Package 'nsp'

July 22, 2025

Title Inference for Multiple Change-Points in Linear Models

Version 1.0.0

Description Implementation of Narrowest Significance Pursuit, a general and flexible methodology for automatically detecting localised regions in data sequences which each must contain a change-point (understood as an abrupt change in the parameters of an underlying linear model), at a prescribed global significance level. Narrowest Significance Pursuit works with a wide range of distributional assumptions on the errors, and yields exact desired finite-sample coverage probabilities, regardless of the form or number of the covariates. For details, see P. Fryzlewicz (2021) <https://stats.lse.ac.uk/fryzlewicz/nsp/nsp.pdf>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.1.2

Depends R (>= 3.0.0)

Imports lpSolve

NeedsCompilation no

Author Piotr Fryzlewicz [aut, cre] (ORCID: <https://orcid.org/0000-0002-9676-902X>)

Maintainer Piotr Fryzlewicz <p.fryzlewicz@lse.ac.uk>

Repository CRAN

Date/Publication 2021-12-21 07:10:06 UTC

Contents

`cov_dep_multi_norm` *Simulate covariate-dependent multiscale sup-norm for use in NSP*

Description

This function simulates the multiscale sup-norm adjusted for the form of the covariates, as described in Section 5.3 of the paper. This is done for i.i.d. N(0,1) innovations.

Usage

```
cov_dep_multi_norm(x, N = 1000)
```

Arguments

`x`	The design matrix with the regressors (covariates) as columns.
`N`	Desired number of simulated values of the norm.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint.

Value

Sample of size `N` containing the simulated norms.

Author(s)

Piotr Fryzlewicz, <`p.fryzlewicz@lse.ac.uk`>

See Also

`cov_dep_multi_norm_poly`, `sim_max_holder`

Examples

```
set.seed(1)
g <- c(rep(0, 100), rep(2, 100))
x.g <- g + stats::rnorm(200)
mscale.norm.200 <- cov_dep_multi_norm(matrix(1, 200, 1), 100)
nsp_poly(x.g, 100, thresh.val = stats::quantile(mscale.norm.200, .95))
```

cov_dep_multi_norm_poly — *Simulate covariate-dependent multiscale sup-norm for use in NSP, for piecewise-polynomial models*

Description

This function simulates the multiscale sup-norm adjusted for the form of the covariates, as described in Section 5.3 of the paper, for piecewise-polynomial models of degree deg. This is done for i.i.d. N(0,1) innovations.

Usage

```
cov_dep_multi_norm_poly(n, deg, N = 10000)
```

Arguments

n	The data length (for which the multiscale norm is to be simulated)
deg	The degree of the polynomial model (0 for the piecewise-constant model; 1 for piecewise-linearity, etc.).
N	Desired number of simulated values of the norm.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint.

Value

Sample of size N containing the simulated norms.

Author(s)

Piotr Fryzlewicz, <p.fryzlewicz@lse.ac.uk>

See Also

cov_dep_multi_norm, sim_max_holder

Examples

```
set.seed(1)
g <- c(rep(0, 100), rep(2, 100))
x.g <- g + stats::rnorm(200)
mscale.norm.200 <- cov_dep_multi_norm_poly(200, 0, 100)
nsp_poly(x.g, 100, thresh.val = stats::quantile(mscale.norm.200, .95))
```

cpt_importance *Change-point importance (prominence) plot*

Description

This function produces a change-point prominence plot based on the NSP object provided. The heights of the bars are arranged in non-decreasing order and correspond directly to the lengths of the NSP intervals of significance. Each bar is labelled as s-e where s (e) is the start (end) of the corresponding NSP interval of significance, respectively. The change-points corresponding to the narrower intervals can be seen as more prominent.

Usage

```
cpt_importance(nsp.obj)
```

Arguments

`nsp.obj`	Object returned by one of the `nsp*` functions.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint.

Value

The function does not return a value.

