the value of mul.method.

If mul.method is PELT then a vector is returned:

cpt	Vector containing the changepoint locations for the penalty supplied. This always ends with n.

If mul.method is SegNeigh then a list is returned with elements:

cps	Matrix containing the changepoint positions for 1,...,Q changepoints.
op.cpts	The optimal changepoint locations for the penalty supplied.
like	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

If mul.method is BinSeg then a list is returned with elements:

cps	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
op.cpts	The optimal changepoint locations for the penalty supplied.
pen	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Change in Poisson model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

cpt.meanvar,PELT.meanvar.poisson,binseg.meanvar.poisson,single.meanvar.poisson,segneigh.meanvar.poiss

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated Poisson data
set.seed(1)
x=c(rpois(50,lambda=1),rpois(50,lambda=5),rpois(50,lambda=1),rpois(50,lambda=10))
multiple.meanvar.poisson(x,mul.method="BinSeg",penalty="Manual",pen.value="2*log(n)",Q=5,
class=FALSE) # returns optimal number of changepoints is 3, locations are 50,100,150.

# Example multiple datasets where the first row has multiple changes in mean and variance and the
#second row has no change in mean or variance
set.seed(1)
x=c(rpois(50,lambda=1),rpois(50,lambda=5),rpois(50,lambda=1),rpois(50,lambda=10))
y=rpois(200,lambda=1)
z=rbind(x,y)
multiple.meanvar.poisson(z,mul.method="SegNeigh",penalty="Manual",pen.value=2*log(200),Q=5,
class= FALSE) # returns list that has two elements, the first has 3 changes in mean and variance at
#50,100,150 and the second has no changes in mean or variance
ans=multiple.meanvar.poisson(z,mul.method="PELT",penalty="Manual",pen.value=2*log(200))
cpts(ans[[1]]) # same results as for the SegNeigh method.
cpts(ans[[2]]) # same results as for the SegNeigh method.
```

`multiple.var.css` *Multiple Changes in Variance - Cumulative Sums of Squares*

Description

Calculates the optimal positioning and number of changepoints for the cumulative sums of squares test statistic using the user specified method.

Usage

```
multiple.var.css(data,mul.method="BinSeg",penalty="SIC",pen.value=0,Q=5,class=TRUE,
param.estimates=TRUE)
```

Arguments

`data`	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`mul.method`	Choice of "SegNeigh" or "BinSeg".
`penalty`	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
`pen.value`	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
`Q`	The maximum number of changepoints to search for using the "BinSeg" method. The maximum number of segments (number of changepoints + 1) to search for using the "SegNeigh" method.
`class`	Logical. If TRUE then an object of class `cpt` is returned.
`param.estimates`	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find multiple changes in variance for data where no assumption about the distribution is made. The changes are found using the method supplied which can be exact (SegNeigh) or approximate (BinSeg). Note that the penalty values are log(.) to be comparable with the distributional penalties.

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are solely returned if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a vector/list is returned depending on the value of mul.method. If data is a matrix (multiple datasets) then a list is returned where each element in the list is either a vector or list depending on the value of mul.method.

If mul.method is SegNeigh then a list is returned with elements:

`cps`	Matrix containing the changepoint positions for 1,...,Q changepoints.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`pen`	Penalty used to find the optimal number of changepoints.

If mul.method is BinSeg then a list is returned with elements:

`cps`	2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.
`op.cpts`	The optimal changepoint locations for the penalty supplied.
`pen`	Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

C. Inclan, G. C. Tiao (1994) Use of Cumulative Sums of Squares for Retrospective Detection of Changes of Variance, *Journal of the American Statistical Association* **89(427)**, 913–923

R. L. Brown, J. Durbin, J. M. Evans (1975) Techniques for Testing the Constancy of Regression Relationships over Time, *Journal of the Royal Statistical Society B* **32(2)**, 149–192

See Also

`multiple.mean.cusum`,`cpt.var`,`binseg.var.css`,`single.var.css`,`segneigh.var.css`,`cpt`

Examples

```
# Example of multiple changes in variance at 50,100,150 in simulated data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
multiple.var.css(x,mul.method="BinSeg",penalty="Manual",pen.value="log(2*log(n))",Q=5,class=FALSE)
#returns optimal number of changepoints is 4, locations are 50,53,99,150.

# Example multiple datasets where the first row has multiple changes in variance and the second row
#has no change in variance
```

```
set.seed(10)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
y=rnorm(200,0,1)
z=rbind(x,y)
multiple.var.css(z,mul.method="SegNeigh",penalty="Asymptotic",pen.value=0.01,Q=5,class=FALSE)
# returns list that has two elements, the first has 3 changes in variance at 52,100,149 and the
#second has no changes in variance
ans=multiple.var.css(z,mul.method="SegNeigh",penalty="Asymptotic",pen.value=0.01)
cpts(ans[[1]]) # same results as for class=FALSE.
cpts(ans[[2]]) # same results as for class=FALSE.
```

`multiple.var.norm` *Multiple Changes in Variance - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using the user specified method.

Usage

```
multiple.var.norm(data,mul.method="PELT",penalty="SIC",pen.value=0,Q=5,know.mean=FALSE,
mu=NA,class=TRUE,param.estimates=TRUE)
```

Arguments

`data`	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`mul.method`	Choice of "PELT", "SegNeigh" or "BinSeg".
`penalty`	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
`pen.value`	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
`Q`	The maximum number of changepoints to search for using the "BinSeg" method. The maximum number of segments (number of changepoints + 1) to search for using the "SegNeigh" method.
`know.mean`	Logical, if TRUE then the mean is assumed known and mu is taken as its value. If FALSE, and mu=-1000 (default value) then the mean is estimated via maximum likelihood. If FALSE and the value of mu is supplied, mu is not estimated but is counted as an estimated parameter for decisions.

`mu` Numerical value of the true mean of the data. Either single value or vector of length nrow(data). If data is a matrix and mu is a single value, the same mean is used for each row.

`class` Logical. If TRUE then an object of class `cpt` is returned.

`param.estimates` Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find multiple changes in variance for data that is assumed to be normally distributed. The changes are found using the method supplied which can be exact (PELT or SegNeigh) or approximate (BinSeg).

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are solely returned if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a vector/list is returned depending on the value of mul.method. If data is a matrix (multiple datasets) then a list is returned where each element in the list is either a vector or list depending on the value of mul.method.

If mul.method is PELT then a vector is returned:

`cpt` Vector containing the changepoint locations for the penalty supplied. This always ends with n.

If mul.method is SegNeigh then a list is returned with elements:

`cps` Matrix containing the changepoint positions for 1,...,Q changepoints.

`op.cpts` The optimal changepoint locations for the penalty supplied.

`like` Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

If mul.method is BinSeg then a list is returned with elements:

`cps` 2xQ Matrix containing the changepoint positions on the first row and the test statistic on the second row.

`op.cpts` The optimal changepoint locations for the penalty supplied.

`pen` Penalty used to find the optimal number of changepoints.

Author(s)

Rebecca Killick

References

Change in variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

Binary Segmentation: Scott, A. J. and Knott, M. (1974) A Cluster Analysis Method for Grouping Means in the Analysis of Variance, *Biometrics* **30(3)**, 507–512

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

multiple.mean.norm,multiple.meanvar.norm,cpt.var,PELT.var.norm,binseg.var.norm,single.var.norm,segneig

Examples

```
# Example of multiple changes in variance at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
multiple.var.norm(x,mul.method="BinSeg",penalty="Manual",pen.value="2*log(n)",Q=5,class=FALSE)
# returns optimal number of changepoints is 3, locations are 50,99,150.

# Example multiple datasets where the first row has multiple changes in variance and the second row
#has no change in variance
set.seed(10)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
y=rnorm(200,0,1)
z=rbind(x,y)
multiple.var.norm(z,mul.method="SegNeigh",penalty="Asymptotic",pen.value=0.01,Q=5,class=FALSE)
# returns list that has two elements, the first has 3 changes in variance at 50,100,149 and the
#second has no changes in variance
ans=multiple.var.norm(z,mul.method="PELT",penalty="Asymptotic",pen.value=0.01)
cpts(ans[[1]]) # same results as for the SegNeigh method.
cpts(ans[[2]]) # same results as for the SegNeigh method.
```

---

ncpts *Generic Function - ncpts*

Description

Generic function

Usage

```
ncpts(object)
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

ncpts-methods

Examples

```
x=new("cpt") # new cpt object
ncpts(x) # returns the number of changepoints (i.e. length of the cpts slot in x minus 1)
```

---

ncpts-methods *~~ Methods for Function ncpts ~~*

Description

~~ Methods for function ncpts ~~

Methods

signature(object = "cpt") Returns the number of changepoints (i.e. length of the cpts slot minus 1) from an object of class cpt

signature(object = "cpt.reg") Returns the number of changepoints (i.e. length of the cpts slot minus 1) from an object of class cpt.reg

ncpts.max *Generic Function - ncpts.max*

Description

Generic function

Usage

```
ncpts.max(object)
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)

Details

Generic function.

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

ncpts.max-methods

Examples

```
x=new("cpt") # new cpt object
ncpts.max(x) # retrieves the ncpts.max slot from x
```

ncpts.max-methods *~~ Methods for Function ncpts.max ~~*

Description

~~ Methods for function ncpts.max ~~

Methods

signature(object = "cpt") Retrieves ncpts.max slot from an object of class cpt

signature(object = "cpt.reg") Retrieves ncpts.max slot from an object of class cpt.reg

ncpts.max<- *Generic Function - ncpts.max<-*

Description

Generic function

Usage

```
ncpts.max(object)<-value
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)
value	Replacement value

Details

Generic function.

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

ncpts.max<--methods

Examples

```
x=new("cpt") # new cpt object
ncpts.max(x)<-10 # replaces the vector of changepoint in object x with 10
```

---

ncpts.max<--methods ~~ *Methods for Function ncpts.max<-* ~~

Description

~~ Methods for function ncpts.max<- ~~

Methods

signature(x = "cpt") Assigns the value following <- to the ncpts.max slot in x

signature(x = "cpt.reg") Assigns the value following <- to the ncpts.max slot in x

---

param *Generic Function - param*

Description

Generic function that returns parameter estimates.

Usage

```
param(object,...)
```

Arguments

object	Depending on the class of object depends on the method used to find the parameter estimates (and if one exists)
...	Other variables that may be required depending on the class of object, see individual methods.

Details

Generic Function that returns parameter estimates

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

param-methods,cpt.mean,cpt.var,cpt.meanvar

Examples

```
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,0,10))
ans=cpt.var(x,penalty="Asymptotic",pen.value=0.01,method="AMOC",param.estimates=FALSE)
# if estimates were not requested in the analysis then they can be created at a later stage if
#required
ans=param(ans) # fills the param.est slot with the parameter estimes.
param.est(ans) # variance is 0.8067621
```

---

param-methods *~~ Methods for Function param ~~*

Description

~~ Methods for function param ~~

Methods

`signature(object = "cpt")` Replaces the `param.value` slot in object with the parameter estimates that are appropriate for the changepoint type (cpttype slot). If the Gamma test statistic is used then the shape parameter is required as a variable.

`signature(object = "cpt.reg")` Replaces the `param.value` slot in object with the parameter estimates that are appropriate for the changepoint test statistic (test.stat slot).

---

param.est *Generic Function - param.est*

Description

Generic function

Usage

```
param.est(object)
```

Arguments

`object`	Depending on the class of `object` depends on the method used (and if one exists)

Details

Generic Function

Value

Depends on the class of `object`, see individual methods

Author(s)

Rebecca Killick

See Also

`param.est-methods`

Examples

```
x=new("cpt") # new cpt object
param.est(x) # retrieves the param.est slot from x
```

`param.est-methods` *~~ Methods for Function param.est ~~*

Description

~~ Methods for function `param.est` ~~

Methods

`signature(object = "cpt")` Retrieves param.est slot from an object of class cpt

`signature(object = "cpt.reg")` Retrieves param.est slot from an object of class cpt.reg

`param.est<-` *Generic Function - param.est<-*

Description

Generic function

Usage

```
param.est(object)<-value
```

Arguments

`object`	Depending on the class of `object` depends on the method used (and if one exists)
`value`	Replacement value

Details

Generic Function

Value

Depends on the class of `object`, see individual methods

Author(s)

Rebecca Killick

See Also

`param.est<--methods`

Examples

```
x=new("cpt") # new cpt object
param.est(x)<-list(mean=0) # replaces the current param.est list in x with list(mean=0)
```

---

param.est<--methods ~~ *Methods for Function param.est<-* ~~

Description

~~ Methods for function `param.est<-` ~~

Methods

`signature(x = "cpt")` Assigns the value following <- to the param.est slot in x

`signature(x = "cpt.reg")` Assigns the value following <- to the param.est slot in x

---

PELT.mean.norm *Multiple Changes in Mean using PELT pruned method - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using PELT pruned method.

Usage

```
PELT.mean.norm(data, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
pen	Numeric value of the linear penalty function. This value is used in the decision for each individual changepoint so that in total the penalty is k*pen where k is the optimal number of changepoints detected.

Details

This function is used to find a multiple changes in mean for data that is assumed to be normally distributed. The value returned is the result of testing H0:existing number of changepoints against H1: one extra changepoint using the log of the likelihood ratio statistic coupled with the penalty supplied. The PELT method keeps track of the optimal number and location of changepoints as it passes through the data.

Value

A vector of the changepoint locations is returned:

cpt	Vector containing the changepoint locations for the penalty supplied. This always ends with n.

Author(s)

Rebecca Killick

References

Change in Normal mean: Hinkley, D. V. (1970) Inference About the Change-Point in a Sequence of Random Variables, *Biometrika* **57**, 1–17

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

PELT.var.norm,PELT.meanvar.norm,cpt.mean,binseg.mean.norm,multiple.mean.norm,single.mean.norm,segneigh

Examples

```
# Example of multiple changes in mean at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,1),rnorm(50,10,1),rnorm(50,3,1))
PELT.mean.norm(x,pen=2*log(200)) # returns c(50,100,150,200)

# Example no change in mean
set.seed(10)
x=rnorm(200,0,1)
PELT.mean.norm(x,pen=2*log(200)) # returns 200 to show no change in mean has been found
```

`PELT.meanvar.exp` *Multiple Changes in Mean and Variance using PELT pruned method - Exponential Data*

Description

Calculates the optimal positioning and number of changepoints for Exponential data using PELT pruned method.

Usage

```
PELT.meanvar.exp(data, pen=0)
```

Arguments

`data`	A vector containing the data within which you wish to find changepoints.
`pen`	Numeric value of the linear penalty function. This value is used in the decision for each individual changepoint so that in total the penalty is k*pen where k is the optimal number of changepoints detected.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be Exponential distributed. The value returned is the result of testing H0:existing number of changepoints against H1: one extra changepoint using the log of the likelihood ratio statistic coupled with the penalty supplied. The PELT method keeps track of the optimal number and location of changepoints as it passes through the data.

Value

A vector of the changepoint locations is returned:

`cpt`	Vector containing the changepoint locations for the penalty supplied. This always ends with n.

Author(s)

Rebecca Killick

References

Change in Exponential model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

PELT.meanvar.norm,PELT.meanvar.gamma,cpt.meanvar,binseg.meanvar.exp,multiple.meanvar.exp,single.meanva

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated Exponential data
set.seed(1)
x=c(rexp(50,rate=1),rexp(50,rate=3),rexp(50,rate=1),rexp(50,rate=10))
PELT.meanvar.exp(x,pen=2*log(200)) # returns c(53,100,150,200)

# Example no change in rate parameter
set.seed(1)
x=rexp(200,rate=1)
PELT.meanvar.exp(x,pen=2*log(200)) # returns 200 to show no change in mean or variance has been
#found
```

PELT.meanvar.gamma *Multiple Changes in Mean and Variance using PELT pruned method - Gamma Data (i.e. change in scale parameter)*

Description

Calculates the optimal positioning and number of changepoints for Gamma data using PELT pruned method.

Usage

```
PELT.meanvar.gamma(data, shape=1, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
shape	Numerical value of the true shape parameter for the data. Either single value or vector of length nrow(data). If data is a matrix and shape is a single value, the same shape parameter is used for each row.
pen	Numeric value of the linear penalty function. This value is used in the decision for each individual changepoint so that in total the penalty is k*pen where k is the optimal number of changepoints detected.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be Gamma distributed. The value returned is the result of testing H0:existing number of changepoints against H1: one extra changepoint using the log of the likelihood ratio statistic coupled with the penalty supplied. The PELT method keeps track of the optimal number and location of changepoints as it passes through the data.

Value

A vector of the changepoint locations is returned:

cpt	Vector containing the changepoint locations for the penalty supplied. This always ends with n.

Author(s)

Rebecca Killick

References

Change in Gamma shape parameter: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

PELT.meanvar.norm,cpt.meanvar,binseg.meanvar.gamma,multiple.meanvar.gamma,single.meanvar.gamma,segneig

Examples

```
# Example of multiple changes in scale parameter at 50,100,150 in simulated gamma data
set.seed(1)
x=c(rgamma(50,shape=1,rate=1),rgamma(50,shape=1,rate=3),rgamma(50,shape=1,rate=1),rgamma(50,shape=1,
rate=10))
PELT.meanvar.gamma(x,pen=2*log(200)) # returns c(47,102,150,200)

# Example no change in scale parameter
set.seed(1)
x=rgamma(200,shape=1,rate=1)
PELT.meanvar.gamma(x,pen=2*log(200)) # returns 200 to show no change in mean or variance has been
#found
```

PELT.meanvar.norm *Multiple Changes in Mean and Variance using PELT pruned method - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using PELT pruned method.

Usage

```
PELT.meanvar.norm(data, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
pen	Numeric value of the linear penalty function. This value is used in the decision for each individual changepoint so that in total the penalty is k*pen where k is the optimal number of changepoints detected.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be normally distributed. The value returned is the result of testing H0:existing number of changepoints against H1: one extra changepoint using the log of the likelihood ratio statistic coupled with the penalty supplied. The PELT method keeps track of the optimal number and location of changepoints as it passes through the data.

Value

A vector of the changepoint locations is returned:

cpt	Vector containing the changepoint locations for the penalty supplied. This always ends with n.

Author(s)

Rebecca Killick

References

Change in Normal mean and variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

PELT.var.norm,PELT.mean.norm,cpt.meanvar,binseg.meanvar.norm,multiple.meanvar.norm,single.meanvar.norm

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,3),rnorm(50,10,1),rnorm(50,3,10))
PELT.meanvar.norm(x,pen=4*log(200)) # returns c(50,100,150,200)

# Example no change in mean or variance
set.seed(1)
x=rnorm(200,0,1)
PELT.meanvar.norm(x,pen=4*log(200)) # returns 200 to show no change in mean or variance has been
#found
```

`PELT.meanvar.poisson` *Multiple Changes in Mean and Variance using PELT pruned method - Poisson Data*

Description

Calculates the optimal positioning and number of changepoints for Poisson data using PELT pruned method.

Usage

```
PELT.meanvar.poisson(data, pen=0)
```

Arguments

`data`	A vector containing the data within which you wish to find changepoints.
`pen`	Numeric value of the linear penalty function. This value is used in the decision for each individual changepoint so that in total the penalty is k*pen where k is the optimal number of changepoints detected.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be Poisson distributed. The value returned is the result of testing H0:existing number of changepoints against H1: one extra changepoint using the log of the likelihood ratio statistic coupled with the penalty supplied. The PELT method keeps track of the optimal number and location of changepoints as it passes through the data.

Value

A vector of the changepoint locations is returned:

`cpt`	Vector containing the changepoint locations for the penalty supplied. This always ends with n.

Author(s)

Rebecca Killick

References

Change in Poisson model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

cpt.meanvar,binseg.meanvar.poisson,multiple.meanvar.poisson,single.meanvar.poisson,segneigh.meanvar.p

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated Poisson data
set.seed(1)
x=c(rpois(50,lambda=1),rpois(50,lambda=3),rpois(50,lambda=1),rpois(50,lambda=10))
PELT.meanvar.poisson(x,pen=2*log(200)) # returns c(50,100,150,200)

# Example no change in parameter
set.seed(1)
x=rpois(200,lambda=1)
PELT.meanvar.poisson(x,pen=2*log(200)) # returns 200 to show no change in mean or variance has been
#found
```

PELT.var.norm *Multiple Changes in Variance using PELT pruned method - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using PELT pruned method.

