Package ‘cpss’

November 23, 2020

Title  Change-Point Detection by Sample-Splitting Methods
Version  0.0.2
Description  Implements multiple change searching algorithms for a variety of
frequently considered parametric change-point models. In particular, it
integrates a criterion proposed by Zou, Wang and Li (2020)
<doi:10.1214/19-AOS1814> to select the number of change-points in a
data-driven fashion. Moreover, it also provides interfaces for
users-customized change-point models with their own cost function and
estimation routines.
License  GPL (>= 3)
Encoding  UTF-8
LazyData  true
RoxygenNote  7.1.1
LinkingTo  Rcpp, RcppArmadillo
Imports  Rcpp, magrittr, methods, stats, mvtnorm, Rfast, tibble, dplyr,
tidy, rlang, ggplot2, gridExtra
Suggests  MASS, L1pack
NeedsCompilation  yes
Author  Guanghui Wang [aut, cre],
        Changliang Zou [aut]
Maintainer  Guanghui Wang <ghwang.nk@gmail.com>
Repository  CRAN
Date/Publication  2020-11-23 09:40:15 UTC

R topics documented:

  coef.cpss-method .................................................. 2
  cpss-class ......................................................... 2
  cpss.custom ....................................................... 3
  cpss.em .......................................................... 6
  cpss.glm .......................................................... 8
Description

coef method

Usage

```r
## S4 method for signature 'cpss'
coef(object)
```

Arguments

- `object` (object)
- `cpss` (cpss)

Description

cpss: an S4 class which collects data and information required for further change-point analyses and summaries
Detecting changes in users-customized models

Usage

```r
cpss.custom(
  dataset,
  n,
  g_subdat,
  g_param,
  g_cost,
  algorithm = "BS",
  dist_min = floor(log(n)),
  ncps_max = ceiling(n^0.4),
  pelt_pen_val = NULL,
  pelt_K = 0,
  wbs_nintervals = 500,
  criterion = "CV",
  times = 2,
  model = NULL,
  g_smry = NULL,
  easy_cost = NULL,
  param.opt = NULL
)
```
Arguments

dataset: an ANY object that could be of any form such as a vector, matrix, tensor, list, etc.
n: an integer indicating the sample size of the dataset.
g_subdat: a customized R function of two arguments dat and indices, that returns a subset of the dat (inheriting the class from that of dataset) according to given indices along the observed time orders. The argument indices is a logical vector with TRUE indicating selected indices.
g_param: a customized R function of two arguments, dat and param.opt, that returns estimates of interested parameters that minimizes users-specified cost for a data set dat. The returned object could be of any class such as a numeric value, vector, matrix, list, etc. The argument param.opt might be used in the estimation procedures.
g_cost: a customized R function of two arguments, dat and param, that returns a numeric value of associated cost for a data set dat, under the knowledge of the interested parameters being param. The argument param inherits from the class of the returned object of the function g_param. If param.opt is needed to evaluate the cost, they should be packed into param when defining the function g_param.
algorithm: a character string specifying the change-point searching algorithm, one of four state-of-the-art candidates "SN" (segment neighborhood), "BS" (binary segmentation), "WBS" (wild binary segmentation) and "PELT" (pruned exact linear time) algorithms.
dist_min: an integer indicating the minimum distance between two successive candidate change-points, with a default value \( \text{floor}(\log(n)) \).
ncps_max: an integer indicating the maximum number of change-points searched for, with a default value \( \text{ceiling}(n^{0.4}) \).
pelt_pen_val: a numeric vector specifying the collection of candidate values of the penalty if the "PELT" algorithm is used.
pelt_K: a numeric value to adjust the pruning tactic, usually is taken to be 0 if negative log-likelihood is used as a cost; more details can be found in Killick et al. (2012).
wbs_nintervals: an integer indicating the number of random intervals drawn in the "WBS" algorithm and a default value 500 is used.
criterion: a character string indicating which model selection criterion, "cross-validation" ("CV") or "multiple-splitting" ("MS"), is used.
times: an integer indicating how many times of sample-splitting should be performed; if "CV" criterion is used, it should be set as 2.
model: a character string indicating the considered change model, and will be set as "custom" if not provided.
g_smry: a customized R function of two arguments dataset and param.opt, which calculates the summary statistics that will be needed in evaluations of the cost. The returned object is a list for convenience.
easy_cost: a customized R function of three arguments data_smry, s and e, that evaluates the cost for a date segment form observed time point $s$ to $e$. The argument data_smry inherits from the returned list of the function g_smry.
param.opt: an ANY object that could be of any form, specifying additional global constant parameters beyond the interested parameters.
Value

