Package 'OLCPM'

February 27, 2023

Type Package

Title Online Change Point Detection for Matrix-Valued Time Series

Version 0.1.0

Author Yong He [aut],
Xinbing Kong [aut],
Lorenzo Trapani [aut],
Long Yu [aut, cre]

Maintainer Long Yu <yulong@mail.shufe.edu.cn>

Description We provide two algorithms for monitoring change points with online matrix-valued time series, under the assumption of a two-way factor structure. The algorithms are based on different calculations of the second moment matrices. One is based on stacking the columns of matrix observations, while another is by a more delicate projected approach. A well-known fact is that, in the presence of a change point, a factor model can be rewritten as a model with a larger number of common factors. In turn, this entails that, in the presence of a change point, the number of spiked eigenvalues in the second moment matrix of the data increases. Based on this, we propose two families of procedures - one based on the fluctuations of partial sums, and one based on extreme value theory - to monitor whether the first non-spiked eigenvalue diverges after a point in time in the monitoring horizon, thereby indicating the presence of a change point. See more details in He et al. (2021)<arXiv:2112.13479>.

License GPL-2 | GPL-3

Encoding UTF-8

Imports LaplacesDemon, RSpectra

Depends R (>= 3.5.0)

NeedsCompilation no

Repository CRAN

Date/Publication 2023-02-27 08:52:30 UTC

R topics documented:

`gen.data` *generate data*

Description

This function generates matrix-valued time series under a two-way factor structure with/without a change point.

Usage

```
gen.data(Sample_T,p1,p2,k1,k2,tau=0.5,change=0,pp=0.3,a=0,cc=0)
```

Arguments

`Sample_T`	positive integer indicating the length of series
`p1`	positive integer indicating the row dimension
`p2`	positive integer indicating the column dimension
`k1`	positive integer indicating the number of row factors
`k2`	positive integer indicating the number of column factors
`tau`	a real number in $(0, 1)$, indicating the location of change point, i.e., $(\tau \times T)$
`change`	the type of change, taking 0 for no change point, taking 1 for the case that the last column of R changes, taking other values for the case that a new row factor occurs.
`pp`	a number in $(0, 1]$, indicating the magnitude of the break. When `change=1`, pp is the proportion of entries in the last column of R changing. When change is not equal to 0 or 1, pp is the proportion of non-zero entries in the new factor loading.
`a`	a number in $[0, \min(p_1, p_2))$, indicating the cross-sectional correlations of the idiosyncratic errors.
`cc`	a number in $[0, 1)$, indicating the AR(1) coefficient of the factor and error processes.

Details

See the paper He et al. (2021).

Value

The return value is a $T \times p_1 \times p_2$ array.

Author(s)

Yong He, Xinbing Kong, Lorenzo Trapani, Long Yu

References

He Y, Kong X, Trapani L, & Yu L(2021). Online change-point detection for matrix-valued time series with latent two-way factor structure. *arXiv preprint*, arXiv:2112.13479.

Examples

```
## generate data
k1=3
k2=3
epsilon=0.05
Sample_T=50
p1=40
p2=20
kmax=8
p=8
m=p2

# generate data
Y=gen.data(Sample_T,p1,p2,k1,k2,tau=0.5,change=1,pp=0.3)
```

---

gen.psi.tau.flat *calculate eigenvalue series by "flat" method*

Description

This function calculates the rolling eigenvalue series for the monitoring process, based on the "flat" version of sample covanriance matrix.

Usage

```
gen.psi.tau.flat(Y, r, m, delta, p)
```

Arguments

Y	the observed $T \times p_1 \times p_2$ array. T is the sample size, p_1 and p_2 are the row and column dimensions, respectively.
r	a positive integer determining which eigenvalue to monitor. $r = 1$ for the largest eigenvalue.
m	a positive integer (> 1) indicating the bandwidth of the rolling windom.

`delta` a number in $(0, 1)$ indicating the rescaling parameter for the eigenvalue. The default approach to calcualte delta is in the paper He et al. (2021).

`p` a positive integer indicating the order of the transformation function $g(x) = |x|^p$. Motivated by the paper, p should be chosen according to the moments of the data; see more details in He et al. (2021).

Details

The rolling eigenvalue series will start at the stage $m + 1$, with length $T - m$.

Value

When $T > m$, the return value is a $(T - m) \times 3$ matrix, containing the original, rescaled, and the transformed eigenvalue series. Otherwise, the function outputs an error.

Author(s)

Yong He, Xinbing Kong, Lorenzo Trapani, Long Yu

References

He Y, Kong X, Trapani L, & Yu L(2021). Online change-point detection for matrix-valued time series with latent two-way factor structure. *arXiv preprint*, arXiv:2112.13479.

Examples

```
## generate data
k1=3
k2=3
epsilon=0.05
Sample_T=50
p1=40
p2=20
kmax=8
p=8
m=p2

# generate data
Y=gen.data(Sample_T,p1,p2,k1,k2,tau=0.5,change=1,pp=0.3)

# calculate delta
temp=log(p1)/log(m*p2)
delta=epsilon*(temp<=0.5)+(epsilon+1-1/(2*temp))*(temp>0.5)

# calculate psi.tau
psi2=gen.psi.tau.flat(Y,k1+1,m,delta,p)
```

`gen.psi.tau.proj` *calculate eigenvalue series by projected method*

Description

This function calculates the rolling eigenvalue series for the monitoring process, based on the projected version of sample covanriance matrix.