Usage

```
PELT.var.norm(data, pen=0, know.mean=FALSE, mu=NA)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
pen	Numeric value of the linear penalty function. This value is used in the decision for each individual changepoint so that in total the penalty is k*pen where k is the optimal number of changepoints detected.
know.mean	Logical, if TRUE then the mean is assumed known and mu is taken as its value. If FALSE, and mu=-1000 (default value) then the mean is estimated via maximum likelihood. If FALSE and the value of mu is supplied, mu is not estimated but is counted as an estimated parameter for decisions.
mu	Numerical value of the true mean of the data. Either single value or vector of length nrow(data). If data is a matrix and mu is a single value, the same mean is used for each row.

Details

This function is used to find a multiple changes in variance for data that is assumed to be normally distributed. The value returned is the result of testing H0:existing number of changepoints against H1: one extra changepoint using the log of the likelihood ratio statistic coupled with the penalty supplied. The PELT method keeps track of the optimal number and location of changepoints as it passes through the data.

Value

A vector of the changepoint locations is returned:

`cpt`	Vector containing the changepoint locations for the penalty supplied. This always ends with n.

Author(s)

Rebecca Killick

References

Change in variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

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

`PELT.mean.norm`,`PELT.meanvar.norm`,`cpt.var`,`binseg.var.norm`,`multiple.var.norm`,`single.var.norm`,`segneigh.va`

Examples

```
# Example of multiple changes in variance at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
PELT.var.norm(x,pen=2*log(200)) # returns c(50,99,150,200)

# Example no change in variance
set.seed(10)
x=rnorm(200,0,1)
PELT.var.norm(x,pen=2*log(200)) # returns 200 to show no change in variance has been found
```

pen.type *Generic Function - pen.type*

Description

Generic function

Usage

```
pen.type(object)
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

pen.type-methods

Examples

```
x=new("cpt") # new cpt object
pen.type(x) # retrieves the pen.type slot from x
```

pen.type-methods *~~ Methods for Function pen.type ~~*

Description

~~ Methods for function pen.type ~~

Methods

signature(object = "cpt") Retrieves pen.type slot from an object of class cpt

signature(object = "cpt.reg") Retrieves pen.type slot from an object of class cpt.reg

pen.type<- *Generic Function - pen.type<-*

Description

Generic function

Usage

```
pen.type(object)<-value
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)
value	Replacement value

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

pen.type<--methods

Examples

```
x=new("cpt") # new cpt object
pen.type(x)<-"SIC" # replaces the existing pen.type slot in x with "SIC"
```

---

pen.type<--methods *~~ Methods for Function pen.type<- ~~*

Description

~~ Methods for function pen.type<- ~~

Methods

signature(x = "cpt") Assigns the value following <- to the pen.type slot in x

signature(x = "cpt.reg") Assigns the value following <- to the pen.type slot in x

---

pen.value *Generic Function - pen.value*

Description

Generic function

Usage

```
pen.value(object)
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

pen.value-methods

Examples

```
x=new("cpt") # new cpt object
pen.value(x) # retrieves the pen.value slot from x
```

pen.value-methods *~~ Methods for Function pen.value ~~*

Description

~~ Methods for function pen.value ~~

Methods

signature(object = "cpt") Retrieves pen.value slot from an object of class cpt

signature(object = "cpt.reg") Retrieves pen.value slot from an object of class cpt.reg

pen.value<- *Generic Function - pen.value<-*

Description

Generic function

Usage

```
pen.value(object)<-value
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)
value	Replacement value

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

pen.value<--methods

Examples

```
x=new("cpt") # new cpt object
pen.value(x)<-5 # replaces the existing pen.value slot in x with 5
```

---

pen.value<--methods *~~ Methods for Function pen.value<- ~~*

Description

~~ Methods for function pen.value<- ~~

Methods

signature(x = "cpt") Assigns the value following <- to the pen.value slot in x

signature(x = "cpt.reg") Assigns the value following <- to the pen.value slot in x

---

plot-methods *~~ Methods for Function plot in Package 'graphics' ~~*

Description

~~ Methods for function plot in Package 'graphics' ~~

Methods

signature(x = "ANY") Generic plot function, see graphics package description using ?plot

signature(x = "cpt") Plots the data and identifies the changepoints using vertical lines (change in variance), horizontal lines (change in mean). Optional arguments to control the lines: cpt.col equivilent to col to change the colour of the changepoint line; cpt.width equivilent to lwd to change the width of the changepoint line; cpt.style equivilent to lty to change the style of the line.

signature(x = "cpt.reg") Plotting is only valid for one regressor. Plots the regressor against the response and identifies the changepoints using horizontal lines. Optional arguments to control the lines: cpt.col equivilent to col to change the colour of the changepoint line; cpt.width equivilent to lwd to change the width of the changepoint line; cpt.style equivilent to lty to change the style of the line.

print-methods *~~ Methods for Function print in Package 'base' ~~*

Description

~~ Methods for function print in Package 'base' ~~

Methods

signature(x = "ANY") Generic print function, see base package description using ?print

signature(x = "cpt") Prints out information contained within the object x including a summary

signature(x = "cpt.reg") Prints out information contained within the object x including a summary

seg.len *Generic Function - seg.len*

Description

Generic function

Usage

```
seg.len(object)
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

seg.len-methods

Examples

```
x=new("cpt") # new cpt object
seg.len(x) # returns the length of each segment in the data (i.e. no. of obs between changepoints)
```

`seg.len-methods` *~~ Methods for Function seg.len ~~*

Description

~~ Methods for function `seg.len` ~~

Methods

`signature(object = "cpt")` Returns the length of segments from an object of class cpt, i.e. the number of observations between changepoints

`signature(object = "cpt.reg")` Returns the length of segments from an object of class cpt.reg, i.e. the number of observations between changepoints

---

`segneigh.mean.cusum` *Multiple Changes in Mean using Segment Neighbourhood method - Cumulative Sums*

Description

Calculates the optimal positioning and number of changepoints for Cumulative Sums test statistic using Segment Neighbourhood method.

Usage

```
segneigh.mean.cusum(data, Q=5, pen=0)
```

Arguments

`data`	A vector containing the data within which you wish to find changepoints.
`Q`	Numeric value of the maximum number of segments (number of changepoints +1) you wish to search for, default is 5.
`pen`	Numeric value of the linear penalty function. This value is used in the final decision as to the optimal number of changepoints.

Details

This function is used to find a multiple changes in mean for data that is not assumed to have a particular distribution. The value returned is the result of finding the optimal location of up to Q changepoints using the cumulative sums test statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using pen as the penalty function.

Value

A list is returned containing the following items

cps	Matrix containing the changepoint positions for 1,...,Q changepoints.
op.cpts	The optimal changepoint locations for the penalty supplied.

Author(s)

Rebecca Killick

References

M. Csorgo, L. Horvath (1997) Limit Theorems in Change-Point Analysis, *Wiley*

E. S. Page (1954) Continuous Inspection Schemes, *Biometrika* **41(1/2)**, 100–115

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

segneigh.mean.cusum,cpt.mean,multiple.mean.cusum,single.mean.cusum,binseg.mean.cusum

Examples

```
# Example of multiple changes in mean at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,1),rnorm(50,10,1),rnorm(50,3,1))
segneigh.mean.cusum(x,Q=5,pen=1) # returns optimal number as 3 and the locations as c(50,101,150)
segneigh.mean.cusum(x,Q=3,pen=1) # returns optimal number as 2 as this is the maximum number of
#changepoints it can find.  If you get the maximum number, you need to increase Q until this is not
#the case.

# Example no change in mean
set.seed(10)
x=rnorm(200,0,1)
segneigh.mean.cusum(x,Q=5,pen=1) # returns optimal number as 0
```

---

segneigh.mean.norm *Multiple Changes in Mean using Segment Neighbourhood method - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using Segment Neighbourhood method. Note that this gives the same results as PELT method but takes more computational time.

Usage

```
segneigh.mean.norm(data, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
Q	Numeric value of the maximum number of segments (number of changepoints +1) you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the final decision as to the optimal number of changepoints, used as k*pen where k is the number of changepoints to be tested.

Details

This function is used to find a multiple changes in mean for data that is assumed to be normally distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using k*pen as the penalty function where k is the number of changepoints tested (k in (1,Q)).

Value

A list is returned containing the following items

cps	Matrix containing the changepoint positions for 1,...,Q changepoints.
op.cpts	The optimal changepoint locations for the penalty supplied.
like	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

Author(s)

Rebecca Killick

References

Change in Normal mean: Hinkley, D. V. (1970) Inference About the Change-Point in a Sequence of Random Variables, *Biometrika* **57**, 1–17

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

segneigh.var.norm,segneigh.meanvar.norm,cpt.mean,PELT.mean.norm,multiple.mean.norm,single.mean.norm,bi

Examples

```
# Example of multiple changes in mean at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,1),rnorm(50,10,1),rnorm(50,3,1))
segneigh.mean.norm(x,Q=5,pen=2*log(200)) # returns optimal number as 3 and the locations as
#c(50,100,150)
segneigh.mean.norm(x,Q=3,pen=2*log(200)) # returns optimal number as 2 as this is the maximum number
#of changepoints it can find.  If you get the maximum number, you need to increase Q until this is
#not the case.

# Example no change in mean
set.seed(10)
x=rnorm(200,0,1)
segneigh.mean.norm(x,Q=5,pen=2*log(200)) # returns optimal number as 0
```

---

segneigh.meanvar.exp *Multiple Changes in Mean and Variance using Segment Neighbourhood method - Exponential Data*

Description

Calculates the optimal positioning and number of changepoints for Exponential data using Segment Neighbourhood method. Note that this gives the same results as PELT method but takes more computational time.

Usage

```
segneigh.meanvar.exp(data, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
Q	Numeric value of the maximum number of segments (number of changepoints +1) you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the final decision as to the optimal number of changepoints, used as k*pen where k is the number of changepoints to be tested.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be Exponential distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using k*pen as the penalty function where k is the number of changepoints tested (k in (1,Q)).

Value

A list is returned containing the following items

cps	Matrix containing the changepoint positions for 1,...,Q changepoints.
op.cpts	The optimal changepoint locations for the penalty supplied.
like	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

Author(s)

Rebecca Killick

References

Change in Exponential model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

segneigh.meanvar.norm,segneigh.meanvar.gamma,cpt.meanvar,PELT.meanvar.exp,multiple.meanvar.exp,single.

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated Exponential data
set.seed(1)
x=c(rexp(50,rate=1),rexp(50,rate=5),rexp(50,rate=1),rexp(50,rate=10))
segneigh.meanvar.exp(x,Q=5, pen=2*log(200)) # returns optimal number as 3 and the locations as
#c(50,100,150)
segneigh.meanvar.exp(x,Q=3, pen=2*log(200)) # returns optimal number as 2 as this is the maximum
#number of changepoints it can find.  If you get the maximum number, you need to increase Q until
#this is not the case.

# Example no change in mean or variance
set.seed(1)
x=rexp(200,rate=1)
segneigh.meanvar.exp(x,pen=2*log(200)) # returns optimal number as 0
```

---

segneigh.meanvar.gamma

Multiple Changes in Mean and Variance using Segment Neighbourhood method - Gamma Data (i.e. change in scale parameter)

Description

Calculates the optimal positioning and number of changepoints for Gamma data using Segment Neighbourhood method. Note that this gives the same results as PELT method but takes more computational time.

Usage

```
segneigh.meanvar.gamma(data, shape=1, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
shape	Numerical value of the true shape parameter for the data. Either single value or vector of length nrow(data). If data is a matrix and shape is a single value, the same shape parameter is used for each row.
Q	Numeric value of the maximum number of segments (number of changepoints +1) you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the final decision as to the optimal number of changepoints, used as k*pen where k is the number of changepoints to be tested.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be Gamma distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using k*pen as the penalty function where k is the number of changepoints tested (k in (1,Q)).

Value

A list is returned containing the following items

cps	Matrix containing the changepoint positions for 1,...,Q changepoints.
op.cpts	The optimal changepoint locations for the penalty supplied.
like	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

Author(s)

Rebecca Killick

References

Change in Gamma shape parameter: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

segneigh.meanvar.norm,cpt.meanvar,PELT.meanvar.gamma,multiple.meanvar.gamma,single.meanvar.gamma,binse

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated Gamma data
set.seed(1)
x=c(rgamma(50,shape=1,rate=1),rgamma(50,shape=1,rate=3),rgamma(50,shape=1,rate=1),rgamma(50,shape=1,
rate=10))
segneigh.meanvar.gamma(x,shape=1, Q=5, pen=2*log(200)) # returns optimal number as 3 and the
#locations as c(47,102,150)
segneigh.meanvar.gamma(x,shape=1, Q=3, pen=2*log(200)) # returns optimal number as 2 as this is the
#maximum number of changepoints it can find.  If you get the maximum number, you need to increase Q
#until this is not the case.