cpss.custom returns an object of an S4 class, called "cpss", which collects data and information required for further change-point analyses and summaries.

dat an ANY object inheriting form the type of user-input data
mdl a character string describing considered change-point model
algo a character string indicating user-specified change-point searching algorithm
algo_param_dim an integer indicating user-specified maximum number of change-points searched for if the algorithm is chosen among "SN", "BS" and "WBS", or a numeric vector collecting user-specified values for the penalty if the algorithm is "PELT"
SC a character string indicating model selection criterion
ncps an integer giving estimated number of change-points based on the entire data
pelt_pen a numeric value indicating selected penalty value if the "PELT" algorithm is performed based on the entire data
cps a numeric vector of detected change-points based on the entire data
params a list object, each of whose members is a list containing estimated parameters in the corresponding segment
S vals a numeric vector of candidate model dimensions in terms of a sequence of numbers of change-points or values of penalty
SC vals a numeric matrix, each column of which records the values of criterion based on the validation data under the corresponding model dimension (S vals), and each row of which represents a splitting at each time

References


Examples

library("cpss")
if (!requireNamespace("L1pack", quietly = TRUE)) {
  stop("Please install the package \"L1pack\".")
}
set.seed(666)
n <- 1000
tau <- c(250, 500, 750)
tau_ext <- c(0, tau, n)
be0 <- c(1, 1, 0, -1)
be <- c(1, -1, -1, 1)
seg_len <- diff(c(0, tau, n))
x <- rnorm(n)
eta <- unlist(lapply(seq(1, length(tau) + 1), function(k) {
  be0[k] + be[k] * x[(tau_ext[k] + 1):tau_ext[k + 1]]
}))
ep <- L1pack::rlaplace(n)
y <- eta + ep
g_subdat_l1 <- function(dat, indices) {
    matrix(dat[indices, ], sum(indices), ncol(dat))
}

g_param_l1 <- function(dat, param_opt = NULL) {
    y <- dat[, 1]
    x <- dat[, -1]
    return(L1pack::l1fit(x, y)$coefficients)
}

g_cost_l1 <- function(dat, param) {
    y <- dat[, 1]
    x <- dat[, -1]
    return(sum(abs(y - cbind(1, x) %*% as.matrix(param))))
}

res <- cpss.custom(
    dataset = cbind(y, x),
    n = n,
    g_subdat = g_subdat_l1,
    g_param = g_param_l1,
    g_cost = g_cost_l1,
    algorithm = "BS",
    dist_min = 10,
    ncps_max = 10,
    g_smry = NULL,
    easy_cost = NULL
)

summary(res)
# 250 500 744

do.call(rbind, res@params)
# Intercept  X
# [1,] 0.9327557 0.9558247
# [2,] 0.9868086 -1.0254999
# [3,] -0.0464067 -0.9076744
# [4,] -0.9746133 0.9671701