Author(s)

Piotr Fryzlewicz, <`p.fryzlewicz@lse.ac.uk`>

See Also

`draw_rects`, `draw_rects_advanced`

Examples

```
set.seed(1)
f <- c(rep(0, 100), 1:100, rep(101, 100))
x.f <- f + 15 * stats::rnorm(300)
x.f.n <- nsp_poly(x.f, 100, "sim", deg=1)
cpt_importance(x.f.n)
```

draw_rects	*Draw NSP intervals of significance as shaded rectangular areas on the current plot*

Description

This function draws intervals of significance returned by one of the `nsp*` functions on the current plot. It shows them as shaded rectangular areas (hence the name of the function).

Usage

```
draw_rects(nsp.obj, yrange, density = 10, col = "red", x.axis.start = 1)
```

Arguments

`nsp.obj`	Object returned by one of the `nsp*` functions.
`yrange`	Vector of length two specifying the (lower, upper) vertical limit of the rectangles.
`density`	Density of the shading.
`col`	Colour of the shading.
`x.axis.start`	Time index the x axis starts from. The NSP intervals of significance get shifted by `x.axis.start-1` prior to plotting.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint.

Value

The function does not return a value.

Author(s)

Piotr Fryzlewicz, <`p.fryzlewicz@lse.ac.uk`>

See Also

`draw_rects_advanced`, `nsp`

Examples

```
set.seed(1)
h <- c(rep(0, 150), 1:150)
x.h <- h + stats::rnorm(300) * 50
x.h.n <- nsp_poly(x.h, 1000, "sim", deg=1)
draw_rects(x.h.n, c(-100, 100))
```

draw_rects_advanced *Plot NSP intervals of significance at appropriate places along the graph of data*

Description

This function plots the intervals of significance returned by one of the nsp* functions, at appropriate places along the graph of data. It shows them as shaded rectangular areas (hence the name of the function) "attached" to the graph of the data. Note: the data sequence y needs to have been plotted beforehand.

Usage

```
draw_rects_advanced(
  y,
  nsp.obj,
  half.height = NULL,
  show.middles = TRUE,
  col.middles = "blue",
  lwd = 3,
  density = 10,
  col.rects = "red",
  x.axis.start = 1
)
```

Arguments

y	The data.
nsp.obj	Object returned by one of the nsp* functions with y on input.
half.height	Half-height of each rectangle; if NULL then set to twice the estimated standard deviation of the data.
show.middles	Whether to display lines corresponding to the midpoints of the rectanlges (rough change-point location estimates).
col.middles	Colour of the midpoint lines.
lwd	Line width for the midpoint lines.
density	Density of the shading.
col.rects	Colour of the shading.
x.axis.start	Time index the x axis starts from. The NSP intervals of significance get shifted by x.axis.start-1 prior to plotting.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint.

Value

The function does not return a value.

Author(s)

Piotr Fryzlewicz, <p.fryzlewicz@lse.ac.uk>

See Also

draw_rects, nsp

Examples

```
set.seed(1)
f <- c(rep(0, 100), 1:100, rep(101, 100))
x.f <- f + 15 * stats::rnorm(300)
x.f.n <- nsp_poly(x.f, 100, "sim", deg=1)
stats::ts.plot(x.f)
draw_rects_advanced(x.f, x.f.n, density = 3)
```

---

nsp *Narrowest Significance Pursuit algorithm with general covariates and user-specified threshold*

Description

This function runs the bare-bones Narrowest Significance Pursuit (NSP) algorithm on data sequence y and design matrix x to obtain localised regions (intervals) of the domain in which the parameters of the linear regression model y_t = beta(t) x_t + z_t significantly depart from constancy (e.g. by containing change-points). For any interval considered by the algorithm, significance is achieved if the multiscale supremum-type deviation measure (see Details for the literature reference) exceeds lambda. This function is mainly to be used by the higher-level functions nsp_poly, nsp_poly_ar and nsp_tvreg (which estimate a suitable lambda so that a given global significance level is guaranteed), and human users may prefer to use those functions instead; however, nsp can also be run directly, if desired. The function works best when the errors z_t in the linear regression formulation y_t = beta(t) x_t + z_t are independent and identically distributed Gaussians.