# Example no change in mean or variance
set.seed(1)
x=rgamma(200,shape=1,rate=1)
segneigh.meanvar.gamma(x,shape=1,pen=2*log(200)) # returns optimal number as 0
```

---

segneigh.meanvar.norm *Multiple Changes in Mean and Variance using Segment Neighbourhood method - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using Segment Neighbourhood method. Note that this gives the same results as PELT method but takes more computational time.

Usage

```
segneigh.meanvar.norm(data, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
Q	Numeric value of the maximum number of segments (number of changepoints +1) you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the final decision as to the optimal number of changepoints, used as k*pen where k is the number of changepoints to be tested.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be normally distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using k*pen as the penalty function where k is the number of changepoints tested (k in (1,Q)).

Value

A list is returned containing the following items

cps	Matrix containing the changepoint positions for 1,...,Q changepoints.
op.cpts	The optimal changepoint locations for the penalty supplied.
like	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

Author(s)

Rebecca Killick

References

Change in Normal mean and variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

segneigh.var.norm,segneigh.mean.norm,cpt.meanvar,PELT.meanvar.norm,multiple.meanvar.norm,single.meanva

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,5,3),rnorm(50,10,1),rnorm(50,3,10))
segneigh.meanvar.norm(x,Q=5, pen=4*log(200)) # returns optimal number as 3 and the locations as
#c(50,100,150)
segneigh.meanvar.norm(x,Q=3, pen=4*log(200)) # returns optimal number as 2 as this is the maximum
#number of changepoints it can find.  If you get the maximum number, you need to increase Q until
#this is not the case.

# Example no change in mean or variance
set.seed(1)
x=rnorm(200,0,1)
segneigh.meanvar.norm(x,pen=4*log(200)) # returns optimal number as 0
```

---

segneigh.meanvar.poisson

Multiple Changes in Mean and Variance using Segment Neighbourhood method - Poisson Data

---

Description

Calculates the optimal positioning and number of changepoints for Poisson data using Segment Neighbourhood method. Note that this gives the same results as PELT method but takes more computational time.

Usage

```
segneigh.meanvar.poisson(data, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
Q	Numeric value of the maximum number of segments (number of changepoints +1) you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the final decision as to the optimal number of changepoints, used as k*pen where k is the number of changepoints to be tested.

Details

This function is used to find a multiple changes in mean and variance for data that is assumed to be Poisson distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using k*pen as the penalty function where k is the number of changepoints tested (k in (1,Q)).

Value

A list is returned containing the following items

cps	Matrix containing the changepoint positions for 1,...,Q changepoints.
op.cpts	The optimal changepoint locations for the penalty supplied.
like	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

Author(s)

Rebecca Killick

References

Change in Poisson model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

cpt.meanvar,PELT.meanvar.poisson,multiple.meanvar.poisson,single.meanvar.poisson,binseg.meanvar.poiss

Examples

```
# Example of multiple changes in mean and variance at 50,100,150 in simulated Poisson data
set.seed(1)
x=c(rpois(50,lambda=1),rpois(50,lambda=5),rpois(50,lambda=1),rpois(50,lambda=10))
segneigh.meanvar.poisson(x,Q=5, pen=2*log(200)) # returns optimal number as 3 and the locations as
#c(50,100,150)
segneigh.meanvar.poisson(x,Q=3, pen=2*log(200)) # returns optimal number as 2 as this is the maximum
#number of changepoints it can find.  If you get the maximum number, you need to increase Q until
#this is not the case.

# Example no change in mean or variance
set.seed(1)
x=rpois(200,lambda=1)
segneigh.meanvar.poisson(x,pen=2*log(200)) # returns optimal number as 0
```

---

segneigh.var.css *Multiple Changes in Variance using Segment Neighbourhood method - Cumulative Sums of Squares*

Description

Calculates the optimal positioning and number of changepoints for Cumulative Sums of Sqaures test statistic using Segment Neighbourhood method.

Usage

```
segneigh.var.css(data, Q=5, pen=0)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
Q	Numeric value of the maximum number of segments (number of changepoints +1) you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the final decision as to the optimal number of changepoints.

Details

This function is used to find a multiple changes in variance for data that is not assumed to have a particular distribution. The value returned is the result of finding the optimal location of up to Q changepoints using the cumulative sums of squares test statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using pen as the penalty function.

Value

A list is returned containing the following items

cps	Matrix containing the changepoint positions for 1,...,Q changepoints.
op.cpts	The optimal changepoint locations for the penalty supplied.

Author(s)

Rebecca Killick

References

C. Inclan, G. C. Tiao (1994) Use of Cumulative Sums of Squares for Retrospective Detection of Changes of Variance, *Journal of the American Statistical Association* **89(427)**, 913–923

R. L. Brown, J. Durbin, J. M. Evans (1975) Techniques for Testing the Constancy of Regression Relationships over Time, *Journal of the Royal Statistical Society B* **32(2)**, 149–192

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

segneigh.var.norm,cpt.var,multiple.var.css,single.var.css,binseg.var.css

Examples

```
# Example of multiple changes in variance at 50,100,150 in simulated normal data
set.seed(10)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
segneigh.var.css(x,Q=5, pen=1.368) # returns optimal number as 3 and the locations as c(52,100,149)
segneigh.var.css(x,Q=3, pen=1.368) # returns optimal number as 2 as this is the maximum number of
#changepoints it can find.  If you get the maximum number, you need to increase Q until this is not
#the case.
# 1.358 is the asymptotic value of the penalty for 95% confidence

# Example no change in variance
set.seed(1)
x=rnorm(200,0,1)
segneigh.var.css(x,Q=5, pen=1.368) # returns optimal number as 0
```

---

segneigh.var.norm *Multiple Changes in Variance using Segment Neighbourhood method - Normal Data*

Description

Calculates the optimal positioning and number of changepoints for Normal data using Segment Neighbourhood method. Note that this gives the same results as PELT method but takes more computational time.

Usage

```
segneigh.var.norm(data, Q=5, pen=0, know.mean=FALSE, mu=NA)
```

Arguments

data	A vector containing the data within which you wish to find changepoints.
Q	Numeric value of the maximum number of segments (number of changepoints +1) you wish to search for, default is 5.
pen	Numeric value of the linear penalty function. This value is used in the final decision as to the optimal number of changepoints, used as k*pen where k is the number of changepoints to be tested.
know.mean	Logical, if TRUE then the mean is assumed known and mu is taken as its value. If FALSE, and mu=-1000 (default value) then the mean is estimated via maximum likelihood. If FALSE and the value of mu is supplied, mu is not estimated but is counted as an estimated parameter for decisions.
mu	Numerical value of the true mean of the data. Either single value or vector of length nrow(data). If data is a matrix and mu is a single value, the same mean is used for each row.

Details

This function is used to find a multiple changes in variance for data that is assumed to be normally distributed. The value returned is the result of finding the optimal location of up to Q changepoints using the log of the likelihood ratio statistic. Once all changepoint locations have been calculated, the optimal number of changepoints is decided using k*pen as the penalty function where k is the number of changepoints tested (k in (1,Q)).

Value

A list is returned containing the following items

cps	Matrix containing the changepoint positions for 1,...,Q changepoints.
op.cpts	The optimal changepoint locations for the penalty supplied.
like	Value of the -2*log(likelihood ratio) + penalty for the optimal number of changepoints selected.

Author(s)

Rebecca Killick

References

Change in variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

Segment Neighbourhoods: Auger, I. E. And Lawrence, C. E. (1989) Algorithms for the Optimal Identification of Segment Neighborhoods, *Bulletin of Mathematical Biology* **51(1)**, 39–54

See Also

segneigh.mean.norm,segneigh.meanvar.norm,cpt.var,PELT.var.norm,multiple.var.norm,single.var.norm,binse

Examples

```
# Example of multiple changes in variance at 50,100,150 in simulated normal data
set.seed(1)
x=c(rnorm(50,0,1),rnorm(50,0,10),rnorm(50,0,5),rnorm(50,0,1))
segneigh.var.norm(x,Q=5, pen=2*log(200)) # returns optimal number as 3 and the locations as
#c(50,99,150)
segneigh.var.norm(x,Q=3, pen=2*log(200)) # returns optimal number as 2 as this is the maximum number
#of changepoints it can find.  If you get the maximum number, you need to increase Q until this is
#not the case.

# Example no change in variance
set.seed(10)
x=rnorm(200,0,1)
segneigh.var.norm(x,Q=5, pen=2*log(200)) # returns optimal number as 0
```

---

single.mean.cusum *Single Change in Mean - Cumulative Sums*

Description

Calculates the cumulative sums (cusum) test statistic for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.mean.cusum(data,penalty="Asymptotic",pen.value=0.05,class=TRUE,
param.estimates=TRUE)
```

Arguments

data	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
penalty	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
pen.value	The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=test statistic, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
class	Logical. If TRUE then an object of class cpt is returned. If FALSE a vector is returned.
param.estimates	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find a single change in mean for data that is is not assumed to follow a specific distribtuion. The value returned is the result of testing H0:no change in mean against H1: single change in mean using the cumulative sums test statistic coupled with the penalty supplied.

Warning: The prescribed penalty values are not defined for use on CUSUM tests. The values tend to be too large and thus manual penalties are preferred.

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are solely returned if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a single value is returned:

`cpt`	The most probable location of a changepoint if H0 was rejected or NA if H1 was rejected.

If data is an mxn matrix (multiple datasets) then a vector is returned:

`cpt`	Vector of length m containing where each element is the result of the test for data[m,]. If cpt[m] is a number then it is the most probable location of a changepoint under H1. Otherwise cpt[m] has the value NA and indicates that H1 was rejected.

