Package ‘CptNonPar’

April 23, 2024

Type Package

Title Nonparametric Change Point Detection for Multivariate Time Series

Version 0.2.1

Depends R (>= 4.1.0)

Maintainer Euan T. McGonigle <e.t.mcgonigle@soton.ac.uk>

License GPL (>= 3)

Description Implements the nonparametric moving sum procedure for detecting changes in the joint characteristic function (NP-MOJO) for multiple change point detection in multivariate time series. See McGonigle, E. T., Cho, H. (2023) <doi:10.48550/arXiv.2305.07581> for description of the NP-MOJO methodology.

Encoding UTF-8

LinkingTo Rcpp

Imports Rcpp, doParallel, parallel, parallelly, foreach, Rfast, iterators, stats

URL https://github.com/EuanMcGonigle/CptNonPar

BugReports https://github.com/EuanMcGonigle/CptNonPar/issues

RoxygenNote 7.3.1

Suggests covr, testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation yes

Author Euan T. McGonigle [aut, cre], Haeran Cho [aut]

Repository CRAN

Date/Publication 2024-04-23 15:30:07 UTC

R topics documented:

`multilag.cpts.merge` *Merge Change Point Estimators from Multiple Lags*

Description

Merges change point estimators from different lagged values into a final set of overall change point estimators.

Usage

```
multilag.cpts.merge(
  x.c,
  eta.merge = 1,
  merge.type = c("sequential", "bottom-up")[1]
)
```

Arguments

`x.c` A `list` object, where each element of the list is the output of the np.mojo function computed at a different lag.

`eta.merge` A positive numeric value for the minimal mutual distance of changes, relative to bandwidth, used to merge change point estimators across different lags.

`merge.type` String indicating the method used to merge change point estimators from different lags. Possible choices are

- `"sequential"`: starting from the left-most change point estimator and proceeding forward in time, estimators are grouped into clusters based on mutual distance. The estimator yielding the smallest corresponding p-value is chosen as the change point estimator for that cluster. See McGonigle and Cho (2023) for details.
- `"bottom-up"`: starting with the smallest p-value, the change points are merged using bottom-up merging (Messer et al. (2014)).

Details

See McGonigle and Cho (2023) for further details.

Value

A `list` object which contains the following fields

`cpts`	A matrix with rows corresponding to final change point estimators, with estimated change point location and associated lag and p-value given in columns.
`cpt.clusters`	A `list` object of length given by the number of detected change points. Each field contains a matrix of all change point estimators that are declared to be associated to the corresponding change point in the `cpts` field.

References

McGonigle, E.T., Cho, H. (2023). Nonparametric data segmentation in multivariate time series via joint characteristic functions. *arXiv preprint arXiv:2305.07581*.

Messer M., Kirchner M., Schiemann J., Roeper J., Neininger R., Schneider G. (2014). A Multiple Filter Test for the Detection of Rate Changes in Renewal Processes with Varying Variance. *The Annals of Applied Statistics*, 8(4), 2027-2067.

See Also

np.mojo, np.mojo.multilag

Examples

```
set.seed(1)
n <- 500
noise <- c(rep(1, 300), rep(0.4, 200)) * stats::arima.sim(model = list(ar = 0.3), n = n)
signal <- c(rep(0, 100), rep(2, 400))
x <- signal + noise
x.c0 <- np.mojo(x, G = 83, lag = 0)
x.c1 <- np.mojo(x, G = 83, lag = 1)
x.c <- multilag.cpts.merge(list(x.c0, x.c1))
x.c
```

`multiscale.np.mojo` *Multiscale Nonparametric Multiple Lag Change Point Detection*

Description

For a given set of bandwidths and lagged values of the time series, performs multiscale nonparametric change point detection of a possibly multivariate time series.