---

cpss.em

Detecting changes in exponential family

Description

Detecting changes in exponential family

Usage

cpss.em(
    dataset,
    family,
    size = NULL,
    algorithm = "BS",
    dist_min = floor(log(n)),
    ncps_max = ceiling(n^0.4),
    pelt_pen_val = NULL,
    pelt_K = 0,
    wbs_nintervals = 500,
    criterion = "CV",
    times = 2
)
Arguments

dataset  a numeric matrix of dimension $n \times d$, where each row represents an observation and each column stands for a variable. A numeric vector could also be acceptable for univariate observations.
family  a character string indicating the underlying distribution. Currently, detecting changes in binomial ("binom"), multinomial ("multinom"), Poisson ("pois"), exponential ("exp"), geometric ("geom"), dirichlet ("diri"), gamma ("gamma"), beta ("beta"), chi-square ("chisq") and inverse gaussian ("invgauss") distributions are supported.
size  an integer indicating the number of trials if family = "binom" or family = "multinom".
algorithm  a character string specifying the change-point searching algorithm, one of four state-of-the-art candidates "SN" (segment neighborhood), "BS" (binary segmentation), "WBS" (wild binary segmentation) and "PELT" (pruned exact linear time) algorithms.
dist_min  an integer indicating the minimum distance between two successive candidate change-points, with a default value $floor(log(n))$.
ncps_max  an integer indicating the maximum number of change-points searched for, with a default value $ceiling(n^{0.4})$.
pelt_pen_val  a numeric vector specifying the collection of candidate values of the penalty if the "PELT" algorithm is used.
pelt_K  a numeric value to adjust the pruning tactic, usually is taken to be 0 if negative log-likelihood is used as a cost; more details can be found in Killick et al. (2012).
wbs_nintervals  an integer indicating the number of random intervals drawn in the "WBS" algorithm and a default value 500 is used.
criterion  a character string indicating which model selection criterion, "cross-validation" ("CV") or "multiple-splitting" ("MS"), is used.
times  an integer indicating how many times of sample-splitting should be performed; if "CV" criterion is used, it should be set as 2.

Value

cpss.em returns an object of an S4 class, called "cpss", which collects data and information required for further change-point analyses and summaries. See cpss.custom.

References


See Also

cpss.meanvar cpss.mean cpss.var
Examples

```r
library("cpss")
set.seed(666)
n <- 1000
tau <- c(100, 300, 700, 900)
tau_ext <- c(0, tau, n)
theta <- c(1, 0.2, 1, 0.2, 1)
seg_len <- diff(c(0, tau, n))
y <- unlist(lapply(seq(1, length(tau) + 1), function(k) {
    rexp(seg_len[k], theta[k])
}))
res <- cpss.em(
    y, family = "exp", algorithm = "WBS",
    dist_min = 10, ncps_max = 10,
    criterion = "MS", times = 10
)
cps(res)
# [1] 100 299 705 901
```

cpss.glm

Detecting changes in GLMs

Description

Detecting changes in GLMs

Usage

```r
cpss.glm(
    formula,
    family,
    data = NULL,
    algorithm = "BS",
    dist_min = floor(log(n)),
    ncps_max = ceiling(n^0.4),
    pelt_pen_val = NULL,
    pelt_K = 0,
    wbs_nintervals = 500,
    criterion = "CV",
    times = 2
)
```

Arguments

- `formula` a formula object describing the change-point model to be fitted.
- `family` a description of the error distribution and link function to be used in the model, which can be a character string naming a family function or a family function.
data an optional data frame, list or environment containing the variables in the model.

algorithm a character string specifying the change-point searching algorithm, one of four state-of-the-art candidates "SN" (segment neighborhood), "BS" (binary segmentation), "WBS" (wild binary segmentation) and "PELT" (pruned exact linear time) algorithms.

dist_min an integer indicating the minimum distance between two successive candidate change-points, with a default value \( \text{floor}(\log(n)) \).

ncps_max an integer indicating the maximum number of change-points searched for, with a default value \( \text{ceiling}(n^{0.4}) \).

pelt_pen_val a numeric vector specifying the collection of candidate values of the penalty if the "PELT" algorithm is used.

pelt_K a numeric value to adjust the pruning tactic, usually is taken to be 0 if negative log-likelihood is used as a cost; more details can be found in Killick et al. (2012).

wbs_nintervals an integer indicating the number of random intervals drawn in the "WBS" algorithm and a default value 500 is used.

criterion a character string indicating which model selection criterion, "cross-validation" ("CV") or "multiple-splitting" ("MS"), is used.

times an integer indicating how many times of sample-splitting should be performed; if "CV" criterion is used, it should be set as 2.