Author(s)

Rebecca Killick

References

M. Csorgo, L. Horvath (1997) Limit Theorems in Change-Point Analysis, *Wiley*

E. S. Page (1954) Continuous Inspection Schemes, *Biometrika* **41(1/2)**, 100–115

See Also

`cpt.mean`,`cpt`

Examples

```
# Example of a change in mean at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,10,1))
single.mean.cusum(x,penalty="Manual",pen.value=1,class=FALSE) # returns 101 to show that the null
#hypothesis was rejected and the change in mean is at 101
ans=single.mean.cusum(x,penalty="Manual",pen.value=1)
cpts(ans) # returns 101 to show that the null hypothesis was rejected, the change in mean is at 101


# Example of a data matrix containing 2 rows, row 1 has a change in mean and row 2 had no change in
#mean
set.seed(1)
```

```
x=c(rnorm(100,0,1),rnorm(100,10,1))
y=rnorm(200,0,1)
z=rbind(x,y)
single.mean.cusum(z,penalty="Manual",pen.value=1,class=FALSE) # returns vector c(101,200) which
#shows that the first dataset has a change in mean at 101 and the second dataset rejected H1 and has
#no change in mean
ans=single.mean.cusum(z,penalty="Manual",pen.value="log(log(n))")
cpts(ans[[1]]) # test using a manual penalty which is the log(SIC) penalty and gives the same values
#for this example
cpts(ans[[2]]) # result is the same as above, c(101, 200)
```

`single.mean.cusum.calc`

Single Change in Mean - Cumulative Sums

Description

Calculates the cumulative sums (cusum) test statistic for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.mean.cusum.calc(data, extrainf = TRUE)
```

Arguments

`data`	A vector or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`extrainf`	Logical, if TRUE the test statistic is returned along with the changepoint location. If FALSE, only the changepoint location is returned.

Details

This function is used to find a single change in mean for data where no distributional assumption is made. The changepoint returned is simply the location where the test statistic is maximised, there is no test performed as to whether this location is a true changepoint or not.

In reality this function should not be used unless you are performing a changepoint test using the output supplied. This function is used in the "see also" functions that perform various changepoint tests, ideally these should be used.

Value

If data is a vector (single dataset) and extrainf=FALSE then a single value is returned:

`cpt`	The most probable location of a changepoint

If data is a vector (single dataset) and extrainf=TRUE then a vector with two elements is returned:

`test statistic`	The cumulative sums test statistic

If data is an mxn matrix (multiple datasets) and extrainf=FALSE then a vector is returned:

cpt — Vector of length m containing the most probable location of a changepoint for each row in data. cpt[1] is the most probable changepoint of the first row in data and cpt[m] is the most probable changepoint for the final row in data.

If data is a matrix (multiple datasets) and extrainf=TRUE then a matrix is returned where the first column is the changepoint location for each row in data, the second column is the test statistic for each row in data.

Author(s)

Rebecca Killick

References

M. Csorgo, L. Horvath (1997) Limit Theorems in Change-Point Analysis, *Wiley*

E. S. Page (1954) Continuous Inspection Schemes, *Biometrika* **41(1/2)**, 100–115

See Also

single.mean.cusum, cpt.mean

Examples

```
# Example of a change in mean at 100 in simulated data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,10,1))
single.mean.cusum.calc(x,extrainf=FALSE) # finds change at 101
single.mean.cusum.calc(x) # finds change at 101 and gives the test statistic as 2.463326


# Example of no change in mean in simulated data
set.seed(1)
x=rnorm(100,0,1)
single.mean.cusum.calc(x,extrainf=FALSE) # finds change at 97, this is the most probable point of
#change but if a changepoint test is performed then no change will be found.
single.mean.cusum.calc(x)# change at 97, test statistic is 0.0398342
```

---

single.mean.norm *Single Change in Mean - Normal Data*

Description

Calculates the scaled log-likelihood (assuming the data is normally distributed) for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.mean.norm(data, penalty="SIC",pen.value=0,class=TRUE,param.estimates=TRUE)
```

Arguments

data	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
penalty	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
pen.value	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
class	Logical. If TRUE then an object of class cpt is returned. If FALSE a vector is returned.
param.estimates	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find a single change in mean for data that is assumed to be normally distributed. The value returned is the result of testing H0:no change in mean against H1: single change in mean using the log of the likelihood ratio statistic coupled with the penalty supplied.

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot cpts contains the changepoints that are returned (with p-value) if class=FALSE. The structure of cpts is as follows.

If data is a vector (single dataset) then a single value is returned:

cpt	The most probable location of a changepoint if H0 was rejected or NA if H1 was rejected.

If data is an mxn matrix (multiple datasets) then a vector is returned:

cpt	Vector of length m containing where each element is the result of the test for data[m,]. If cpt[m] is a number then it is the most probable location of a changepoint under H1. Otherwise cpt[m] has the value NA and indicates that H1 was rejected.

Author(s)

Rebecca Killick

References

Change in Normal mean: Hinkley, D. V. (1970) Inference About the Change-Point in a Sequence of Random Variables, *Biometrika* **57**, 1–17

See Also

cpt.mean,single.meanvar.norm,single.var.norm,cpt

Examples

```
# Example of a change in mean at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,10,1))
single.mean.norm(x,penalty="SIC",class=FALSE) # returns 100 to show that the null hypothesis was
#rejected and the change in mean is at 100
ans=single.mean.norm(x,penalty="Asymptotic",pen.value=0.01)
cpts(ans) # returns 100 to show that the null hypothesis was rejected, the change in mean is at 100
#and we are 99% confident of this result


# Example of a data matrix containing 2 rows, row 1 has a change in mean and row 2 had no change in
#mean
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,10,1))
y=rnorm(200,0,1)
z=rbind(x,y)
single.mean.norm(z,penalty="SIC",class=FALSE) # returns vector c(100,200) which shows that the first
#dataset has a change in mean at 100 and the second dataset rejected H1 and has no change in mean
ans=single.mean.norm(z,penalty="Manual",pen.value="log(n)")
cpts(ans[[1]]) # test using a manual penalty which is the same as the SIC penalty for this example
cpts(ans[[2]]) # result is the same as above, c(100, 200)
```

---

single.mean.norm.calc *Single Change in Mean - Normal Data*

Description

Calculates the scaled log-likelihood (assuming the data is normally distributed) for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.mean.norm.calc(data, extrainf = TRUE)
```

Arguments

data	A vector or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
extrainf	Logical, if TRUE the scaled null and alternative likelihood values are returned along with the changepoint location. If FALSE, only the changepoint location is returned.

Details

This function is used to find a single change in mean for data that is assumed to be normally distributed. The changepoint returned is simply the location where the log likelihood is maximised, there is no test performed as to whether this location is a true changepoint or not.

The returned likelihoods are scaled so that a test can be directly performed using the log of the likelihood ratio, $\lambda = \frac{1}{\sigma^2}\{null - alt\}$, which should be maximised.

In reality this function should not be used unless you are performing a changepoint test using the output supplied. This function is used in the "see also" functions that perform various changepoint tests, ideally these should be used.

Value

If data is a vector (single dataset) and extrainf=FALSE then a single value is returned:

`cpt`	The most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations

If data is a vector (single dataset) and extrainf=TRUE then a vector with three elements is returned:

`cpt`	The most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations
`null`	The scaled null likelihood (log likelihood of entire data with no change)
`alt`	The scaled alternative liklihood at cpt (log likelihood of entire data with a change at cpt)

If data is an mxn matrix (multiple datasets) and extrainf=FALSE then a vector is returned:

`cpt`	Vector of length m containing the most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations for each row in data. cpt[1] is the most probable changepoint of the first row in data and cpt[m] is the most probable changepoint for the final row in data.

If data is a matrix (multiple datasets) and extrainf=TRUE then a matrix is returned where the first column is the changepoint location for each row in data, the second column is the scaled null likelihood for each row in data, the final column is the scaled maximum of the alternative likelihoods for each row in data.

Author(s)

Rebecca Killick

References

Change in Normal mean: Hinkley, D. V. (1970) Inference About the Change-Point in a Sequence of Random Variables, *Biometrika* **57**, 1–17

See Also

`single.mean.norm`, `cpt.mean`

Examples

```
# Example of a change in mean at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,10,1))
single.mean.norm.calc(x,extrainf=FALSE) # finds change at 100
single.mean.norm.calc(x) # finds change at 100 and gives null likelihood as 5025.0857 and
#alternative likelihood as 170.7051


# Example of no change in mean in simulated normal data
set.seed(1)
x=rnorm(100,0,1)
single.mean.norm.calc(x,extrainf=FALSE) # finds change at 96, this is the most probable point of
#change but if a changepoint test is performed then no change will be found.
single.mean.norm.calc(x)# change at 96, null liklihood is 79.86945 and alternative is 75.73725
```

---

single.meanvar.exp *Single Change in Mean and Variance - Exponential Data*

Description

Calculates the scaled log-likelihood (assuming the data is Exponential distributed) for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.meanvar.exp(data,penalty="SIC",pen.value=0,class=TRUE,param.estimates=TRUE)
```

Arguments

data	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
penalty	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
pen.value	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
class	Logical. If TRUE then an object of class cpt is returned. If FALSE a vector is returned.
param.estimates	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find a single change in scale parameter (mean and variance) for data that is assumed to be Exponential distributed. The value returned is the result of testing H0:no change in mean or variance against H1: single change in mean and/or variance using the log of the likelihood ratio statistic coupled with the penalty supplied.

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are returned (with p-value) if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a single value is returned:

`cpt`	The most probable location of a changepoint if H0 was rejected or NA if H1 was rejected.

If data is an mxn matrix (multiple datasets) then a vector is returned:

`cpt`	Vector of length m containing where each element is the result of the test for data[m,]. If cpt[m] is a number then it is the most probable location of a changepoint under H1. Otherwise cpt[m] has the value NA and indicates that H1 was rejected.

Author(s)

Rebecca Killick

References

Change in Exponential model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

`cpt.meanvar`,`single.meanvar.norm`,`single.meanvar.gamma`,`cpt`

Examples

```
# Example of a change in mean and variance at 100 in simulated Exponential data
set.seed(10)
x=c(rexp(100,rate=1),rexp(100,rate=5))
single.meanvar.exp(x,penalty="SIC",class=FALSE) # returns 99 to show that the null hypothesis was
#rejected and the change in mean and variance is at 99
ans=single.meanvar.exp(x,penalty="AIC")
cpts(ans) # returns 99 to show that the null hypothesis was rejected, the change in mean and
#variance is at 99

# Example of a data matrix containing 2 rows, row 1 has a change in rate parameter and row 2 had no
#change in rate parameter
set.seed(10)
x=c(rexp(100,rate=1),rexp(100,rate=10))
y=rexp(200,rate=1)
```

```
z=rbind(x,y)
single.meanvar.exp(z,penalty="SIC",class=FALSE) # returns vector c(99,200) which shows that the
#first dataset has a change in scale parameter at 99 and the second dataset rejected H1 and has no
#change in scale parameter
ans=single.meanvar.exp(z,penalty="Manual",pen.value="diffparam*log(n)") # list returned
cpts(ans[[1]]) # test using a manual penalty which is the same as the SIC penalty for this example.
#The same changepoint is detected for the first dataset
cpts(ans[[2]]) # same as above, no change found
```