Usage

```
multiscale.np.mojo(
  x,
  G,
  lags = c(0, 1),
```

```
  kernel.f = c("quad.exp", "gauss", "euclidean", "laplace", "sine")[1],
  kern.par = 1,
  data.driven.kern.par = TRUE,
  threshold = c("bootstrap", "manual")[1],
  threshold.val = NULL,
  alpha = 0.1,
  reps = 199,
  boot.dep = 1.5 * (nrow(as.matrix(x))^(1/3)),
  parallel = FALSE,
  boot.method = c("mean.subtract", "no.mean.subtract")[1],
  criterion = c("eta", "epsilon", "eta.and.epsilon")[3],
  eta = 0.4,
  epsilon = 0.02,
  use.mean = FALSE,
  eta.merge = 1,
  merge.type = c("sequential", "bottom-up")[1],
  eta.bottom.up = 0.8
)
```

Arguments

`x` — Input data (a `numeric` vector or an object of classes `ts` and `timeSeries`, or a `numeric` matrix with rows representing observations and columns representing variables).

`G` — A numeric vector containing the moving sum bandwidths; all values in the vector `G` should be less than half the length of the time series.

`lags` — A `numeric` vector giving the range of lagged values of the time series that will be used to detect changes. See np.mojo.multilag for further details.

`kernel.f` — String indicating which kernel function to use when calculating the NP-MOJO detector statistics; with `kern.par` $= a$, possible values are

- `"quad.exp"`: kernel h_2 in McGonigle and Cho (2023), kernel 5 in Fan et al. (2017):
$$h(x,y) = \prod_{i=1}^{2p} \frac{(2a - (x_i - y_i)^2) \exp(-\frac{1}{4a}(x_i - y_i)^2)}{2a}.$$
- `"gauss"`: kernel h_1 in McGonigle and Cho (2023), the standard Gaussian kernel:
$$h(x,y) = \exp(-\frac{a^2}{2}\|x-y\|^2).$$
- `"euclidean"`: kernel h_3 in McGonigle and Cho (2023), the Euclidean distance-based kernel:
$$h(x,y) = \|x-y\|^a.$$
- `"laplace"`: kernel 2 in Fan et al. (2017), based on a Laplace weight function:
$$h(x,y) = \prod_{i=1}^{2p} \left(1 + a^2(x_i - y_i)^2\right)^{-1}.$$

- `"sine"`: kernel 4 in Fan et al. (2017), based on a sinusoidal weight function:

$$h(x, y) = \prod_{i=1}^{2p} \frac{-2|x_i - y_i| + |x_i - y_i - 2a| + |x_i - y_i + 2a|}{4a}.$$

`kern.par` The tuning parameter that appears in the expression for the kernel function, which acts as a scaling parameter.

`data.driven.kern.par`
A `logical` variable, if set to `TRUE`, then the kernel tuning parameter is calculated using the median heuristic, if `FALSE` it is given by `kern.par`.

`threshold` String indicating how the threshold is computed. Possible values are

- `"bootstrap"`: the threshold is calculated using the bootstrap method with significance level `alpha`.
- `"manual"`: the threshold is set by the user and must be specified using the `threshold.val` parameter.

`threshold.val` The value of the threshold used to declare change points, only to be used if `threshold = "manual"`.

`alpha` a numeric value for the significance level with `0 <= alpha <= 1`; use iff `threshold = "bootstrap"`.

`reps` An integer value for the number of bootstrap replications performed, if `threshold = "bootstrap"`.

`boot.dep` A positive value for the strength of dependence in the multiplier bootstrap sequence, if `threshold = "bootstrap"`.

`parallel` A `logical` variable, if set to `TRUE`, then parallel computing is used in the bootstrapping procedure if bootstrapping is performed.

`boot.method` A string indicating the method for creating bootstrap replications. It is not recommended to change this. Possible choices are

- `"mean.subtract"`: the default choice, as described in McGonigle and Cho (2023). Empirical mean subtraction is performed to the bootstrapped replicates, improving power.
- `"no.mean.subtract"`: empirical mean subtraction is not performed, improving size control.