Value

cpss.glm returns an object of an S4 class, called "cpss", which collects data and information required for further change-point analyses and summaries. See cpss.custom.

References


See Also

cpss.lm

Examples

library("cpss")
set.seed(666)
n <- 200
size <- rpois(n, 20 - 1) + 1
tau <- c(75, 100, 175)
tau_ext <- c(0, tau, n)
be <- list(c(0, 0.5), c(0, -0.5), c(0.5, -0.5), c(-0.5, -0.5))
seg_len <- diff(c(0, tau, n))
x <- rnorm(n)
etta <- lapply(seq(1, length(tau) + 1), function(k) {
  be[k][1] + be[k][2] * x[tau_ext[k] + 1:tau_ext[k + 1]]
})
eta <- do.call(c, eta)
p <- 1 / (1 + exp(-eta))
y <- rbinom(n, size = size, prob = p)

pelt_pen_val <- (log(n))^seq(0.5, 2, by = 0.1)
res <- cpss.glm(
  formula = cbind(y, size - y) ~ x, family = binomial(),
  algorithm = "PELT", pelt_pen_val = pelt_pen_val,
  dist_min = 5, ncps_max = 10
)

summary(res)
# 75 105 175
c coef(res)
# [1,]  0.02540872  0.08389551  0.5284425 -0.4980768
# [2,]  0.57222684 -0.45430385 -0.5203319 -0.4581678

cpss.lm

Detecting changes in linear models

Description

Detecting changes in linear models

Usage

```
cpss.lm(
  formula, 
  data = NULL, 
  algorithm = "BS", 
  dist_min = floor(log(n)), 
  ncps_max = ceiling(n^0.4), 
  pelt_pen_val = NULL, 
  pelt_K = 0, 
  wbs_nintervals = 500, 
  criterion = "CV", 
  times = 2
)
```

Arguments

- `formula` - a formula object describing the change-point model to be fitted.
- `data` - an optional data frame, list or environment containing the variables in the model.
- `algorithm` - a character string specifying the change-point searching algorithm, one of four state-of-the-art candidates "SN" (segment neighborhood), "BS" (binary segmentation), "WBS" (wild binary segmentation) and "PELT" (pruned exact linear time) algorithms.
dist_min an integer indicating the minimum distance between two successive candidate change-points, with a default value $\text{floor}(\log(n))$.

ncps_max an integer indicating the maximum number of change-points searched for, with a default value $\text{ceiling}(n^{0.4})$.

pelt_pen_val a numeric vector specifying the collection of candidate values of the penalty if the "PELT" algorithm is used.

pelt_K a numeric value to adjust the pruning tactic, usually is taken to be 0 if negative log-likelihood is used as a cost; more details can be found in Killick et al. (2012).

wbs_nintervals an integer indicating the number of random intervals drawn in the "WBS" algorithm and a default value 500 is used.

criterion a character string indicating which model selection criterion, "cross-validation" ("CV") or "multiple-splitting" ("MS"), is used.

times an integer indicating how many times of sample-splitting should be performed; if "CV" criterion is used, it should be set as 2.

Value
cpss.lm returns an object of an S4 class, called "cpss", which collects data and information required for further change-point analyses and summaries. See cpss.custom.