`single.meanvar.exp.calc`

Single Change in Mean and Variance - Exponential Data

Description

Calculates the scaled log-likelihood (assuming the data is Exponential distributed) for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.meanvar.exp.calc(data, extrainf = TRUE)
```

Arguments

`data`	A vector or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`extrainf`	Logical, if TRUE the scaled null and alternative likelihood values are returned along with the changepoint location. If FALSE, only the changepoint location is returned.

Details

This function is used to find a single change in mean and variance for data that is assumed to be Exponential distributed. The changepoint returned is simply the location where the log likelihood is maximised, there is no test performed as to whether this location is a true changepoint or not.

The returned likelihoods are scaled so that a test can be directly performed using the log of the likelihood ratio, $\lambda = \{null - alt\}$, which should be maximised.

In reality this function should not be used unless you are performing a changepoint test using the output supplied. This function is used in the "see also" functions that perform various changepoint tests, ideally these should be used.

Value

If data is a vector (single dataset) and extrainf=FALSE then a single value is returned:

`cpt`	The most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations

If data is a vector (single dataset) and extrainf=TRUE then a vector with three elements is returned:

cpt	The most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations
null	The scaled null likelihood (log likelihood of entire data with no change)
altlike	The scaled alternative liklihood at cpt (log likelihood of entire data with a change at cpt)

If data is an mxn matrix (multiple datasets) and extrainf=FALSE then a vector is returned:

cpt	Vector of length m containing the most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations for each row in data. cpt[1] is the most probable changepoint of the first row in data and cpt[m] is the most probable changepoint for the final row in data.

If data is a matrix (multiple datasets) and extrainf=TRUE then a matrix is returned where the first column is the changepoint location for each row in data, the second column is the scaled null likelihood for each row in data, the final column is the scaled maximum of the alternative likelihoods for each row in data.

Author(s)

Rebecca Killick

References

Change in Exponential Model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

single.meanvar.exp, cpt.meanvar

Examples

```
# Example of a change in mean and variance at 100 in simulated Exponential data
set.seed(10)
x=c(rexp(100,rate=1),rexp(100,rate=5))
single.meanvar.exp.calc(x,extrainf=FALSE) # finds change at 99
single.meanvar.exp.calc(x) # finds change at 99 and gives null likelihood as -192.5052 and
#alternative likelihood as -299.7263


# Example of no change in rate parameter (mean or variance) in simulated Exponential data
set.seed(10)
x=rexp(100,rate=1)
single.meanvar.exp.calc(x,extrainf=FALSE) # finds change at 90, this is the most probable point of
#change but if a changepoint test is performed then no change will be found.
single.meanvar.exp.calc(x)# change at 90, null liklihood is 3.196690 and alternative is 1.060281
```

`single.meanvar.gamma` *Single Change in Mean and Variance - Gamma Data (i.e. change in scale parameter)*

Description

Calculates the scaled log-likelihood (assuming the data is Gamma distributed) for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.meanvar.gamma(data,shape=1,penalty="SIC",pen.value=0,class=TRUE,
param.estimates=TRUE)
```

Arguments

`data`	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`shape`	Numerical value of the true shape parameter for the data. Either single value or vector of length nrow(data). If data is a matrix and shape is a single value, the same shape parameter is used for each row.
`penalty`	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
`pen.value`	The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
`class`	Logical. If TRUE then an object of class `cpt` is returned. If FALSE a vector is returned.
`param.estimates`	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find a single change in scale parameter (mean and variance) for data that is assumed to be Gamma distributed. The value returned is the result of testing H0:no change in mean or variance against H1: single change in mean and/or variance using the log of the likelihood ratio statistic coupled with the penalty supplied.

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are solely returned if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a single value is returned:

`cpt`	The most probable location of a changepoint if H0 was rejected or NA if H1 was rejected.

If data is an mxn matrix (multiple datasets) then a vector is returned:

`cpt`	Vector of length m containing where each element is the result of the test for data[m,]. If cpt[m] is a number then it is the most probable location of a changepoint under H1. Otherwise cpt[m] has the value NA and indicates that H1 was rejected.

Author(s)

Rebecca Killick

References

Change in Gamma scale parameter: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

`cpt.meanvar`,`single.meanvar.norm`,`cpt`

Examples

```
# Example of a change in scale parameter (mean and variance) at 100 in simulated gamma data
set.seed(1)
x=c(rgamma(100,shape=1,rate=1),rgamma(100,shape=1,rate=5))
single.meanvar.gamma(x,penalty="SIC",class=FALSE) # returns 97 to show that the null hypothesis was
#rejected and the change in scale parameter is at 97
ans=single.meanvar.gamma(x,penalty="AIC")
cpts(ans) # returns 97 to show that the null hypothesis was rejected, the change in scale parameter
#is at 97

# Example of a data matrix containing 2 rows, row 1 has a change in scale parameter and row 2 had no
#change in scale parameter
set.seed(10)
x=c(rgamma(100,shape=1,rate=1),rgamma(100,shape=1,rate=10))
y=rgamma(200,shape=1,rate=1)
z=rbind(x,y)
single.meanvar.gamma(z,penalty="SIC",class=FALSE) # returns vector c(99,200) which shows that the
#first dataset has a change in scale parameter at 99 and the second dataset rejected H1 and has no
#change in scale parameter
ans=single.meanvar.gamma(z,penalty="Manual",pen.value="diffparam*log(n)") # list returned
cpts(ans[[1]]) # test using a manual penalty which is the same as the SIC penalty for this example.
#The same changepoint is detected for the first dataset
cpts(ans[[2]]) # same as above, no change found
```

`single.meanvar.gamma.calc` *Single Change in Mean and Variance - Gamma Data (i.e. change in scale parameter)*

Description

Calculates the scaled log-likelihood (assuming the data is Gamma distributed) for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.meanvar.gamma.calc(data, shape=1, extrainf = TRUE)
```

Arguments

`data`	A vector or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`shape`	Numerical value of the true shape parameter for the data. Either single value or vector of length nrow(data). If data is a matrix and shape is a single value, the same shape parameter is used for each row.
`extrainf`	Logical, if TRUE the scaled null and alternative likelihood values are returned along with the changepoint location. If FALSE, only the changepoint location is returned.

Details

This function is used to find a single change in mean and variance for data that is assumed to be Gamma distributed. The changepoint returned is simply the location where the log likelihood is maximised, there is no test performed as to whether this location is a true changepoint or not.

The returned likelihoods are scaled so that a test can be directly performed using the log of the likelihood ratio, $\lambda = \{null - alt\}$, which should be maximised.

In reality this function should not be used unless you are performing a changepoint test using the output supplied. This function is used in the "see also" functions that perform various changepoint tests, ideally these should be used.

Value

If data is a vector (single dataset) and extrainf=FALSE then a single value is returned:

`cpt`	The most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations

If data is a vector (single dataset) and extrainf=TRUE then a vector with three elements is returned:

`cpt`	The most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations

null	The scaled null likelihood (log likelihood of entire data with no change)
altlike	The scaled alternative liklihood at cpt (log likelihood of entire data with a change at cpt)

If data is an mxn matrix (multiple datasets) and extrainf=FALSE then a vector is returned:

cpt	Vector of length m containing the most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations for each row in data. cpt[1] is the most probable changepoint of the first row in data and cpt[m] is the most probable changepoint for the final row in data.

If data is a matrix (multiple datasets) and extrainf=TRUE then a matrix is returned where the first column is the changepoint location for each row in data, the second column is the scaled null likelihood for each row in data, the final column is the scaled maximum of the alternative likelihoods for each row in data.

Author(s)

Rebecca Killick

References

Change in Gamma scale parameter: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

single.meanvar.gamma, cpt.meanvar

Examples

```
# Example of a change in scale parameter (mean and variance) at 100 in simulated gamma data
set.seed(1)
x=c(rgamma(100,shape=1,rate=1),rgamma(100,shape=1,rate=5))
single.meanvar.gamma.calc(x,extrainf=FALSE) # finds change at 97
single.meanvar.gamma.calc(x) # finds change at 97 and gives null likelihood as -230.8446 and
#alternative likelihood as -318.8848


# Example of no change in scale parameter (mean or variance) in simulated gamma data
set.seed(10)
x=rgamma(100,shape=1,rate=1)
single.meanvar.gamma.calc(x,extrainf=FALSE) # finds change at 72, this is the most probable point of
#change but if a changepoint test is performed then no change will be found.
single.meanvar.gamma.calc(x)# change at 72, null liklihood is -13.28644 and alternative is -16.27421
```

`single.meanvar.norm` *Single Change in Mean and Variance - Normal Data*

Description

Calculates the scaled log-likelihood (assuming the data is normally distributed) for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.meanvar.norm(data, penalty="SIC",pen.value=0,class=TRUE,param.estimates=TRUE)
```

Arguments

`data`	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`penalty`	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
`pen.value`	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
`class`	Logical. If TRUE then an object of class `cpt` is returned. If FALSE a vector is returned.
`param.estimates`	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find a single change in mean and variance for data that is assumed to be normally distributed. The value returned is the result of testing H0:no change in mean or variance against H1: single change in mean and/or variance using the log of the likelihood ratio statistic coupled with the penalty supplied.

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are returned (with p-value) if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a single value is returned:

cpt — The most probable location of a changepoint if H0 was rejected or NA if H1 was rejected.

If data is an mxn matrix (multiple datasets) then a vector is returned:

cpt — Vector of length m containing where each element is the result of the test for data[m,]. If cpt[m] is a number then it is the most probable location of a changepoint under H1. Otherwise cpt[m] has the value NA and indicates that H1 was rejected.

Author(s)

Rebecca Killick

References

Change in Normal mean and variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

cpt.meanvar,single.mean.norm,single.var.norm,cpt

Examples

```
# Example of a change in mean and variance at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,1,10))
single.meanvar.norm(x,penalty="SIC",class=FALSE) # returns 99 to show that the null hypothesis was
#rejected and the change in mean and variance is at 99
ans=single.meanvar.norm(x,penalty="Asymptotic",pen.value=0.01)
cpts(ans) # returns 99 to show that the null hypothesis was rejected, the change in mean and
#variance is at 99 and we are 99% confident of this result

# Example of a data matrix containing 2 rows, row 1 has a change in mean and variance and row 2 had
#no change in mean or variance
set.seed(10)
x=c(rnorm(100,0,1),rnorm(100,1,10))
y=rnorm(200,0,1)
z=rbind(x,y)
single.meanvar.norm(z,penalty="Asymptotic",pen.value=0.01,class=FALSE) # returns vector c(99,200)
#which shows that the first dataset has a change in mean and variance at 99 and the second dataset
#rejected H1 and has no change in mean or variance
ans=single.meanvar.norm(z,penalty="Manual",pen.value="diffparam*log(n)") # list returned
cpts(ans[[1]]) # test using a manual penalty which is the same as the SIC penalty for this example,
#result shows that the penalty is too small.  The same changepoint is detected for the first
#dataset
cpts(ans[[2]]) # but the second dataset returns a change in mean and variance at 198 which was
#rejected under the asymptotic penalty above.
```