`criterion` String indicating how to determine whether each point `k` at which NP-MOJO statistic exceeds the threshold is a change point; possible values are

- `"epsilon"`: `k` is the maximum of its local exceeding environment, which has at least size `epsilon*G`.
- `"eta"`: there is no larger exceeding in an `eta*G` environment of `k`.
- `"eta.and.epsilon"`: the recommended default option; `k` satisfies both the eta and epsilon criterion. Recommended to use with the standard value of eta that would be used if `criterion = "eta"` (e.g. 0.4), but much smaller value of epsilon than would be used if `criterion = "epsilon"`, e.g. 0.02.

`eta` A positive numeric value for the minimal mutual distance of changes, relative to bandwidth (if `criterion = "eta"` or `criterion = "eta.and.epsilon"`).

epsilon — a numeric value in (0,1] for the minimal size of exceeding environments, relative to moving sum bandwidth (if `criterion = "epsilon"` or `criterion = "eta.and.epsilon"`).

use.mean — `Logical variable`, only to be used if `data.drive.kern.par=TRUE`. If set to `TRUE`, the mean of pairwise distances is used to set the kernel function tuning parameter, instead of the median. May be useful for binary data, not recommended to be used otherwise.

eta.merge — A positive numeric value for the minimal mutual distance of changes, relative to bandwidth, used to merge change point estimators across different lags.

merge.type — String indicating the method used to merge change point estimators from different lags. Possible choices are

- `"sequential"`: Starting from the left-most change point estimator and proceeding forward in time, estimators are grouped into clusters based on mutual distance. The estimator yielding the smallest corresponding p-value is chosen as the change point estimator for that cluster. See McGonigle and Cho (2023) for details.
- `"bottom-up"`: starting with the smallest p-value, the change points are merged using bottom-up merging (Messer et al. (2014)).

eta.bottom.up — A positive numeric value for the minimal mutual distance of changes, relative to bandwidth, for use in bottom-up merging of change point estimators across multiple bandwidths.

Details

The multi-lag NP-MOJO algorithm for nonparametric change point detection is described in McGonigle, E. T. and Cho, H. (2023) Nonparametric data segmentation in multivariate time series via joint characteristic functions. *arXiv preprint arXiv:2305.07581*. The multiscale version uses bottom-up merging to combine the results of the multi-lag NP-MOJO algorithm performed over a given set of bandwidths.

Value

A `list` object that contains the following fields:

G — Set of moving window bandwidths

lags — Lags used to detect changes

`kernel.f`, `data.driven.kern.par`, `use.mean` — Input parameters

`threshold`, `alpha`, `reps`, `boot.dep`, `boot.method`, `parallel` — Input parameters

`criterion`, `eta`, `epsilon` — Input parameters

cpts — A matrix with rows corresponding to final change point estimators, with estimated change point location and associated detection bandwidth, lag and p-value given in columns.

References

McGonigle, E.T., Cho, H. (2023). Nonparametric data segmentation in multivariate time series via joint characteristic functions. *arXiv preprint arXiv:2305.07581*.

Fan, Y., de Micheaux, P.L., Penev, S. and Salopek, D. (2017). Multivariate nonparametric test of independence. *Journal of Multivariate Analysis*, 153, pp.189-210.

Messer M., Kirchner M., Schiemann J., Roeper J., Neininger R., Schneider G. (2014). A Multiple Filter Test for the Detection of Rate Changes in Renewal Processes with Varying Variance. *The Annals of Applied Statistics*, 8(4), 2027-2067.