Usage

```
nsp(y, x, M, lambda, overlap = FALSE, buffer = 0)
```

Arguments

y A vector containing the data sequence being the response in the linear model y_t = beta(t) x_t + z_t.

x The design matrix in the regression model above, with the regressors as columns.

`M` The minimum number of intervals considered at each recursive stage, unless the number of all intervals is smaller, in which case all intervals are used.

`lambda` The threshold parameter for measuring the significance of non-constancy (of the linear regression parameters), for use with the multiscale supremum-type deviation measure described in the paper.

`overlap` If `FALSE`, then on discovering a significant interval, the search continues recursively to the left and to the right of that interval. If `TRUE`, then the search continues to the left and to the right of the midpoint of that interval.

`buffer` A non-negative integer specifying how many observations to leave out immediately to the left and to the right of a detected interval of significance before recursively continuing the search for the next interval.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint.

Value

A list with the following components:

`intervals` A data frame containing the estimated intervals of significance: `starts` and `ends` is where the intervals start and end, respectively; `values` are the values of the deviation measure on each given interval; `midpoints` are the midpoints of the intervals.

`threshold.used` The threshold `lambda`.

Author(s)

Piotr Fryzlewicz, <`p.fryzlewicz@lse.ac.uk`>

See Also

`nsp_poly`, `nsp_poly_ar`, `nsp_tvreg`, `nsp_selfnorm`, `nsp_poly_selfnorm`

Examples

```
set.seed(1)
f <- c(1:100, 100:1, 1:100)
y <- f + stats::rnorm(300) * 15
x <- matrix(0, 300, 2)
x[,1] <- 1
x[,2] <- seq(from = 0, to = 1, length = 300)
nsp(y, x, 100, 15 * thresh_kab(300, .1))
```

nsp_poly *Narrowest Significance Pursuit algorithm for piecewise-polynomial signals*

Description

This function runs the Narrowest Significance Pursuit (NSP) algorithm on a data sequence `y` believed to follow the model y_t = f_t + z_t, where f_t is a piecewise polynomial of degree `deg`, and z_t is noise. It returns localised regions (intervals) of the domain, such that each interval must contain a change-point in the parameters of the polynomial f_t at the global significance level `alpha`. For any interval considered by the algorithm, significant departure from parameter constancy is achieved if the multiscale supremum-type deviation measure (see Details for the literature reference) exceeds a threshold, which is either provided as input or determined from the data (as a function of `alpha`). The function works best when the errors z_t are independent and identically distributed Gaussians.

Usage

```
nsp_poly(
  y,
  M = 1000,
  thresh.type = "univ",
  thresh.val = NULL,
  sigma = NULL,
  alpha = 0.1,
  deg = 0,
  overlap = FALSE
)
```

Arguments

`y` A vector containing the data sequence.

`M` The minimum number of intervals considered at each recursive stage, unless the number of all intervals is smaller, in which case all intervals are used.

`thresh.type` `"univ"` if the significance threshold is to be determined as in Kabluchko (2007); `"sim"` for the degree-dependent threshold determined by simulation (this is only available if the length of `y` does not exceed 2150; for longer sequences obtain a suitable threshold by running `cov_dep_multi_norm_poly` first).

`thresh.val` Numerical value of the significance threshold (lambda in the paper); or `NULL` if the threshold is to be determined from the data (see `thresh.type`).

`sigma` The standard deviation of the errors z_t; if `NULL` then will be estimated from the data via Median Absolute Deviation (for i.i.d. Gaussian sequences) of the first difference.

`alpha` Desired maximum probability of obtaining an interval that does not contain a change-point (the significance threshold will be determined as a function of this parameter).

`deg` The degree of the polynomial pieces in f_t (0 for the piecewise-constant model; 1 for piecewise-linearity, etc.).

`overlap` If `FALSE`, then on discovering a significant interval, the search continues recursively to the left and to the right of that interval. If `TRUE`, then the search continues to the left and to the right of the midpoint of that interval.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint. For how to determine the `"univ"` threshold, see Kabluchko, Z. (2007) "Extreme-value analysis of standardized Gaussian increments". Unpublished.