`single.meanvar.norm.calc` *Single Change in Mean and Variance - Normal Data*

Description

Calculates the scaled log-likelihood (assuming the data is normally distributed) for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.meanvar.norm.calc(data, extrainf = TRUE)
```

Arguments

`data`	A vector or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`extrainf`	Logical, if TRUE the scaled null and alternative likelihood values are returned along with the changepoint location. If FALSE, only the changepoint location is returned.

Details

This function is used to find a single change in mean and variance for data that is assumed to be normally distributed. The changepoint returned is simply the location where the log likelihood is maximised, there is no test performed as to whether this location is a true changepoint or not.

The returned likelihoods are scaled so that a test can be directly performed using the log of the likelihood ratio, $\lambda = \{null - alt\}$, which should be maximised.

In reality this function should not be used unless you are performing a changepoint test using the output supplied. This function is used in the "see also" functions that perform various changepoint tests, ideally these should be used.

Value

If data is a vector (single dataset) and extrainf=FALSE then a single value is returned:

`cpt`	The most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations

If data is a vector (single dataset) and extrainf=TRUE then a vector with three elements is returned:

`cpt`	The most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations
`null`	The scaled null likelihood (log likelihood of entire data with no change)
`altlike`	The scaled alternative liklihood at cpt (log likelihood of entire data with a change at cpt)

If data is an mxn matrix (multiple datasets) and extrainf=FALSE then a vector is returned:

`cpt` Vector of length m containing the most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations for each row in data. cpt[1] is the most probable changepoint of the first row in data and cpt[m] is the most probable changepoint for the final row in data.

If data is a matrix (multiple datasets) and extrainf=TRUE then a matrix is returned where the first column is the changepoint location for each row in data, the second column is the scaled null likelihood for each row in data, the final column is the scaled maximum of the alternative likelihoods for each row in data.

Author(s)

Rebecca Killick

References

Change in Normal mean and variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

`single.meanvar.norm`, `cpt.meanvar`

Examples

```
# Example of a change in mean and variance at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,1,10))
single.meanvar.norm.calc(x,extrainf=FALSE) # finds change at 100
single.meanvar.norm.calc(x) # finds change at 100 and gives null likelihood as 765.2189 and
#alternative likelihood as 453.6101


# Example of no change in mean or variance in simulated normal data
set.seed(10)
x=rnorm(100,0,1)
single.meanvar.norm.calc(x,extrainf=FALSE) # finds change at 99, this is the most probable point of
#change but if a changepoint test is performed then no change will be found.
single.meanvar.norm.calc(x)# change at 99, null liklihood is -13.11733 and alternative is -37.16001
```

---

single.meanvar.poisson

Single Change in Mean and Variance - Poisson Data

Description

Calculates the scaled log-likelihood (assuming the data is Poisson distributed) for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.meanvar.poisson(data,penalty="SIC",pen.value=0,class=TRUE,param.estimates=TRUE)
```

Arguments

data	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
penalty	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
pen.value	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
class	Logical. If TRUE then an object of class cpt is returned. If FALSE a vector is returned.
param.estimates	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find a single change in scale parameter (mean and variance) for data that is assumed to be Poisson distributed. The value returned is the result of testing H0:no change in mean or variance against H1: single change in mean and/or variance using the log of the likelihood ratio statistic coupled with the penalty supplied.

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot cpts contains the changepoints that are solely returned if class=FALSE. The structure of cpts is as follows.

If data is a vector (single dataset) then a single value is returned:

cpt	The most probable location of a changepoint if H0 was rejected or NA if H1 was rejected.

If data is an mxn matrix (multiple datasets) then a vector is returned:

cpt	Vector of length m containing where each element is the result of the test for data[m,]. If cpt[m] is a number then it is the most probable location of a changepoint under H1. Otherwise cpt[m] has the value NA and indicates that H1 was rejected.

Author(s)

Rebecca Killick

References

Change in Poisson model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

cpt.meanvar,cpt

Examples

```
# Example of a change in mean and variance at 100 in simulated Poisson data
set.seed(10)
x=c(rpois(100,lambda=1),rpois(100,lambda=5))
single.meanvar.poisson(x,penalty="SIC",class=FALSE) # returns 99 to show that the null hypothesis
#was rejected and the change in mean and variance is at 99
ans=single.meanvar.poisson(x,penalty="AIC")
cpts(ans) # returns 99 to show that the null hypothesis was rejected, the change in mean and
#variance is at 99

# Example of a data matrix containing 2 rows, row 1 has a change in parameter and row 2 had no
#change in parameter
set.seed(1)
x=c(rpois(100,lambda=1),rpois(100,lambda=10))
y=rpois(200,lambda=1)
z=rbind(x,y)
single.meanvar.poisson(z,penalty="SIC",class=FALSE) # returns vector c(99,200) which shows that the
#first dataset has a change in scale parameter at 99 and the second dataset rejected H1 and has no
#change in scale parameter
ans=single.meanvar.poisson(z,penalty="Manual",pen.value="diffparam*log(n)") # list returned
cpts(ans[[1]]) # test using a manual penalty which is the same as the SIC penalty for this example.
#The same changepoint is detected for the first dataset.
cpts(ans[[2]]) # same as above, no change found
```

---

single.meanvar.poisson.calc

Single Change in Mean and Variance - Poisson Data

Description

Calculates the scaled log-likelihood (assuming the data is Poisson distributed) for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.meanvar.poisson.calc(data, extrainf = TRUE)
```

Arguments

data	A vector or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
extrainf	Logical, if TRUE the scaled null and alternative likelihood values are returned along with the changepoint location. If FALSE, only the changepoint location is returned.

Details

This function is used to find a single change in mean and variance for data that is assumed to be Poisson distributed. The changepoint returned is simply the location where the log likelihood is maximised, there is no test performed as to whether this location is a true changepoint or not.

The returned likelihoods are scaled so that a test can be directly performed using the log of the likelihood ratio, $\lambda = \{null - alt\}$, which should be maximised.

In reality this function should not be used unless you are performing a changepoint test using the output supplied. This function is used in the "see also" functions that perform various changepoint tests, ideally these should be used.

Value

If data is a vector (single dataset) and extrainf=FALSE then a single value is returned:

cpt	The most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations

If data is a vector (single dataset) and extrainf=TRUE then a vector with three elements is returned:

cpt	The most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations
null	The scaled null likelihood (log likelihood of entire data with no change)
altlike	The scaled alternative liklihood at cpt (log likelihood of entire data with a change at cpt)

If data is an mxn matrix (multiple datasets) and extrainf=FALSE then a vector is returned:

cpt	Vector of length m containing the most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations for each row in data. cpt[1] is the most probable changepoint of the first row in data and cpt[m] is the most probable changepoint for the final row in data.

If data is a matrix (multiple datasets) and extrainf=TRUE then a matrix is returned where the first column is the changepoint location for each row in data, the second column is the scaled null likelihood for each row in data, the final column is the scaled maximum of the alternative likelihoods for each row in data.

Author(s)

Rebecca Killick

References

Change in Poisson Model: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

single.meanvar.poisson, cpt.meanvar

Examples

```
# Example of a change in mean and variance at 100 in simulated Poisson data
set.seed(10)
x=c(rpois(100,lambda=1),rpois(100,lambda=5))
single.meanvar.poisson.calc(x,extrainf=FALSE) # finds change at 99
single.meanvar.poisson.calc(x) # finds change at 99 and gives null likelihood as -1243.330 and
#alternative likelihood as -1580.522


# Example of no change in parameter (mean or variance) in simulated Poisson data
set.seed(10)
x=rpois(100,lambda=1)
single.meanvar.poisson.calc(x,extrainf=FALSE) # finds change at 17, this is the most probable point
#of change but if a changepoint test is performed then no change will be found.
single.meanvar.poisson.calc(x)# change at 17, null liklihood is 34.13681 and alternative is 31.20430
```

---

single.var.css *Single Change in Variance - Cumulative Sums of Squares*

Description

Calculates the cumulative sums of squares (css) test statistic for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.var.css(data,penalty="SIC",pen.value=0,class=TRUE,param.estimates=TRUE)
```

Arguments

data	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
penalty	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.

`pen.value` The theoretical type I error e.g.0.05 when using the Asymptotic penalty (options are 0.01,0.05,0.1,0.25,0.5,0.75,0.9,0.95). The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=test statistic, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.

`class` Logical. If TRUE then an object of class `cpt` is returned. If FALSE a vector is returned.

`param.estimates` Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find a single change in variance for data that is is not assumed to follow a specific distribtuion. The value returned is the result of testing H0:no change in variance against H1: single change in variance using the cumulative sums of squares test statistic coupled with the penalty supplied.

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are solely returned if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a single value is returned:

`cpt` The most probable location of a changepoint if H0 was rejected or NA if H1 was rejected.

If data is an mxn matrix (multiple datasets) then a vector is returned:

`cpt` Vector of length m containing where each element is the result of the test for data[m,]. If cpt[m] is a number then it is the most probable location of a changepoint under H1. Otherwise cpt[m] has the value NA and indicates that H1 was rejected.

Author(s)

Rebecca Killick

References

C. Inclan, G. C. Tiao (1994) Use of Cumulative Sums of Squares for Retrospective Detection of Changes of Variance, *Journal of the American Statistical Association* **89(427)**, 913–923

R. L. Brown, J. Durbin, J. M. Evans (1975) Techniques for Testing the Constancy of Regression Relationships over Time, *Journal of the Royal Statistical Society B* **32(2)**, 149–192

See Also

`cpt.var`,`cpt`

Examples

```
# Example of a change in variance at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,0,10))
single.var.css(x,penalty="Asymptotic",pen.value=0.05,class=FALSE) # returns 105 to show that the
#null hypothesis was rejected and the change in variance is at 105
ans=single.var.css(x,penalty="Asymptotic",pen.value=0.01)
cpts(ans) # returns 105 to show that the null hypothesis was rejected, the change in variance is at
#105 and we are 99% confident of this result

# Example of a data matrix containing 2 rows, row 1 has a change in variance and row 2 had no change
#in variance
set.seed(10)
x=c(rnorm(100,0,1),rnorm(100,0,10))
y=rnorm(200,0,1)
z=rbind(x,y)
single.var.css(z,penalty="Asymptotic",pen.value=0.05,class=FALSE) # returns vector c(102,200) which
#shows that the first dataset has a change in variance at 102 and the second dataset rejected H1 and
#has no change in variance
ans=single.var.css(z,penalty="Manual",pen.value=2)
cpts(ans[[1]]) # test using a manual penalty which is the same as the AIC penalty for this example
cpts(ans[[2]]) # result is the same as above, c(102, 200)
```

---

single.var.css.calc *Single Change in Variance - Cumulative Sums of Squares*

Description

Calculates the cumulative sums of squares (css) test statistic for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.var.css.calc(data, extrainf = TRUE)
```

Arguments

data	A vector or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
extrainf	Logical, if TRUE the test statistic is returned along with the changepoint location. If FALSE, only the changepoint location is returned.

Details

This function is used to find a single change in variance for data where no distributional assumption is made. The changepoint returned is simply the location where the test statistic is maximised, there is no test performed as to whether this location is a true changepoint or not.

In reality this function should not be used unless you are performing a changepoint test using the output supplied. This function is used in the "see also" functions that perform various changepoint tests, ideally these should be used.

Value

If data is a vector (single dataset) and extrainf=FALSE then a single value is returned:

`cpt` The most probable location of a changepoint

If data is a vector (single dataset) and extrainf=TRUE then a vector with two elements is returned:

`test statistic` The cumulative sums of squares test statistic

If data is an mxn matrix (multiple datasets) and extrainf=FALSE then a vector is returned:

`cpt` Vector of length m containing the most probable location of a changepoint for each row in data. cpt[1] is the most probable changepoint of the first row in data and cpt[m] is the most probable changepoint for the final row in data.

If data is a matrix (multiple datasets) and extrainf=TRUE then a matrix is returned where the first column is the changepoint location for each row in data, the second column is the test statistic for each row in data.

Author(s)

Rebecca Killick

References

C. Inclan, G. C. Tiao (1994) Use of Cumulative Sums of Squares for Retrospective Detection of Changes of Variance, *Journal of the American Statistical Association* **89(427)**, 913–923

R. L. Brown, J. Durbin, J. M. Evans (1975) Techniques for Testing the Constancy of Regression Relationships over Time, *Journal of the Royal Statistical Society B* **32(2)**, 149–192

See Also

`single.var.css`, `cpt.var`

Examples