References

See Also
cpss.glm

Examples
library("cpss")
set.seed(666)
n <- 400
tau <- c(0, 200, 300)
tau_ext <- c(0, tau, n)
be <- list(c(0, 1), c(1, 0.5), c(0, 1), c(-1, 0.5))
seg_len <- diff(c(0, tau, n))
x <- rnorm(n)
mu <- lapply(seq(1, length(tau) + 1), function(k) {
  be[[k]][1] + be[[k]][2] * x[(tau_ext[k] + 1):tau_ext[k + 1]]
})
mu <- do.call(c, mu)
sig <- unlist(lapply(seq(1, length(tau) + 1), function(k) {
  rep(be[[k]][2], seg_len[k])
}))
y <- rnorm(n, mu, sig)
res <- cpss.lm(
  formula = y ~ x,
algorithm = "BS",
    dist_min = 5, ncps_max = 10
)
)
summary(res)
# 80 202 291
coef(res)
# $coef
# [1,] -0.00188792 1.0457718 -0.03963209 -0.9444813
# [2,] 0.91061557 0.6291965 1.20694409 0.4410036
#
# $sigma
# [1] 0.8732233 0.4753216 0.9566516 0.4782329

---

cpss.mean

*Detecting changes in mean*

**Description**

Detecting changes in mean

**Usage**

cpss.mean(
    dataset,
    algorithm = "BS",
    dist_min = floor(log(n)),
    ncps_max = ceiling(n^0.4),
    pelt_pen_val = NULL,
    pelt_K = 0,
    wbs_nintervals = 500,
    criterion = "CV",
    times = 2,
    Sigma = NULL
)

**Arguments**

dataset a numeric matrix of dimension $n \times d$, where each row represents an observation and each column stands for a variable. A numeric vector could also be acceptable for univariate observations.

algorithm a character string specifying the change-point searching algorithm, one of four state-of-the-art candidates "SN" (segment neighborhood), "BS" (binary segmentation), "WBS" (wild binary segmentation) and "PELT" (pruned exact linear time) algorithms.

dist_min an integer indicating the minimum distance between two successive candidate change-points, with a default value $floor(log(n))$. 
ncps_max  an integer indicating the maximum number of change-points searched for, with a default value $\text{ceiling}(n^{0.4})$.

pelt_pen_val  a numeric vector specifying the collection of candidate values of the penalty if the "PELT" algorithm is used.

pelt_K  a numeric value to adjust the pruning tactic, usually is taken to be 0 if negative log-likelihood is used as a cost; more details can be found in Killick et al. (2012).

wbs_nintervals  an integer indicating the number of random intervals drawn in the "WBS" algorithm and a default value 500 is used.

criterion  a character string indicating which model selection criterion, "cross-validation" ("CV") or "multiple-splitting" ("MS"), is used.

times  an integer indicating how many times of sample-splitting should be performed; if "CV" criterion is used, it should be set as 2.

Sigma  if a numeric matrix (or constant) is supplied, it would be taken as the value of known overall covariance (or variance). By default it is set as NULL, and the common covariance of the data is estimated based on the difference method, i.e.,

$$\hat{\Sigma} = \frac{1}{2(n-1)} \sum_{i=1}^{n-1} (Y_i - Y_{i+1})(Y_i - Y_{i+1})'$$

Value  

cpss.mean returns an object of an S4 class, called "cpss", which collects data and information required for further change-point analyses and summaries. See cpss.custom.

References  


See Also  

	cpss.meanvar cpss.var

Examples  

library("cpss")
set.seed(666)

n <- 2048
tau <- c(205, 267, 308, 472, 512, 820, 902, 1332, 1557, 1598, 1659)
seg_len <- diff(c(0, tau, n))
mu <- rep(c(0, 14.64, -3.66, 7.32, -7.32, 10.98, -4.39, 3.29, 19.03, 7.68, 15.37, 0), seg_len)
ep <- 7 * rnorm(n)
y <- mu + ep

res <- cpss.mean(y, algorithm = "SN", dist_min = 10, ncps_max = 20)
summary(res)
# 205 267 307 471 512 820 897 1332 1557 1601 1659
plot(res, type = "scatter")
`cpss.meanvar(res, type = "path")`
`out <- update(res, dim_update = 12)`
`out$ CPS_update
# 205 267 387 471 512 820 897 1332 1557 1601 1659 1769`
`out$ params_update

---

**cpss.meanvar**  
*Detecting changes in mean and (co)variance*

**Description**

Detecting changes in mean and (co)variance

**Usage**

```r
cpss.meanvar(
  dataset,
  algorithm = "BS",
  dist_min = floor(log(n)),
  ncps_max = ceiling(n^0.4),
  pelt_pen_val = NULL,
  pelt_K = 0,
  wbs_nintervals = 500,
  criterion = "CV",
  times = 2
)
```

**Arguments**

- **dataset**  
a numeric matrix of dimension $n \times d$, where each row represents an observation and each column stands for a variable. A numeric vector could also be acceptable for univariate observations.

- **algorithm**  
a character string specifying the change-point searching algorithm, one of four state-of-the-art candidates "SN" (segment neighborhood), "BS" (binary segmentation), "WBS" (wild binary segmentation) and "PELT" (pruned exact linear time) algorithms.

- **dist_min**  
an integer indicating the minimum distance between two successive candidate change-points, with a default value $floor(log(n))$.

- **ncps_max**  
an integer indicating the maximum number of change-points searched for, with a default value $ceiling(n^{0.4})$.

- **pelt_pen_val**  
a numeric vector specifying the collection of candidate values of the penalty if the "PELT" algorithm is used.

- **pelt_K**  
a numeric value to adjust the pruning tactic, usually is taken to be 0 if negative log-likelihood is used as a cost; more details can be found in Killick et al. (2012).
**cpss.var**

wbs_nintervals  an integer indicating the number of random intervals drawn in the "WBS" algorithm and a default value 500 is used.

criterion     a character string indicating which model selection criterion, "cross-validation" ("CV") or "multiple-splitting" ("MS"), is used.

times        an integer indicating how many times of sample-splitting should be performed; if "CV" criterion is used, it should be set as 2.

Value

cpss.meanvar returns an object of an **S4** class, called "cpss", which collects data and information required for further change-point analyses and summaries. See **cpss.custom**.

References


See Also

**cpss.mean** **cpss.var**

Examples

```r
library("cpss")
if (!requireNamespace("MASS", quietly = TRUE)) {
  stop("Please install the package \"MASS\"."")
}
set.seed(666)
n <- 1000
tau <- c(200, 400, 600, 800)
mu <- list(rep(0, 2), rep(1, 2), rep(0, 2), rep(0, 2))
Sigma <- list(diag(2), diag(2), matrix(c(1,-1,-1, 4), 2), matrix(c(1, 0.5, 0.5, 1), 2), diag(2))
seg_len <- diff(c(0, tau, n))
y <- lapply(seq(1, length(tau) + 1), function(k) { MASS::mvrnorm(n = seg_len[k], mu = mu[[k]], Sigma = Sigma[[k]]) })
y <- do.call(rbind, y)
res <- cpss.meanvar(y, algorithm = "BS", dist_min = 20)
cps(res)
# [1]  211  402  598  804
plot(res, type = "coef")
```

**Description**

Detecting changes in (co)variance
Usage

cpss.var(
    dataset,
    algorithm = "BS",
    dist_min = floor(log(n)),
    ncps_max = ceiling(n^0.4),
    pelt_pen_val = NULL,
    pelt_K = 0,
    wbs_nintervals = 500,
    criterion = "CV",
    times = 2,
    mu = NULL
)

Arguments

dataset  a numeric matrix of dimension $n \times d$, where each row represents an observation and each column stands for a variable. A numeric vector could also be acceptable for univariate observations.

algorithm a character string specifying the change-point searching algorithm, one of four state-of-the-art candidates "SN" (segment neighborhood), "BS" (binary segmentation), "WBS" (wild binary segmentation) and "PELT" (pruned exact linear time) algorithms.

dist_min  an integer indicating the minimum distance between two successive candidate change-points, with a default value $\text{floor}(\log(n))$.

ncps_max  an integer indicating the maximum number of change-points searched for, with a default value $\text{ceiling}(n^{0.4})$.