Usage

```
gen.psi.tau.proj(Y, r, m, delta, p, kmax)
```

Arguments

`Y`	the observed $T \times p_1 \times p_2$ array. T is the sample size, p_1 and p_2 are the row and column dimensions, respectively.
`r`	a positive integer determining which eigenvalue to monitor. $r = 1$ for the largest eigenvalue.
`m`	a positive integer (> 1) indicating the bandwidth of the rolling windom.
`delta`	a number in $(0, 1)$ indicating the rescaling parameter for the eigenvalue. The default approach to calcualte delta is in the paper He et al. (2021).
`p`	a positive integer indicating the order of the transformation function $g(x) = \|x\|^p$. Motivated by the paper, p should be chosen according to the moments of the data; see more details in He et al. (2021).
`kmax`	a positive integer indicating the column number of the projection matrix, should be larger than 0 but smaller than p_2.

Details

The rolling eigenvalue series will start at the stage $m + 1$, with length $T - m$.

Value

When $T > m$, the return value is a $(T - m) \times 3$ matrix, containing the original, rescaled, and the transformed eigenvalue series. Otherwise, the function outputs an error.

Author(s)

Yong He, Xinbing Kong, Lorenzo Trapani, Long Yu

References

He Y, Kong X, Trapani L, & Yu L(2021). Online change-point detection for matrix-valued time series with latent two-way factor structure. *arXiv preprint*, arXiv:2112.13479.

Examples

```
## generate data
k1=3
k2=3
epsilon=0.05
Sample_T=50
p1=40
p2=20
kmax=8
p=8
m=p2

# generate data
Y=gen.data(Sample_T,p1,p2,k1,k2,tau=0.5,change=1,pp=0.3)

# calculate delta
temp=log(p1)/log(m*p2)
delta=epsilon*(temp<=0.5)+(epsilon+1-1/(2*temp))*(temp>0.5)

# calculate psi.tau
psi2=gen.psi.tau.proj(Y,k1+1,m,delta,p,kmax)
```

`getcv` *calculate critical values*

Description

This function calculates critical values for the partial-sum statistic with $\eta = 0.5$ or the the worst-cas statistic.

Usage

```
getcv(Tm, alpha = 0.05, method = "ps")
```

Arguments

`Tm`	the length of the eigenvalue series, equal to $T - m$
`alpha`	significance level of the test, taking value from $(0, 1)$
`method`	"ps" for the partial-sum staistic, others for the worst-case statistic.

Details

For the partial-sum statistic, this function only works for $\eta = 0.5$. For other values of η, the critical value should be given by the user according to Horvath and Huskova (2004).

Value

The return value is a real number.

Author(s)

Yong He, Xinbing Kong, Lorenzo Trapani, Long Yu

References

Horváth L, Hušková M, Kokoszka P, et al (2004). Monitoring changes in linear models. Journal of statistical Planning and Inference, 126(1): 225-251.

He Y, Kong X, Trapani L, & Yu L(2021). Online change-point detection for matrix-valued time series with latent two-way factor structure. *arXiv preprint*, arXiv:2112.13479.

Examples

```
cv1=getcv(160,0.05,method="ps")
cv2=getcv(160,0.05,method="wc")
print(c(cv1,cv2))
```

`test.matrix.flat` *test change point for matrix-valued online time series-"flat" version*

Description

This function tests change point for matrix-valued online time series, under a two-way factor structure, using "flat" sample covariance matrix.

Usage

```
test.matrix.flat(Y, r, m, epsilon, p, decrease = 0, method = "ps", eta = 0.25,
  cv = 2.3860)
```

Arguments

`Y`	data, a $T \times p_1 \times p_2$ array.
`r`	a positive integer indicating which eigenvalue to monitor. $r = 1$ for the largest eigenvalue.
`m`	a positive integer (> 1) indicating the bandwidth of the rolling windom.
`epsilon`	the rescaling parameter taking value in $(0, 1)$; see He et al. (2021).
`p`	a positive integer indicating the order of the transformation function $g(x) = \|x\|^p$. Motivated by He et al. (2021), p should be chosen according to the moments of the data; see more details in He et al. (2021).
`decrease`	a logical value. If `decrease=1`, testing the decrease of factor number.
`method`	indicating the test statistic, "ps" for the partial-sum method; others for the worst-case method.
`eta`	a number between $[0, 1)$, indicating the parameter η used in the partial-sum statistic.

cv	critical value, related to the significance level and test statistic. For worst-case statistic, calculate cv by function get.cv. For partial-sum statistic, the default cv only works for $\eta = 0.25$ or $\eta = 0.75$. For other η, see Horváth et al. (2004).

Details

See He et al. (2021).

Value

The return value is a list. In this list, it contains the following:

test	a logical value. 1 indicating the existence of change point, 0 indicating no change point.
loc	an integer larger than m, indicating the location of change point; or NA when no change point is reported

Author(s)

Yong He, Xinbing Kong, Lorenzo Trapani, