Package ‘kerSeg’

July 22, 2025

Type Package

Title New Kernel-Based Change-Point Detection

Version 1.1

Author Hoseung Song [aut, cre],
Hao Chen [aut]

Maintainer Hoseung Song <hosong@ucdavis.edu>

Description New kernel-based test and fast tests for detecting change-points or changed-intervals where the distributions abruptly change. They work well particularly for high-dimensional data.
Song, H. and Chen, H. (2022)
<doi:10.48550/arXiv.2206.01853>.

Imports Rcpp (>= 1.0.7)

LinkingTo Rcpp

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-08-22 20:40:04 UTC

Contents

gaussiankernel *Compute the Gaussian kernel matrix*

Description

This function provides the Gaussian kernel matrix computed with the median heuristic bandwidth.

Usage

```
gaussiankernel(X)
```

Arguments

X	The samples in the sequence.

Value

Returns a numeric matrix, the Gaussian kernel matrix computed with the specified bandwidth.

See Also

kerSeg-package,kerseg1,kerseg2

Examples

```
## Sequence : change in the mean in the middle of the sequence.
d = 50
mu = 2
tau = 50
n = 100
set.seed(1)
y = rbind(matrix(rnorm(d*tau),tau), matrix(rnorm(d*(n-tau),mu/sqrt(d)), n-tau))

K = gaussiankernel(y) # Gaussian kernel matrix
```

kerSeg *New kernel-based change-point detection*

Description

This package can be used to detect change-points where the distributions abruptly change. The Gaussian kernel with the median heuristic, which is the median of all pairwise distances among observations, is used.

Details

To compute the Gaussian kernel matrix with the median heuristic bandwidth, the function gaussiankernel should be used. The main functions are kerseg1 for the single change-point alternative and kerseg2 for the changed-interval alternative.

Author(s)

Hoseung Song and Hao Chen

Maintainer: Hoseung Song (hosong@ucdavis.edu)

References

Song, H. and Chen, H. (2022). New kernel-based change-point detection. arXiv:2206.01853

See Also

kerseg1, kerseg2, gaussiankernel

Examples

```
## Sequence 1: change in the mean in the middle of the sequence.
d = 50
mu = 2
tau = 15
n = 50
set.seed(1)
y = rbind(matrix(rnorm(d*tau),tau), matrix(rnorm(d*(n-tau),mu/sqrt(d)), n-tau))
K = gaussiankernel(y) # Gaussian kernel matrix
a = kerseg1(n, K, pval.perm=TRUE, B=1000)
# output results based on the permutation and the asymptotic results.
# the scan statistics can be found in a$scanZ.
# the approximated p-values can be found in a$appr.
# the permutation p-values can be found in a$perm.

## Sequence 2: change in both the mean and variance away from the middle of the sequence.
d = 50
mu = 2
sigma = 0.7
tau = 35
n = 50
set.seed(1)
y = rbind(matrix(rnorm(d*tau),tau), matrix(rnorm(d*(n-tau),mu/sqrt(d),sigma), n-tau))
K = gaussiankernel(y)
a = kerseg1(n, K, pval.perm=TRUE, B=1000)

## Sequence 3: change in both the mean and variance happens on an interval.
d = 50
mu = 2
sigma = 0.5
tau1 = 25
tau2 = 35
```

```
n = 50
set.seed(1)
y1 = matrix(rnorm(d*tau1),tau1)
y2 = matrix(rnorm(d*(tau2-tau1),mu/sqrt(d),sigma), tau2-tau1)
y3 = matrix(rnorm(d*(n-tau2)), n-tau2)
y = rbind(y1, y2, y3)
K = gaussiankernel(y)
a = kerseg2(n, K, pval.perm=TRUE, B=1000)
```

---

kerseg1 *Kernel-based change-point detection for single change-point alternatives*

Description

This function finds a break point in the sequence where the underlying distribution changes.