Value

A list with the following components:

`intervals` A data frame containing the estimated intervals of significance: `starts` and `ends` is where the intervals start and end, respectively; `values` are the values of the deviation measure on each given interval; `midpoints` are their midpoints.

`threshold.used` The threshold value.

Author(s)

Piotr Fryzlewicz, <`p.fryzlewicz@lse.ac.uk`>

See Also

`nsp`, `nsp_poly_ar`, `nsp_tvreg`, `nsp_selfnorm`, `nsp_poly_selfnorm`

Examples

```
set.seed(1)
f <- c(1:100, 100:1, 1:100)
y <- f + stats::rnorm(300) * 15
nsp_poly(y, 100, deg = 1)
```

`nsp_poly_ar` *Narrowest Significance Pursuit algorithm for piecewise-polynomial signals with autoregression*

Description

This function runs the Narrowest Significance Pursuit (NSP) algorithm on a data sequence y believed to follow the model Phi(B)y_t = f_t + z_t, where f_t is a piecewise polynomial of degree deg, Phi(B) is a characteristic polynomial of autoregression of order ord with unknown coefficients, and z_t is noise. The function returns localised regions (intervals) of the domain, such that each interval must contain a change-point in the parameters of the polynomial f_t, or in the autoregressive parameters, at the global significance level alpha. For any interval considered by the algorithm, significant departure from parameter constancy is achieved if the multiscale deviation measure (see Details for the literature reference) exceeds a threshold, which is either provided as input or determined from the data (as a function of alpha). The function works best when the errors z_t are independent and identically distributed Gaussians.

Usage

```
nsp_poly_ar(
  y,
  ord = 1,
  M = 1000,
  thresh.type = "univ",
  thresh.val = NULL,
  sigma = NULL,
  alpha = 0.1,
  deg = 0,
  power = 1/2,
  min.size = 20,
  overlap = FALSE,
  buffer = ord
)
```

Arguments

y A vector containing the data sequence.

ord The assumed order of the autoregression.

M The minimum number of intervals considered at each recursive stage, unless the number of all intervals is smaller, in which case all intervals are used.

thresh.type "univ" if the significance threshold is to be determined as in Kabluchko (2007); "sim" for the degree-dependent threshold determined by simulation (this is only available if the length of y does not exceed 2150; for longer sequences obtain a suitable threshold by running cov_dep_multi_norm_poly first).

thresh.val Numerical value of the significance threshold (lambda in the paper); or NULL if the threshold is to be determined from the data (see thresh.type).

sigma The standard deviation of the errors z_t; if NULL then will be estimated from the data via the MOLS estimator described in the paper.

alpha Desired maximum probability of obtaining an interval that does not contain a change-point (the significance threshold will be determined as a function of this parameter).

`deg` The degree of the polynomial pieces in f_t (0 for the piecewise-constant model; 1 for piecewise-linearity, etc.).

`power` A parameter for the MOLS estimator of sigma; the span of the moving window in the MOLS estimator is `min(n, max(round(n^power), min.size))`, where `n` is the length of `y` (minus `ord`).

`min.size` (See immediately above.)

`overlap` If `FALSE`, then on discovering a significant interval, the search continues recursively to the left and to the right of that interval. If `TRUE`, then the search continues to the left and to the right of the midpoint of that interval.

`buffer` A non-negative integer specifying how many observations to leave out immediately to the left and to the right of a detected interval of significance before recursively continuing the search for the next interval.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint. For how to determine the `"univ"` threshold, see Kabluchko, Z. (2007) "Extreme-value analysis of standardized Gaussian increments". Unpublished.

Value

A list with the following components:

`intervals` A data frame containing the estimated intervals of significance: `starts` and `ends` is where the intervals start and end, respectively; `values` are the values of the deviation measure on each given interval; `midpoints` are their midpoints.

`threshold.used` The threshold value.

Author(s)

Piotr Fryzlewicz, <`p.fryzlewicz@lse.ac.uk`>

See Also

`nsp`, `nsp_poly`, `nsp_tvreg`, `nsp_selfnorm`, `nsp_poly_selfnorm`

Examples

```
set.seed(1)
g <- c(rep(0, 100), rep(10, 100), rep(0, 100))
nsp_poly_ar(stats::filter(g + 2 * stats::rnorm(300), .5, "recursive"), thresh.type="sim")
```

nsp_poly_selfnorm *Self-normalised Narrowest Significance Pursuit algorithm for piecewise-polynomial signals*

Description

This function runs the Narrowest Significance Pursuit (NSP) algorithm on a data sequence `y` believed to follow the model y_t = f_t + z_t, where f_t is a piecewise polynomial of degree `deg`, and z_t is noise. It returns localised regions (intervals) of the domain, such that each interval must contain a change-point in the parameters of the polynomial f_t at the global significance level `alpha`. For any interval considered by the algorithm, significant departure from parameter constancy is achieved if the multiscale deviation measure (see Details for the literature reference) exceeds a threshold, which is either provided as input or determined from the data (as a function of `alpha`). The function assumes independence, symmetry and finite variance of the errors z_t, but little else; in particular they do not need to have a constant variance across t.

Usage

```
nsp_poly_selfnorm(
  y,
  M = 1000,
  thresh.val = NULL,
  power = 1/2,
  min.size = 20,
  alpha = 0.1,
  deg = 0,
  eps = 0.03,
  c = exp(1 + 2 * eps),
  overlap = FALSE
)
```

Arguments

`y` A vector containing the data sequence.

`M` The minimum number of intervals considered at each recursive stage, unless the number of all intervals is smaller, in which case all intervals are used.

`thresh.val` Numerical value of the significance threshold (lambda in the paper); or `NULL` if the threshold is to be determined from the data.

`power` A parameter for the (rough) estimator of the global sum of squares of z_t; the span of the moving window in that estimator is `min(n, max(round(n^power), min.size))`, where `n` is the length of `y`.

`min.size` (See immediately above.)

`alpha` Desired maximum probability of obtaining an interval that does not contain a change-point (the significance threshold will be determined as a function of this parameter).

`deg` The degree of the polynomial pieces in f_t (0 for the piecewise-constant model; 1 for piecewise-linearity, etc.).

`eps` Parameter of the self-normalisation statistic as described in the paper; use default if unsure how to set.

`c` Parameter of the self-normalisation statistic as described in the paper; use default if unsure how to set.

`overlap` If `FALSE`, then on discovering a significant interval, the search continues recursively to the left and to the right of that interval. If `TRUE`, then the search continues to the left and to the right of the midpoint of that interval.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint.

Value

A list with the following components:

`intervals` A data frame containing the estimated intervals of significance: `starts` and `ends` is where the intervals start and end, respectively; `values` are the values of the deviation measure on each given interval; `midpoints` are their midpoints.

`threshold.used` The threshold value.

Author(s)

Piotr Fryzlewicz, <`p.fryzlewicz@lse.ac.uk`>

See Also

`nsp_poly`, `nsp_poly`, `nsp_poly_ar`, `nsp_tvreg`, `nsp_selfnorm`

Examples

```
set.seed(1)
g <- c(rep(0, 100), rep(10, 100), rep(0, 100))
x.g <- g + stats::rnorm(300) * seq(from = 1, to = 4, length = 300)
nsp_poly_selfnorm(x.g, 100)
```

`nsp_selfnorm` *Self-normalised Narrowest Significance Pursuit algorithm with general covariates and user-specified threshold*

Description

This function runs the self-normalised Narrowest Significance Pursuit (NSP) algorithm on data sequence `y` and design matrix `x` to obtain localised regions (intervals) of the domain in which the parameters of the linear regression model y_t = beta(t) x_t + z_t significantly depart from constancy (e.g. by containing change-points). For any interval considered by the algorithm, significant departure from parameter constancy is achieved if the self-normalised multiscale deviation measure (see Details for the literature reference) exceeds `lambda`. This function is used by the higher-level function `nsp_poly_selfnorm` (which estimates a suitable `lambda` so that a given global significance level is guaranteed), and human users may prefer to use that function if `x` describe polynomial covariates; however, `nsp_selfnorm` can also be run directly, if desired. The function assumes independence, symmetry and finite variance of the errors z_t, but little else; in particular they do not need to have a constant variance across t.

