Package ‘PCpluS’

July 21, 2025

Title Piecewise Constant Plus Smooth Regression

Version 1.0.1

Depends R (>= 3.0.2)

Imports Rcpp (>= 0.12.3), changepoint, methods, Matrix

LinkingTo Rcpp (>= 0.12.3), RcppEigen

Suggests testthat (>= 1.0.0), glmnet

Description Allows for nonparametric regression where one assumes that the signal is given by the sum of a piecewise constant function and a smooth function. More precisely, it implements the estimator PCpluS (piecewise constant plus smooth regression estimator) from Pein and Shah (2025) <doi:10.48550/arXiv.2112.03878>.

License GPL-3

NeedsCompilation yes

Author Pein Florian [aut, cre]

Maintainer Pein Florian <`f.pein@lancaster.ac.uk`>

Repository CRAN

Date/Publication 2025-03-11 16:40:06 UTC

Contents

`cvpcplus` *Tuning parameter selection by crossvalidation*

Description

Selects the tuning parameters, bandwidth and the penalty lambda, of the PCPLUS estimator *F. Pein (2021).* The values obtained can be used in the estimator `pcplus`.

Usage

```
cv.pcplus(y, bandwidth, lambda, nbandwidth = 30L, nlambda = 30L,
          lambda.min.ratio = 0.01, sd = NULL, thresh = 1e-7, maxit = 1e5L)
```

Arguments

`y` a numeric vector containing the observations, only finite values are allowed

`bandwidth` a numeric vector specifying possible values for the bandwidth of the kernel smoother; each entry must be between `2 / length(n)` and `0.25` or `Inf`, smaller values are replaced by `2 / n` and larger by `Inf` with a warning; see *F. Pein (2021)* for an interpretation of `bandwidth == Inf`. If missing an exponential grid of length `nbandwidth` will be used

`lambda` a decreasing sequence of numerics specifying possible values for the penalty penalty of the fused lasso; each value must be positive. If missing an exponential grid of length `nlambda` is used

`nbandwidth` a single integer giving the length of the grid for `bandwidth`; ignored if `bandwidth` is given

`nlambda` a single integer giving the length of the grid for `lambda`; ignored if `lambda` is given

`lambda.min.ratio` a single numeric between `0` and `1` speciyfing the range of the grid for `lambda`; ignored if `lambda` is given. More precisely, for each `bandwdith` value the largest value of the grid is chosen such that no changes are found and the smallest value is the largest value times `lambda.min.ratio`

`sd` a single positive value giving the standard deviation of the observations; may be `NULL`, in which case a robust estimator is used

`thresh` a single positive numeric value giving a convergence threshold for coordinate descent. Each inner coordinate-descent loop continues until the maximum change in the objective after any coefficient update is less than `thresh` times the null deviance

`maxit` a single positive integer giving the maximum number of passes over the data for all lambda values

Value

a `list` containing the entries `lambda` and `bandwidth` giving the best parameter for the tuning parameters. Both can be passed directly to `pcplus`. Note that `lambda` is a decaying sequence instead of a single value. This improves the runtime of the estimator. The last value is the suggested tuning parameter. Furthermoore, it has the entries `cv` with the loss for the selected parameters, `bandwidths` with the grid of bandwidths used, and `cvs` with the loss for all bandwidths.

References

Pein, F. (2021). Change-point regression with a smooth additive disturbance. *arXiv preprint* arXiv:2112.03878.

See Also

`pcplus`

Examples

```
library(PCpluS)
set.seed(1)
y <- c(rnorm(125), rnorm(125, 3)) + sin(2 * pi * 1:250 / 250)

CV <- cv.pcplus(y)
ret <- pcplus(y, lambda = CV$lambda, bandwidth = CV$bandwidth)

plot(y, pch = 16)
lines(ret$est, col = "red")
abline(v = ret$cps)
```

`pcplus` *Piecewise constant plus smooth estimation*

Description

Computes the PCPLUS estimator *F. Pein (2021)* for a given bandwidth and the penalty lambda. Bandwidth and lambda can be obtained by cross-validation using the function `cv.pcplus`. The PCPLUS estimator returns a piecewise constant function plus a smooth function as well as the change-points of the piecewise constant function.

Usage

```
pcplus(y, bandwidth, lambda, sd = NULL, nlambda = 30L, thresh = 1e-7, maxit = 1e5L)
```

Arguments

`y` a numeric vector containing the observations, only finite values are allowed

`bandwidth` a single positive value specifying the bandwidth for the kernel smoother; must be between `1 / length(n)` and `0.25` or `Inf`, smaller values are replaced by `1 / n` and larger by `Inf` with a warning; see *F. Pein (2021)* for an interpretation of `bandwidth == Inf`

`lambda` a single positive numeric or a decreasing sequence of positive numeric values giving the penalty for the fused lasso. If a sequence is passed, then only the smallest value is used to compute the estimator. However, passing a sequence of lambda values is often much faster than passing a single value

`sd` a single positive value giving the standard deviation of the observations; may be `NULL`, in which case a robust estimator is used

`nlambda` a single positive integer specifying the number of lambda values to pass to the lasso; only used if `lambda` is a single value, in which case an exponentially decreasing sequence is generated, with `lambda` being the smallest value. As explained for `lambda`, only this value is used for the estimator, but adding other values may reduce the run time `thresh` a single positive numeric value giving a convergence threshold for coordinate descent. Each inner coordinate-descent loop continues until the maximum change in the objective after any coefficient update is less than `thresh` times the null deviance

`maxit` a single positive integer giving the maximum number of passes over the data for all lambda values

Value

a `list` containing the entries `est` with the fitted values of the esitmator, `smooth` with the smooth part of the estimator, `cpfun` with the change-point part of the estimator, and `cps` with the estimated change-point locations.

References

Pein, F. (2021). Change-point regression with a smooth additive disturbance. *arXiv preprint* arXiv:2112.03878.

See Also

`cv.pcplus`

Examples

```
library(PCpluS)
set.seed(1)
y <- c(rnorm(125), rnorm(125, 3)) + sin(2 * pi * 1:250 / 250)

CV <- cv.pcplus(y)
ret <- pcplus(y, lambda = CV$lambda, bandwidth = CV$bandwidth)

plot(y, pch = 16)
lines(ret$est, col = "red")
abline(v = ret$cps)
```

Index