Package 'ChangePointInference'

July 6, 2023

Title Inference for Change Points in Piecewise Polynomials via Differencing

Version 0.0.0.9000

Description R implementation of the procedure introduced in the paper
``Fast and Optimal Inference for Change Points in Piecewise Polynomials via Differencing".
Given a vector of observations from a one dimensional signal + noise model,
where the signal is a piecewise polynomial function of known degree,
the procedure returns disjoint intervals which must each contain a change point location
uniformly at some level specified by the parameter `alpha`.

License GPL-3License: GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

R topics documented:

`cpt` *Estimate Change Point Locations*

Description

Estimates the most likely change point location within each of the intervals obtained by calling `diffInf`.

Usage

```
cpt(obj, cpt_loc = "RSS")
```

Arguments

obj	An object of class cptInference returned by diffInf.
cpt_loc	Determines how the change point locations will be estimated. Should be one of "midpoint" or "RSS". If cpt_loc == "midpoint" the change point locations are simply taken to be the midpoints of each interval. If cpt_loc == "RSS" the change point locations are chosen to be the split points within each interval associated with the piecewsie polynomial fit providing the lowest sum of squared residuals.

Value

A vector of estimated change point locations.

Examples

```
# Piecewise linear mean with i.i.d. Gaussian noise
set.seed(42)

waves_signal <- c((1:150) * (2**-3), (150:1) * (2**-3), (1:150) * (2**-3), (150:1) * (2**

yy <- waves_signal + rnorm(length(waves_signal), sd = 5)


#' Recover intervals of significance

 diffInf_obj <- diffInf(yy, degree = 1)

 diffInf_obj

#' plot the intervals of significance

 diffInf_obj |> plot(type = "l", col = "grey")

 waves_signal |> lines(lty = 2, lwd = 2)


#' recover likely change point locations

cpt(diffInf_obj, "RSS")

cpt(diffInf_obj, "midpoint")

abline(v = cpt(diffInf_obj, "RSS"), col = "red", lty = 2)

abline(v = cpt(diffInf_obj, "midpoint"), col = "blue", lty = 3)
```

Description

Identifies sub-intervals of the data sequence which each must contain a change point, in the sense that the mean function cannot be described as a polynomial of degree p, uniformly with probability asymptotically larger than $1 - \alpha + o(1)$ where $\alpha \in (0, 1)$ can be set by adjusting the parameter `alpha`. The object returned by this function can be passed to:

- `plot`: for potting the data along with the intervals of significance returned.
- `cpt`: for estimating the most probable change point location within each interval recovered.
- `predict`: for estimating the unobserved piecewise polynomial mean function, using the most likely change point locations recovered by `cpt`.

Usage

```
diffInf(
  yy,
  degree,
  alpha = 0.1,
  gaussian_noise = TRUE,
  independent_noise = TRUE,
  tau = NULL,
  aa = sqrt(2),
  min_scale = floor(sqrt(length(yy))/2),
  HH = NULL
)
```

Arguments

`yy`	A numeric vector containing the data to be inspected for change points.
`degree`	The degree of polynomial parametrization of the mean of the data.
`alpha`	Desired maximum probability of obtaining an interval that does not contain a change point.
`gaussian_noise`	Set to `TRUE` if the contaminating noise noise is assumed to be independently distributed and Gaussian, with common variance, else set to `FALSE`.
`independent_noise`	Set to `TRUE` if the contaminating noise is assumed to be independently distributed, else set to `FALSE`.
`tau`	Noise level in as measured by the (long run) standard deviation of the noise, if know. If `tau=NULL` the noise level will be estimated via: • mad_diff if `gaussian_noise` = `TRUE` and `independent_noise` = `TRUE`. • sd_diff if `gaussian_noise` = `FALSE` and `independent_noise` = `TRUE`. • lrsd_block_diff if `independent_noise` = `FALSE`.
`aa`	Decay parameter controlling the density of the grid on which local tests for the presence of a change point will be performed. Tests will be performed on all contiguous sub-intervals of $\{1, \ldots, n\}$ whose length is larger than `min_scale` and can be expressed as an integer power of `aa`.
`min_scale`	Minimum scale at which local tests for the presence of a change point will be performed.

HH — Numeric constant appearing in the extreme value limit of the supremum over all local tests, under the null of no change points. If pre-computed the value can be passed to the function. Otherwise, the value is computed automatically using.

Details

The data supplied in yy are assumed to follow the 'signal + noise' model:

$$Y_t = f_\circ(t/n) + \zeta_t \qquad t = 1, \ldots, n.$$

Where n is the length of the data sequence, $f_\circ(t/n)$ is a piecewise polynomial function of know degree p, and the noise terms are one of:

1. Independently distributed and Gaussian, with common variance.
2. Independently distributed with common variance, but not necessarily Gaussian.
3. Weakly stationary with strictly positive long run variance.

Value

An object of class "list" and "not", which contains the following fields:

- intervals: intervals of significance returned by the procedure.
- thresh: the threshold used to for each local test.
- data: the original input data yy.
- degree: degree of polynomial parametrization of the mean of the data.