```
# Example of a change in variance at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,0,10))
single.var.css.calc(x,extrainf=FALSE) # finds change at 105
single.var.css.calc(x) # finds change at 105 and gives test statistic as 4.979771


# Example of no change in variance in simulated normal data
set.seed(1)
x=rnorm(100,0,1)
single.var.css.calc(x,extrainf=FALSE) # finds change at 53, this is the most probable point of
```

```
#change but if a changepoint test is performed then no change will be found.
single.var.css.calc(x)# change at 53, test statistic is 0.6922931
```

`single.var.norm` *Single Change in Variance - Normal Data*

Description

Calculates the scaled log-likelihood (assuming the data is normally distributed) for all possible changepoint locations and returns the single most probable (max).

Usage

```
single.var.norm(data,penalty="SIC",pen.value=0,know.mean=FALSE,mu=NA,class=TRUE,
param.estimates=TRUE)
```

Arguments

`data`	A vector, ts object or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`penalty`	Choice of "None", "SIC", "BIC", "AIC", "Hannan-Quinn", "Asymptotic" and "Manual" penalties. If Manual is specified, the manual penalty is contained in the pen.value parameter. If Asymptotic is specified, the theoretical type I error is contained in the pen.value parameter. The predefined penalties listed do NOT count the changepoint as a parameter, postfix a 1 e.g."SIC1" to count the changepoint as a parameter.
`pen.value`	The theoretical type I error e.g.0.05 when using the Asymptotic penalty. The value of the penalty when using the Manual penalty option. This can be a numeric value or text giving the formula to use. Available variables are, n=length of original data, null=null likelihood, alt=alternative likelihood, tau=proposed changepoint, diffparam=difference in number of alternatve and null parameters.
`know.mean`	Logical, if TRUE then the mean is assumed known and mu is taken as its value. If FALSE, and mu=-1000 (default value) then the mean is estimated via maximum likelihood. If FALSE and the value of mu is supplied, mu is not estimated but is counted as an estimated parameter for decisions.
`mu`	Numerical value of the true mean of the data. Either single value or vector of length nrow(data). If data is a matrix and mu is a single value, the same mean is used for each row.
`class`	Logical. If TRUE then an object of class `cpt` is returned. If FALSE a vector is returned.
`param.estimates`	Logical. If TRUE and class=TRUE then parameter estimates are returned. If FALSE or class=FALSE no parameter estimates are returned.

Details

This function is used to find a single change in variance for data that is assumed to be normally distributed. The value returned is the result of testing H0:no change in variance against H1: single change in variance using the log of the likelihood ratio statistic coupled with the penalty supplied.

Value

If class=TRUE then an object of S4 class "cpt" is returned. The slot `cpts` contains the changepoints that are returned (with p-value) if class=FALSE. The structure of `cpts` is as follows.

If data is a vector (single dataset) then a single value is returned:

`cpt`	The most probable location of a changepoint if H0 was rejected or NA if H1 was rejected.

If data is an mxn matrix (multiple datasets) then a vector is returned:

`cpt`	Vector of length m containing where each element is the result of the test for data[m,]. If cpt[m] is a number then it is the most probable location of a changepoint under H1. Otherwise cpt[m] has the value NA and indicates that H1 was rejected.

Author(s)

Rebecca Killick

References

Change in variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

`cpt.var`,`single.meanvar.norm`,`single.mean.norm`,`cpt`

Examples

```
# Example of a change in variance at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,0,10))
single.var.norm(x,penalty="SIC",class=FALSE) # returns 100 to show that the null hypothesis was
#rejected and the change in variance is at 100
ans=single.var.norm(x,penalty="Asymptotic",pen.value=0.01)
cpts(ans) # returns 100 to show that the null hypothesis was rejected, the change in variance is at
#100 and we are 99% confident of this result

# Example of a data matrix containing 2 rows, row 1 has a change in variance and row 2 had no change
#in variance
set.seed(10)
x=c(rnorm(100,0,1),rnorm(100,0,10))
y=rnorm(200,0,1)
z=rbind(x,y)
```

```
single.var.norm(z,penalty="SIC",class=FALSE) # returns vector c(100,200) which shows that the first
#dataset has a change in variance at 100 and the second dataset rejected H1 and has no change in
#variance
ans=single.var.norm(z,penalty="Manual",pen.value="diffparam*log(n)")
cpts(ans[[1]]) # test using a manual penalty which is the same as the SIC penalty for this example
cpts(ans[[2]]) # result is the same as above, c(100, 200)
```

`single.var.norm.calc` *Single Change in Variance - Normal Data*

Description

Calculates the scaled negative log-likelihood (assuming the data is normally distributed) for all possible changepoint locations and returns the single most probable (min).

Usage

```
single.var.norm.calc(data, know.mean=FALSE, mu=NA, extrainf = TRUE)
```

Arguments

`data`	A vector or matrix containing the data within which you wish to find a changepoint. If data is a matrix, each row is considered a separate dataset.
`know.mean`	Logical, if TRUE then the mean is assumed known and mu is taken as its value. If FALSE, and mu=-1000 (default value) then the mean is estimated via maximum likelihood. If FALSE and the value of mu is supplied, mu is not estimated but is counted as an estimated parameter for decisions.
`mu`	Numerical value of the true mean of the data. Either single value or vector of length nrow(data). If data is a matrix and mu is a single value, the same mean is used for each row.
`extrainf`	Logical, if TRUE the scaled null and alternative negative likelihood values are returned along with the changepoint location. If FALSE, only the changepoint location is returned.

Details

This function is used to find a single change in variance for data that is assumed to be normally distributed. The changepoint returned is simply the location where the log likelihood ratio is maximised, there is no test performed as to whether this location is a true changepoint or not.

The returned negative log likelihoods are scaled so that a test can be directly performed using the log of the likelihood ratio, $\lambda = \{null - alt\}$, which should be maximised.

In reality this function should not be used unless you are performing a changepoint test using the output supplied. This function is used in the "see also" functions that perform various changepoint tests, ideally these should be used.

Value

If data is a vector (single dataset) and extrainf=FALSE then a single value is returned:

cpt	The most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations

If data is a vector (single dataset) and extrainf=TRUE then a vector with three elements is returned:

null	The scaled null likelihood (negative log likelihood of entire data with no change)
alt	The scaled alternative liklihood at cpt (negative log likelihood of entire data with a change at cpt)

If data is an mxn matrix (multiple datasets) and extrainf=FALSE then a vector is returned:

cpt	Vector of length m containing the most probable location of a changepoint (scaled max log likelihood over all possible changepoint locations for each row in data. cpt[1] is the most probable changepoint of the first row in data and cpt[m] is the most probable changepoint for the final row in data.

If data is a matrix (multiple datasets) and extrainf=TRUE then a matrix is returned where the first column is the changepoint location for each row in data, the second column is the scaled null likelihood for each row in data, the final column is the scaled maximum of the alternative likelihoods for each row in data.

Author(s)

Rebecca Killick

References

Change in variance: Chen, J. and Gupta, A. K. (2000) *Parametric statistical change point analysis*, Birkhauser

See Also

single.var.norm, cpt.var

Examples

```
# Example of a change in variance at 100 in simulated normal data
set.seed(1)
x=c(rnorm(100,0,1),rnorm(100,0,10))
single.var.norm.calc(x,extrainf=FALSE) # finds change at 100
single.var.norm.calc(x) # finds change at 100 and gives null likelihood as 765.1905 and alternative
#likelihood as 435.6529


# Example of no change in variance in simulated normal data
set.seed(1)
x=rnorm(100,0,1)
single.var.norm.calc(x,extrainf=FALSE) # finds change at 53, this is the most probable point of
#change but if a changepoint test is performed then no change will be found.
single.var.norm.calc(x)# change at 53, null liklihood is -22.47768 and alternative is -24.39894.
```

summary-methods *~~ Methods for Function summary in Package 'base' ~~*

Description

~~ Methods for function summary in Package 'base' ~~

Methods

signature(object = "ANY") Generic summary function, see base package description using ?summary

signature(object = "cpt") Prints out a summary of the object to the terminal.

signature(object = "cpt.reg") Prints out a summary of the object to the terminal.

---

test.stat *Generic Function - test.stat*

Description

Generic function

Usage

```
test.stat(object)
```

Arguments

object	Depending on the class of object depends on the method used (and if one exists)

Details

Generic Function

Value

Depends on the class of object, see individual methods

Author(s)

Rebecca Killick

See Also

test.stat-methods

Examples

```
x=new("cpt") # new cpt object
test.stat(x) # retrieves the test.stat slot from x
```

`test.stat-methods` *~~ Methods for Function test.stat ~~*

Description

~~ Methods for function `test.stat` ~~

Methods

`signature(object = "cpt")` Retrieves test.stat slot from an object of class cpt

`signature(object = "cpt.reg")` Retrieves test.stat slot from an object of class cpt.reg

`test.stat<-` *Generic Function - test.stat<-*

Description

Generic function

Usage

```
test.stat(object)<-value
```

Arguments

`object`	Depending on the class of `object` depends on the method used (and if one exists)
`value`	Replacement value

Details

Generic Function

Value

Depends on the class of `object`, see individual methods

Author(s)

Rebecca Killick

See Also

test.stat<--methods

Examples

```
x=new("cpt") # new cpt object
test.stat(x)<-"normal" # replaces the current test.stat slot of x with "normal"
```

---

test.stat<--methods *~~ Methods for Function test.stat<- ~~*

Description

~~ Methods for function test.stat<- ~~

Methods

signature(x = "cpt") Assigns the value following <- to the test.stat slot in x

signature(x = "cpt.reg") Assigns the value following <- to the test.stat slot in x

---

wave.c44137 *Wave data from buoy c44137*

Description

This dataset gives the significant wave heights from buoy c44137 obtained from the Fisheries and Oceans Canada, East Scotian Slop. The data are taken at hourly intervals from January 2005 until September 2012.

Usage

```
wave.c44137
```

Format

A vector of length 63651.

Source

http://www.meds-sdmm.dfo-mpo.gc.ca/isdm-gdsi/waves-vagues/search-recherche/list-liste/data-donnees-eng.asp?medsid=C44137

Index