Usage

```
kerseg1(n, K, r1=1.2, r2=0.8, n0=0.05*n, n1=0.95*n,
   pval.appr=TRUE, skew.corr=TRUE, pval.perm=FALSE, B=100)
```

Arguments

n	The number of observations in the sequence.
K	The kernel matrix of observations in the sequence.
r1	The constant in the test statistics $\mathrm{Z}_{W,r1}(t)$.
r2	The constant in the test statistics $\mathrm{Z}_{W,r2}(t)$.
n0	The starting index to be considered as a candidate for the change-point.
n1	The ending index to be considered as a candidate for the change-point.
pval.appr	If it is TRUE, the function outputs the p-value approximation based on asymptotic properties.
skew.corr	This argument is useful only when pval.appr=TRUE. If skew.corr is TRUE, the p-value approximation would incorporate skewness correction.
pval.perm	If it is TRUE, the function outputs the p-value from doing B permutations, where B is another argument that you can specify. Doing permutation could be time consuming, so use this argument with caution as it may take a long time to finish the permutation.
B	This argument is useful only when pval.perm=TRUE. The default value for B is 100.

Value

Returns a list `stat` containing the each scan statistic, `tauhat` containing the estimated location of change-point, `appr` containing the approximated p-values of the fast tests when argument 'pval.appr' is TRUE, and `perm` containing the permutation p-values of the fast tests and GKCP when argument 'pval.perm' is TRUE. See below for more details.

`seq`	A vector of each scan statistic (standardized counts).
`Zmax`	The test statistics (maximum of the scan statistics).
`tauhat`	An estimate of the location of the change-point.
`fGKCP1_bon`	The p-value of $fGKCP_1$ obtained by the Bonferroni procedure.
`fGKCP1_sim`	The p-value of $fGKCP_1$ obtained by the Simes procedure.
`fGKCP2_bon`	The p-value of $fGKCP_2$ obtained by the Bonferroni procedure.
`fGKCP2_sim`	The p-value of $fGKCP_2$ obtained by the Simes procedure.
`GKCP`	The p-value of GKCP obtained by the random permutation.

See Also

`kerSeg-package`, `kerseg1`, `gaussiankernel`, `kerseg2`

Examples

```
## Sequence 1: change in the mean in the middle of the sequence.
d = 50
mu = 2
tau = 25
n = 50
set.seed(1)
y = rbind(matrix(rnorm(d*tau),tau), matrix(rnorm(d*(n-tau),mu/sqrt(d)), n-tau))
K = gaussiankernel(y) # Gaussian kernel matrix
a = kerseg1(n, K, pval.perm=TRUE, B=1000)
# output results based on the permutation and the asymptotic results.
# the scan statistics can be found in a$scanZ.
# the approximated p-values can be found in a$appr.
# the permutation p-values can be found in a$perm.

## Sequence 2: change in both the mean and variance away from the middle of the sequence.
d = 50
mu = 2
sigma = 0.7
tau = 35
n = 50
set.seed(1)
y = rbind(matrix(rnorm(d*tau),tau), matrix(rnorm(d*(n-tau),mu/sqrt(d),sigma), n-tau))
K = gaussiankernel(y)
a = kerseg1(n, K, pval.perm=TRUE, B=1000)
```

`kerseg2` *Kernel-based change-point detection for changed-interval alternatives*

Description

This function finds an interval in the sequence where their underlying distribution differs from the rest of the sequence.

Usage

```
kerseg2(n, K, r1=1.2, r2=0.8, l0=0.05*n, l1=0.95*n,
   pval.appr=TRUE, skew.corr=TRUE, pval.perm=FALSE, B=100)
```

Arguments

- `n` The number of observations in the sequence.
- `K` The kernel matrix of observations in the sequence.
- `r1` The constant in the test statistics $\mathrm{Z}_{W,r1}(t_1,t_2)$.
- `r2` The constant in the test statistics $\mathrm{Z}_{W,r2}(t_1,t_2)$.
- `l0` The minimum length of the interval to be considered as a changed interval.
- `l1` The maximum length of the interval to be considered as a changed interval.
- `pval.appr` If it is TRUE, the function outputs the p-value approximation based on asymptotic properties.
- `skew.corr` This argument is useful only when pval.appr=TRUE. If skew.corr is TRUE, the p-value approximation would incorporate skewness correction.
- `pval.perm` If it is TRUE, the function outputs the p-value from doing B permutations, where B is another argument that you can specify. Doing permutation could be time consuming, so use this argument with caution as it may take a long time to finish the permutation.
- `B` This argument is useful only when pval.perm=TRUE. The default value for B is 100.