Examples

```
## Piecewise constant mean with i.i.d. Gaussian noise

set.seed(42)

blocks_signal <- c(rep(0,205),rep(14.64,62),rep(-3.66,41),rep(7.32,164),rep(-7.32,40))

yy <- blocks_signal + rnorm(length(blocks_signal), sd = 5)


## Recover intervals of significance

diffInf_obj <- diffInf(yy, degree = 0)

diffInf_obj


## some examples of how the `diffInf` object can be used

# plot the intervals of significance

diffInf_obj |> plot(type = "l", col = "grey")

blocks_signal |> lines(lty = 2, lwd = 2)


# recover likely change point locations
```

```
cpt(diffInf_obj)

abline(v = cpt(diffInf_obj), col = "red", lty = 2)


# estimate the best piecewise polynomial fit based on the change point locations

predict(diffInf_obj)

lines(predict(diffInf_obj), col = "red")
```

`lrsd_block_diff` *(Long Run) Standard Deviation Estimator Based on* $(p+1)$*-th Differences of Local Sums of the Data*

Description

Estimates the (long run) standard deviation based on $(p+1)$-th differences of non-overlapping local sums of the data.

Usage

```
lrsd_block_diff(
  yy,
  ww = length(yy)^{
     (1/3)
 },
  degree = 0
)
```

Arguments

`yy`	A numeric vector containing the data.
`ww`	The scale at which local sums of the data will be calculated.
`degree`	The degree of polynomial parametrization of the mean of the data.

Examples

```
degree <- 3

nn <- 500

yy <- arima.sim(model = list(ar = 0.5), n = nn) + (1:nn)^{degree}

sd_diff(yy, degree)
```

mad_diff *Median Absolute Deviation Estimator Using $(p+1)$-th Differences*

Description

Estimates the standard deviation using the median of the $(p+1)$-th difference of the data. Suitable for estimating the scale of the noise when the data sequence is Gaussian with a piecewise polynomial mean of degree p.

Usage

```
mad_diff(yy, degree)
```

Arguments

yy	A numeric vector containing the data.
degree	The degree of polynomial parametrization of the mean of the data.

Examples

```
degree <- 3

nn <- 500

yy <- rnorm(nn) + (1:nn)^{degree}

mad_diff(yy, degree)
```

plot.cptInference *Plot a 'cptInference' Object*

Description

Plots the data and as well as intervals of significance returned by the diffInf.

Usage

```
## S3 method for class 'cptInference'
plot(obj, cpt_loc_est = FALSE, ...)
```

Arguments

obj	An object of class 'cptInference'.
cpt_loc_est	One of "RSS", "midpoint", or FALSE. If "RSS" or "midpoint" the mosyt likely change pint location within each interval will be plotted. If FALSE only the intervals are plotted.
...	Additional graphical parameters passed to plot.

Examples

```
# Piecewise linear mean with i.i.d. Gaussian noise

set.seed(42)

waves_signal <- c((1:150) * (2**-3), (150:1) * (2**-3), (1:150) * (2**-3), (150:1) * (2**

yy <- waves_signal + rnorm(length(waves_signal), sd = 5)


# Recover intervals of significance

diffInf_obj <- diffInf(yy, degree = 1)

diffInf_obj

# plot the intervals of significance

diffInf_obj |> plot("RSS", type = "l", col = "grey")

# plot intervals and minimum RSS split points

diffInf_obj |> plot("RSS", type = "l", col = "grey")

# plot the intervals and their midpoints

diffInf_obj |> plot("midpoint", type = "l", col = "grey")
```

`predict.cptInference`
Predict a 'cptInference' Object

Description

Fits a piecewise polynomial signal to the to the data, using change point locations recovered by `cpt`

Usage

```
## S3 method for class 'cptInference'
predict(obj, cpt_loc = "RSS")
```

Arguments

`obj`	An object of class 'cptInference'.
`cpt_loc`	Determines how the change points will be estimated. If `cpt_loc = "RSS"` the change points are taken to be the split points within each interval which provide the lowest RSS when a piecewise polynomial function is fitted on the same interval. If `cpt_loc = "midpoint"` the change points are taken to be the midpoint of each interval.

Examples

```
#' Piecewise linear mean with i.i.d. Gaussian noise

set.seed(42)

waves_signal <- c((1:150) * (2**-3), (150:1) * (2**-3), (1:150) * (2**-3), (150:1) * (2**

yy <- waves_signal + rnorm(length(waves_signal), sd = 5)


#' Recover intervals of significance

diffInf_obj <- diffInf(yy, degree = 1)

diffInf_obj

#' plot the intervals of significance

diffInf_obj |> plot(type = "l", col = "grey")

waves_signal |> lines(lty = 2, lwd = 2)


#' recover fitted signal

lines(predict(diffInf_obj, "RSS"), col = "red", lty = 2, lwd = 2)

lines(predict(diffInf_obj, "midpoint"), col = "blue", lty = 3, lwd = 2)
```

---

sd_diff *Standard Deviation Estimator Based On $(p+1)$-th Differences*

Description

Estimates the standard deviation of the data using the mean on the squared $(p+1)$-th difference of the data sequence. Suitable for estimating the scale of the noise when the data sequence consists of a piecewise polynomial function contaminated with independently distributed noise having bounded fourth moment.

Usage

```
sd_diff(yy, degree)
```

Arguments

yy	A numeric vector containing the data.
degree	The degree of polynomial parametrization of the mean of the data.

Examples

```
degree <- 3

nn <- 500

yy <- rt(nn, df = 5) * sqrt(3/5) + (1:nn)^{degree}

sd_diff(yy, degree)
```

Index