See Also

np.mojo.multilag

Examples

```
set.seed(1)
n <- 500
noise <- c(rep(1, 300), rep(0.4, 200)) * stats::arima.sim(model = list(ar = 0.3), n = n)
signal <- c(rep(0, 100), rep(2, 400))
x <- signal + noise
x.c <- multiscale.np.mojo(x, G = c(50, 80), lags = c(0, 1))
x.c$cpts
```

---

np.mojo — *Nonparametric Single Lag Change Point Detection*

Description

For a given lagged value of the time series, performs nonparametric change point detection of a possibly multivariate time series. If lag $\ell = 0$, then only marginal changes are detected. If lag $\ell \neq 0$, then changes in the pairwise distribution of $(X_t, X_{t+\ell})$ are detected.

Usage

```
np.mojo(
  x,
  G,
  lag = 0,
  kernel.f = c("quad.exp", "gauss", "euclidean", "laplace", "sine")[1],
  kern.par = 1,
  data.driven.kern.par = TRUE,
  alpha = 0.1,
  threshold = c("bootstrap", "manual")[1],
  threshold.val = NULL,
  reps = 199,
  boot.dep = 1.5 * (nrow(as.matrix(x))^(1/3)),
  parallel = FALSE,
```

```
  boot.method = c("mean.subtract", "no.mean.subtract")[1],
  criterion = c("eta", "epsilon", "eta.and.epsilon")[3],
  eta = 0.4,
  epsilon = 0.02,
  use.mean = FALSE
)
```

Arguments

`x` Input data (a `numeric` vector or an object of classes `ts` and `timeSeries`, or a `numeric` matrix with rows representing observations and columns representing variables).

`G` An integer value for the moving sum bandwidth; `G` should be less than half the length of the time series.

`lag` The lagged values of the time series used to detect changes. If `lag` $\ell = 0$, then only marginal changes are detected. If `lag` $\ell \neq 0$, then changes in the pairwise distribution of $(X_t, X_{t+\ell})$ are detected.

`kernel.f` String indicating which kernel function to use when calculating the NP-MOJO detectors statistics; with `kern.par` $= a$, possible values are

- `"quad.exp"`: kernel h_2 in McGonigle and Cho (2023), kernel 5 in Fan et al. (2017):
$$h(x,y) = \prod_{i=1}^{2p} \frac{(2a - (x_i - y_i)^2) \exp(-\frac{1}{4a}(x_i - y_i)^2)}{2a}.$$
- `"gauss"`: kernel h_1 in McGonigle and Cho (2023), the standard Gaussian kernel:
$$h(x,y) = \exp(-\frac{a^2}{2}\|x - y\|^2).$$
- `"euclidean"`: kernel h_3 in McGonigle and Cho (2023), the Euclidean distance-based kernel:
$$h(x,y) = \|x - y\|^a.$$
- `"laplace"`: kernel 2 in Fan et al. (2017), based on a Laplace weight function:
$$h(x,y) = \prod_{i=1}^{2p} \left(1 + a^2(x_i - y_i)^2\right)^{-1}.$$
- `"sine"`: kernel 4 in Fan et al. (2017), based on a sinusoidal weight function:
$$h(x,y) = \prod_{i=1}^{2p} \frac{-2|x_i - y_i| + |x_i - y_i - 2a| + |x_i - y_i + 2a|}{4a}.$$

`kern.par` The tuning parameter that appears in the expression for the kernel function, which acts as a scaling parameter, only to be used if `data.driven.kern.par = FALSE`. If `kernel.f = "euclidean"`, then `kern.par` $\in (0,2)$, otherwise `kern.par` > 0.

`data.driven.kern.par`
: A `logical` variable, if set to `TRUE`, then the kernel tuning parameter is calculated using the median heuristic, if `FALSE` it is given by `kern.par`.

`alpha`
: A numeric value for the significance level with `0 <= alpha <= 1`; use iff `threshold = "bootstrap"`.

`threshold`
: String indicating how the threshold is computed. Possible values are
  - `"bootstrap"`: the threshold is calculated using the bootstrap method with significance level `alpha`.
  - `"manual"`: the threshold is set by the user and must be specified using the `threshold.val` parameter.