Usage

```
nsp_selfnorm(
  y,
  x,
  M,
  lambda,
  power = 1/2,
  min.size = 20,
  eps = 0.03,
  c = exp(1 + 2 * eps),
  overlap = FALSE
)
```

Arguments

`y` A vector containing the data sequence being the response in the linear model y_t = beta(t) x_t + z_t.

`x` The design matrix in the regression model above, with the regressors as columns.

`M` The minimum number of intervals considered at each recursive stage, unless the number of all intervals is smaller, in which case all intervals are used.

`lambda` The threshold parameter for measuring the significance of non-constancy (of the linear regression parameters), for use with the self-normalised multiscale supremum-type deviation measure described in the paper.

`power` A parameter for the (rough) estimator of the global sum of squares of z_t; the span of the moving window in that estimator is `min(n, max(round(n^power), min.size))`, where `n` is the length of `y`.

`min.size` (See immediately above.)

`eps` Parameter of the self-normalisation statistic as described in the paper; use default if unsure how to set.

`c` Parameter of the self-normalisation statistic as described in the paper; use default if unsure how to set.

`overlap` If `FALSE`, then on discovering a significant interval, the search continues recursively to the left and to the right of that interval. If `TRUE`, then the search continues to the left and to the right of the midpoint of that interval.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint.

Value

A list with the following components:

`intervals` A data frame containing the estimated intervals of significance: `starts` and `ends` is where the intervals start and end, respectively; `values` are the values of the deviation measure on each given interval; `midpoints` are their midpoints.

`threshold.used` The threshold `lambda`.

Author(s)

Piotr Fryzlewicz, <`p.fryzlewicz@lse.ac.uk`>

See Also

`nsp_poly`, `nsp_poly`, `nsp_poly_ar`, `nsp_tvreg`, `nsp_poly_selfnorm`

Examples

```
set.seed(1)
g <- c(rep(0, 100), rep(10, 100), rep(0, 100))
x.g <- g + stats::rnorm(300) * seq(from = 1, to = 4, length = 300)
wn003 <- sim_max_holder(100, 500, .03)
lambda <- as.numeric(stats::quantile(wn003, .9))
nsp_selfnorm(x.g, matrix(1, 300, 1), 100, lambda)
```

`nsp_tvreg` *Narrowest Significance Pursuit algorithm with general covariates*

Description

This function runs the Narrowest Significance Pursuit (NSP) algorithm on data sequence `y` and design matrix `x` to return localised regions (intervals) of the domain in which the parameters of the linear regression model y_t = beta(t) x_t + z_t significantly depart from constancy (e.g. by containing change-points), at the global significance level `alpha`. For any interval considered by the algorithm, significant departure from parameter constancy is achieved if the multiscale deviation measure (see Details for the literature reference) exceeds a threshold, which is either provided as input or determined from the data (as a function of `alpha`). The function works best when the errors z_t in the linear regression formulation y_t = beta(t) x_t + z_t are independent and identically distributed Gaussians.

Usage

```
nsp_tvreg(
  y,
  x,
  M = 1000,
  thresh.val = NULL,
  sigma = NULL,
  alpha = 0.1,
  power = 1/2,
  min.size = 20,
  overlap = FALSE
)
```

Arguments

`y` A vector containing the data sequence being the response in the linear model y_t = beta(t) x_t + z_t.

`x` The design matrix in the regression model above, with the regressors as columns.

`M` The minimum number of intervals considered at each recursive stage, unless the number of all intervals is smaller, in which case all intervals are used.

`thresh.val` Numerical value of the significance threshold (lambda in the paper); or `NULL` if the threshold is to be determined from the data (see `thresh.type`).

`sigma` The standard deviation of the errors z_t; if `NULL` then will be estimated from the data via the MOLS estimator described in the paper.