Value

Returns a list `stat` containing the each scan statistic, `tauhat` containing the estimated changed-interval, `appr` containing the approximated p-values of the fast tests when argument 'pval.appr' is TRUE, and `perm` containing the permutation p-values of the fast tests and GKCP when argument 'pval.perm' is TRUE. See below for more details.

- `seq` A matrix of each scan statistic (standardized counts).
- `Zmax` The test statistics (maximum of the scan statistics).
- `tauhat` An estimate of the two ends of the changed-interval.
- `fGKCP1_bon` The p-value of fGKCP_1 obtained by the Bonferroni procedure.

`fGKCP1_sim`	The p-value of fGKCP_1 obtained by the Simes procedure.
`fGKCP2_bon`	The p-value of fGKCP_2 obtained by the Bonferroni procedure.
`fGKCP2_sim`	The p-value of fGKCP_2 obtained by the Simes procedure.
`GKCP`	The p-value of GKCP obtained by the random permutation.

See Also

kerSeg-package, kerseg2, gaussiankernel, kerseg1

Examples

```
## Sequence 3: change in both the mean and variance happens on an interval.
d = 50
mu = 2
sigma = 0.5
tau1 = 25
tau2 = 35
n = 50
set.seed(1)
y1 = matrix(rnorm(d*tau1),tau1)
y2 = matrix(rnorm(d*(tau2-tau1),mu/sqrt(d),sigma), tau2-tau1)
y3 = matrix(rnorm(d*(n-tau2)), n-tau2)
y = rbind(y1, y2, y3)
K = gaussiankernel(y)
a = kerseg2(n, K, pval.perm=TRUE, B=1000)
```

---

skew — *Compute some components utilized in the third moment fomulas.*

Description

This function provides some components used in the third moment fomulas.

Usage

```
skew(K, Rtemp, Rtemp2, R0, R2)
```

Arguments

`K`	A kernel matrix of observations in the sequence.
`Rtemp`	A numeric vector of $k_{i.}$, the sum of kernel values for each row i.
`Rtemp2`	A numeric vector, the sum of squared kernel values for each row i.
`R0`	The term R_0, defined in the paper.
`R2`	The term R_2, defined in the paper.

Value

Returns a list of components used in the third moment fomulas.

statint	*Compute the test statistics, D and W, for the changed-interval alternatives.*

Description

This function provides the test statistics, D(t_1, t_2), W(t_1, t_2), and the weighted W(t_1, t_2) for the changed-interval alternatives.

Usage

```
statint(K, Rtemp, R0, r1, r2)
```

Arguments

K	A kernel matrix of observations in the sequence.
Rtemp	A numeric vector of $k_{i.}$, the sum of kernel values for each row i.
R0	The term R_0, defined in the paper.
r1	The constant in the test statistics Z$_{W,r1}(t_1, t_2)$.
r2	The constant in the test statistics Z$_{W,r2}(t_1, t_2)$.

Value

Returns a list of test statistics, D(t_1, t_2), W(t_1, t_2), W$_{r1}(t_1, t_2)$, and W$_{r2}(t_1, t_2)$.

Examples

```
## Sequence : change in the mean in the middle of the sequence.
d = 50
mu = 2
tau = 50
n = 100
set.seed(1)
y = rbind(matrix(rnorm(d*tau),tau), matrix(rnorm(d*(n-tau),mu/sqrt(d)), n-tau))
K = gaussiankernel(y) # Gaussian kernel matrix
R_temp = rowSums(K)
R0 = sum(K)
a = statint(K, R_temp, R0, r1=1.2, r2=0.8)
```

Index