`threshold.val`
: The value of the threshold used to declare change points, only to be used if `threshold = "manual"`.

`reps`
: An integer value for the number of bootstrap replications performed, if `threshold = "bootstrap"`.

`boot.dep`
: A positive value for the strength of dependence in the multiplier bootstrap sequence, if `threshold = "bootstrap"`.

`parallel`
: A `logical` variable, if set to `TRUE`, then parallel computing is used in the bootstrapping procedure if bootstrapping is performed.

`boot.method`
: A string indicating the method for creating bootstrap replications. It is not recommended to change this. Possible choices are
  - `"mean.subtract"`: the default choice, as described in McGonigle and Cho (2023). Empirical mean subtraction is performed to the bootstrapped replicates, improving power.
  - `"no.mean.subtract"`: empirical mean subtraction is not performed, improving size control.

`criterion`
: String indicating how to determine whether each point `k` at which NP-MOJO statistic exceeds the threshold is a change point; possible values are
  - `"epsilon"`: `k` is the maximum of its local exceeding environment, which has at least size `epsilon*G`.
  - `"eta"`: there is no larger exceeding in an `eta*G` environment of `k`.
  - `"eta.and.epsilon"`: the recommended default option; `k` satisfies both the eta and epsilon criterion. Recommended to use with the standard value of eta that would be used if `criterion = "eta"` (e.g. 0.4), but much smaller value of epsilon than would be used if `criterion = "epsilon"`, e.g. 0.02.

`eta`
: A positive numeric value for the minimal mutual distance of changes, relative to bandwidth (if `criterion = "eta"` or `criterion = "eta.and.epsilon"`).

`epsilon`
: a numeric value in (0,1] for the minimal size of exceeding environments, relative to moving sum bandwidth (if `criterion = "epsilon"` or `criterion = "eta.and.epsilon"`).

`use.mean`
: `Logical variable`, only to be used if `data.drive.kern.par=TRUE`. If set to `TRUE`, the mean of pairwise distances is used to set the kernel function tuning parameter, instead of the median. May be useful for binary data, not recommended to be used otherwise.

Details

The single-lag NP-MOJO algorithm for nonparametric change point detection is described in McGonigle, E. T. and Cho, H. (2023) Nonparametric data segmentation in multivariate time series via joint characteristic functions. *arXiv preprint arXiv:2305.07581*.

Value

A `list` object that contains the following fields:

`x`	Input data
`G`	Moving window bandwidth
`lag`	Lag used to detect changes
`kernel.f, data.driven.kern.par, use.mean`	Input parameters
`kern.par`	The value of the kernel tuning parameter
`threshold, alpha, reps, boot.dep, boot.method, parallel`	Input parameters
`threshold.val`	Threshold value for declaring change points
`criterion, eta, epsilon`	Input parameters
`test.stat`	A vector containing the NP-MOJO detector statistics computed from the input data
`cpts`	A vector containing the estimated change point locations
`p.vals`	The corresponding p values of the change points, if the bootstrap method was used

References

McGonigle, E.T., Cho, H. (2023). Nonparametric data segmentation in multivariate time series via joint characteristic functions. *arXiv preprint arXiv:2305.07581*.

Fan, Y., de Micheaux, P.L., Penev, S. and Salopek, D. (2017). Multivariate nonparametric test of independence. *Journal of Multivariate Analysis*, 153, pp.189-210.

See Also

np.mojo.multilag

Examples

```
set.seed(1)
n <- 500
noise <- c(rep(1, 300), rep(0.4, 200)) * stats::arima.sim(model = list(ar = 0.3), n = n)
signal <- c(rep(0, 100), rep(2, 400))
x <- signal + noise
x.c <- np.mojo(x, G = 83, lag = 0)
x.c$cpts
x.c$p.vals
```

np.mojo.multilag *Nonparametric Multiple Lag Change Point Detection*

Description

For a given set of lagged values of the time series, performs nonparametric change point detection of a possibly multivariate time series.