pelt_pen_val a numeric vector specifying the collection of candidate values of the penalty if the "PELT" algorithm is used.

pelt_K  a numeric value to adjust the pruning tactic, usually is taken to be 0 if negative log-likelihood is used as a cost; more details can be found in Killick et al. (2012).

wbs_nintervals an integer indicating the number of random intervals drawn in the "WBS" algorithm and a default value 500 is used.

criterion  a character string indicating which model selection criterion, "cross-validation" ("CV") or "multiple-splitting" ("MS"), is used.

times  an integer indicating how many times of sample-splitting should be performed; if "CV" criterion is used, it should be set as 2.

mu  if a numeric vector or constant is supplied, it would be taken as the value of known overall mean. By default it is set as NULL, and the common mean of the data is estimated by the sample mean based on the entire data set.

Value

cpss.var returns an object of an S4 class, called "cpss", which collects data and information required for further change-point analyses and summaries. See cpss.custom.
References


See Also

cpss.meanvar cpss.mean

Examples

library("cpss")
if (!requireNamespace("MASS", quietly = TRUE)) {
  stop("Please install the package \"MASS\".")
}
set.seed(666)
N <- 1000
tau <- c(200, 500, 750)
mu <- list(rep(0, 2), rep(0, 2), rep(0, 2), rep(0, 2))
Sigma <- list(diag(2), matrix(c(1, 0, 0, 4), 2), matrix(c(1, -0.5, -0.5, 4), 2), diag(2))
seg_len <- diff(c(0, tau, N))
y <- lapply(seq(1, length(tau) + 1), function(k) {
  MASS::mvrnorm(N = seg_len[k], mu = mu[[k]], Sigma = Sigma[[k]])
})
y <- do.call(rbind, y)
res <- cpss.var(y, algorithm = "BS", dist_min = 20)
cps(res)
# [1] 215 515 751

plot,cpss-method  plot method

Description

plot method

Usage

## S4 method for signature 'cpss'
plot(obj, type, x = c(), y = c(), ...)

Arguments

obj obj
type type
x x
y y
... ...
cpss cpss
## Summary Method

### Description

**summary method**

### Usage

```r
## S4 method for signature 'cpss'
summary(object)
```

### Arguments

- `object`: object
- `cpss`: object

---

**tool1**

### Description

**tool1**

### Usage

```r
dat(x)
mdl(x)
algo(x)
algo_param_dim(x)
SC(x)
ncps(x)
pelt_pen(x)
cps(x)
params(x)
S_vals(x)
```
SC_vals(x)
upcalle.inputs(x)

Arguments

\[
\begin{array}{ll}
\text{tool}2 & \text{tool}2 \\
\end{array}
\]

Description

tool2

Usage

dat(x) <- value
mdl(x) <- value
algo(x) <- value
algo_param_dim(x) <- value
SC(x) <- value
ncps(x) <- value
pelt_pen(x) <- value
cps(x) <- value
params(x) <- value
S_vals(x) <- value
SC_vals(x) <- value
upcalle.inputs(x) <- value

Arguments

\[
\begin{array}{ll}
x & x \\
value & value \\
\end{array}
\]
Description

tool3

Usage

```r
## S4 method for signature 'cpss'
dat(x)

## S4 method for signature 'cpss'
mdl(x)

## S4 method for signature 'cpss'
algo(x)

## S4 method for signature 'cpss'
algo_param_dim(x)

## S4 method for signature 'cpss'
SC(x)

## S4 method for signature 'cpss'
cmps(x)

## S4 method for signature 'cpss'
pelt_pen(x)

## S4 method for signature 'cpss'
cps(x)

## S4 method for signature 'cpss'
params(x)

## S4 method for signature 'cpss'
S_vals(x)

## S4 method for signature 'cpss'
SC_vals(x)

## S4 method for signature 'cpss'
upcalle.inputs(x)
```

Arguments

x x
Description

tool4

Usage

```r
## S4 replacement method for signature 'cpss'
dat(x) <- value

## S4 replacement method for signature 'cpss'
mdl(x) <- value

## S4 replacement method for signature 'cpss'
algo(x) <- value

## S4 replacement method for signature 'cpss'
algo_param_dim(x) <- value

## S4 replacement method for signature 'cpss'
SC(x) <- value

## S4 replacement method for signature 'cpss'
cmps(x) <- value

## S4 replacement method for signature 'cpss'
pelt_pen(x) <- value

## S4 replacement method for signature 'cpss'
cps(x) <- value

## S4 replacement method for signature 'cpss'
params(x) <- value

## S4 replacement method for signature 'cpss'
S_vals(x) <- value

## S4 replacement method for signature 'cpss'
SC_vals(x) <- value

## S4 replacement method for signature 'cpss'
upcall.e.inputs(x) <- value
```
Arguments

- x  x
- value  value
- cpss  cpss

Description

update method

Usage

```r
## S4 method for signature 'cpss'
update(object, dim_update)
```

Arguments

- object  object
- dim_update  dim_update
- cpss  cpss
# Index

|algo (tool1), 18|  
|algo, cpss-method (tool3), 20|  
|algo< (tool2), 19|  
|algo<, cpss-method (tool4), 21|  
|algo_param_dim (tool1), 18|  
|algo_param_dim, cpss-method (tool3), 20|  
|algo_param_dim< (tool2), 19|  
|algo_param_dim<, cpss-method (tool4), 21|  
|coef, cpss-method, 2|  
|cps (tool1), 18|  
|cps, cpss-method (tool3), 20|  
|cps< (tool2), 19|  
|cps<, cpss-method (tool4), 21|  
|cpss-class, 2|  
|cpss.custom, 3, 7, 9, 11, 13, 15, 16|  
|cpss.em, 6|  
|cpss.glm, 8, 11|  
|cpss.lm, 9, 10|  
|cpss.mean, 7, 12, 15, 17|  
|cpss.meanvar, 7, 13, 14, 17|  
|cpss.var, 7, 13, 15, 15|  
|dat (tool1), 18|  
|dat, cpss-method (tool3), 20|  
|dat< (tool2), 19|  
|dat<, cpss-method (tool4), 21|  
|mdl (tool1), 18|  
|mdl, cpss-method (tool3), 20|  
|mdl< (tool2), 19|  
|mdl<, cpss-method (tool4), 21|  
|ncps (tool1), 18|  
|ncps, cpss-method (tool3), 20|  
|ncps< (tool2), 19|  
|ncps<, cpss-method (tool4), 21|  
|params (tool1), 18|  
|params, cpss-method (tool3), 20|  
|params< (tool2), 19|  
|params<, cpss-method (tool4), 21|  
|pelt_pen (tool1), 18|  
|pelt_pen, cpss-method (tool3), 20|  
|pelt_pen< (tool2), 19|  
|pelt_pen<, cpss-method (tool4), 21|  
|plot, cpss-method, 17|  
|S_vals (tool1), 18|  
|S_vals, cpss-method (tool3), 20|  
|S_vals< (tool2), 19|  
|S_vals<, cpss-method (tool4), 21|  
|SC (tool1), 18|  
|SC, cpss-method (tool3), 20|  
|SC< (tool2), 19|  
|SC<, cpss-method (tool4), 21|  
|SC_vals (tool1), 18|  
|SC_vals, cpss-method (tool3), 20|  
|SC_vals< (tool2), 19|  
|SC_vals<, cpss-method (tool4), 21|  
|summary, cpss-method, 18|  
|tool1, 18|  
|tool2, 19|  
|tool3, 20|  
|tool4, 21|  
|upcall.inputs (tool1), 18|  
|upcall.inputs, cpss-method (tool3), 20|  
|upcall.inputs< (tool2), 19|  
|upcall.inputs<, cpss-method (tool4), 21|  
|update, cpss-method, 22|