`alpha` Desired maximum probability of obtaining an interval that does not contain a change-point (the significance threshold will be determined as a function of this parameter).

`power` A parameter for the MOLS estimator of sigma; the span of the moving window in the MOLS estimator is `min(n, max(round(n^power), min.size))`, where `n` is the length of `y`.

`min.size` (See immediately above.)

`overlap` If `FALSE`, then on discovering a significant interval, the search continues recursively to the left and to the right of that interval. If `TRUE`, then the search continues to the left and to the right of the midpoint of that interval.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint.

Value

A list with the following components:

`intervals` A data frame containing the estimated intervals of significance: `starts` and `ends` is where the intervals start and end, respectively; `values` are the values of the deviation measure on each given interval; `midpoints` are their midpoints.

`threshold.used` The threshold value.

Author(s)

Piotr Fryzlewicz, <`p.fryzlewicz@lse.ac.uk`>

See Also

`nsp`, `nsp_poly`, `nsp_poly_ar`, `nsp_selfnorm`, `nsp_poly_selfnorm`

Examples

```
set.seed(1)
f <- c(1:100, 100:1, 1:100)
y <- f + stats::rnorm(300) * 15
x <- matrix(0, 300, 2)
x[,1] <- 1
x[,2] <- seq(from = 0, to = 1, length = 300)
nsp_tvreg(y, x, 100)
```

`sim_max_holder` *Simulate Holder-like norm of the Wiener process for use in self-normalised NSP*

Description

This function simulates a sample of size `N` of values of the Holder-like norm of the Wiener process discretised with step 1/`n`. The sample can then be used to find a suitable threshold for use with the self-normalised NSP.

Usage

```
sim_max_holder(n, N, eps, c = exp(1 + 2 * eps))
```

Arguments

n	Number of equispaced sampling points for the Wiener process on [0,1].
N	Desired number of simulated values of the norm.
eps	Parameter of the self-normalisation statistic as described in the paper.
c	Parameter of the self-normalisation statistic as described in the paper; use default if unsure how to set.

Details

The NSP algorithm is described in P. Fryzlewicz (2021) "Narrowest Significance Pursuit: inference for multiple change-points in linear models", preprint.

Value

Sample of size N containing the simulated norms.

Author(s)

Piotr Fryzlewicz, <p.fryzlewicz@lse.ac.uk>

See Also

nsp_selfnorm, nsp_poly_selfnorm, cov_dep_multi_norm, cov_dep_multi_norm_poly

Examples

```
set.seed(1)
g <- c(rep(0, 100), rep(10, 100), rep(0, 100))
x.g <- g + stats::rnorm(300) * seq(from = 1, to = 4, length = 300)
wn003 <- sim_max_holder(100, 500, .03)
lambda <- as.numeric(stats::quantile(wn003, .9))
nsp_poly_selfnorm(x.g, M = 100, thresh.val = lambda)
```

thresh_kab *Compute the theoretical threshold for the multiscale sup-norm if the underlying distribution is standard normal*

Description

This function computes the theoretical threshold, corresponding to the given significance level alpha, for the multiscale sup-norm if the underlying distribution is standard normal.

Usage

```
thresh_kab(n, alpha = 0.1, method = "asymp")
```

Arguments

`n`	The sample size.
`alpha`	The significance level.
`method`	"asymp" for the asymptotic method; "bound" for the Bonferroni method.

Details

For the underlying theory, see Z. Kabluchko (2007) Extreme-value analysis of standardized Gaussian increments. Unpublished.

Value

The desired threshold.

Author(s)

Piotr Fryzlewicz, <`p.fryzlewicz@lse.ac.uk`>

See Also

`cov_dep_multi_norm`, `cov_dep_multi_norm_poly`, `sim_max_holder`

Examples

```
set.seed(1)
f <- c(1:100, 100:1, 1:100)
y <- f + stats::rnorm(300) * 15
x <- matrix(0, 300, 2)
x[,1] <- 1
x[,2] <- seq(from = 0, to = 1, length = 300)
nsp(y, x, 100, 15 * thresh_kab(300, .1))
```

Index