Usage

```
np.mojo.multilag(
  x,
  G,
  lags = c(0, 1),
  kernel.f = c("quad.exp", "gauss", "euclidean", "laplace", "sine")[1],
  kern.par = 1,
  data.driven.kern.par = TRUE,
  threshold = c("bootstrap", "manual")[1],
  threshold.val = NULL,
  alpha = 0.1,
  reps = 199,
  boot.dep = 1.5 * (nrow(as.matrix(x))^(1/3)),
  parallel = FALSE,
  boot.method = c("mean.subtract", "no.mean.subtract")[1],
  criterion = c("eta", "epsilon", "eta.and.epsilon")[3],
  eta = 0.4,
  epsilon = 0.02,
  use.mean = FALSE,
  eta.merge = 1,
  merge.type = c("sequential", "bottom-up")[1]
)
```

Arguments

`x` Input data (a `numeric` vector or an object of classes `ts` and `timeSeries`, or a `numeric` matrix with rows representing observations and columns representing variables).

`G` An integer value for the moving sum bandwidth; `G` should be less than half the length of the time series.

`lags` A `numeric` vector giving the range of lagged values of the time series that will be used to detect changes. See np.mojo for further details.

`kernel.f` String indicating which kernel function to use when calculating the NP-MOJO detector statistics; with `kern.par` $= a$, possible values are

- "quad.exp": kernel h_2 in McGonigle and Cho (2023), kernel 5 in Fan et al. (2017):

$$h(x,y) = \prod_{i=1}^{2p} \frac{(2a - (x_i - y_i)^2)\exp(-\frac{1}{4a}(x_i - y_i)^2)}{2a}.$$

- "gauss": kernel h_1 in McGonigle and Cho (2023), the standard Gaussian kernel:

$$h(x,y) = \exp(-\frac{a^2}{2}\|x - y\|^2).$$

- "euclidean": kernel h_3 in McGonigle and Cho (2023), the Euclidean distance-based kernel:

$$h(x,y) = \|x - y\|^a.$$

- "laplace": kernel 2 in Fan et al. (2017), based on a Laplace weight function:

$$h(x,y) = \prod_{i=1}^{2p} \left(1 + a^2(x_i - y_i)^2\right)^{-1}.$$

- "sine": kernel 4 in Fan et al. (2017), based on a sinusoidal weight function:

$$h(x,y) = \prod_{i=1}^{2p} \frac{-2|x_i - y_i| + |x_i - y_i - 2a| + |x_i - y_i + 2a|}{4a}.$$

kern.par
: The tuning parameter that appears in the expression for the kernel function, which acts as a scaling parameter.

data.driven.kern.par
: A logical variable, if set to TRUE, then the kernel tuning parameter is calculated using the median heuristic, if FALSE it is given by kern.par.

threshold
: String indicating how the threshold is computed. Possible values are
  - "bootstrap": the threshold is calculated using the bootstrap method with significance level alpha.
  - "manual": the threshold is set by the user and must be specified using the threshold.val parameter.

threshold.val
: The value of the threshold used to declare change points, only to be used if threshold = "manual".

alpha
: a numeric value for the significance level with 0 <= alpha <= 1; use iff threshold = "bootstrap".

reps
: An integer value for the number of bootstrap replications performed, if threshold = "bootstrap".

boot.dep
: A positive value for the strength of dependence in the multiplier bootstrap sequence, if threshold = "bootstrap".

parallel
: A logical variable, if set to TRUE, then parallel computing is used in the bootstrapping procedure if bootstrapping is performed.

boot.method
: A string indicating the method for creating bootstrap replications. It is not recommended to change this. Possible choices are

- "mean.subtract": the default choice, as described in McGonigle and Cho (2023). Empirical mean subtraction is performed to the bootstrapped replicates, improving power.
- "no.mean.subtract": empirical mean subtraction is not performed, improving size control.

criterion
: String indicating how to determine whether each point k at which NP-MOJO statistic exceeds the threshold is a change point; possible values are
  - "epsilon": k is the maximum of its local exceeding environment, which has at least size epsilon*G.
  - "eta": there is no larger exceeding in an eta*G environment of k.
  - "eta.and.epsilon": the recommended default option; k satisfies both the eta and epsilon criterion. Recommended to use with the standard value of eta that would be used if criterion = "eta" (e.g. 0.4), but much smaller value of epsilon than would be used if criterion = "epsilon", e.g. 0.02.

eta
: A positive numeric value for the minimal mutual distance of changes, relative to bandwidth (if criterion = "eta" or criterion = "eta.and.epsilon").

epsilon
: a numeric value in (0,1] for the minimal size of exceeding environments, relative to moving sum bandwidth (if criterion = "epsilon" or criterion = "eta.and.epsilon").

use.mean
: Logical variable, only to be used if data.drive.kern.par=TRUE. If set to TRUE, the mean of pairwise distances is used to set the kernel function tuning parameter, instead of the median. May be useful for binary data, not recommended to be used otherwise.

eta.merge
: A positive numeric value for the minimal mutual distance of changes, relative to bandwidth, used to merge change point estimators across different lags.

merge.type
: String indicating the method used to merge change point estimators from different lags. Possible choices are
  - "sequential": Starting from the left-most change point estimator and proceeding forward in time, estimators are grouped into clusters based on mutual distance. The estimator yielding the smallest corresponding p-value is chosen as the change point estimator for that cluster. See McGonigle and Cho (2023) for details.
  - "bottom-up": starting with the smallest p-value, the change points are merged using bottom-up merging (Messer et al. (2014)).

Details

The multi-lag NP-MOJO algorithm for nonparametric change point detection is described in McGonigle, E. T. and Cho, H. (2023) Nonparametric data segmentation in multivariate time series via joint characteristic functions. *arXiv preprint arXiv:2305.07581*.

Value

A list object that contains the following fields:

G
: Moving window bandwidth

`lags` Lags used to detect changes

`kernel.f, data.driven.kern.par, use.mean`
Input parameters

`threshold, alpha, reps, boot.dep, boot.method, parallel`
Input parameters

`criterion, eta, epsilon`
Input parameters

`cpts` A matrix with rows corresponding to final change point estimators, with estimated change point location and associated lag and p-value given in columns.

`cpt.clusters` A `list` object of length given by the number of detected change points. Each field contains a matrix of all change point estimators that are declared to be associated to the corresponding change point in the `cpts` field.

References

McGonigle, E.T., Cho, H. (2023). Nonparametric data segmentation in multivariate time series via joint characteristic functions. *arXiv preprint arXiv:2305.07581*.

Fan, Y., de Micheaux, P.L., Penev, S. and Salopek, D. (2017). Multivariate nonparametric test of independence. *Journal of Multivariate Analysis*, 153, pp.189-210.

Messer M., Kirchner M., Schiemann J., Roeper J., Neininger R., Schneider G. (2014). A Multiple Filter Test for the Detection of Rate Changes in Renewal Processes with Varying Variance. *The Annals of Applied Statistics*, 8(4), 2027-2067.

See Also

np.mojo, multilag.cpts.merge

Examples

```
set.seed(1)
n <- 500
noise <- c(rep(1, 300), rep(0.4, 200)) * stats::arima.sim(model = list(ar = 0.3), n = n)
signal <- c(rep(0, 100), rep(2, 400))
x <- signal + noise
x.c <- np.mojo.multilag(x, G = 83, lags = c(0, 1))
x.c$cpts
x.c$cpt.clusters
```

Index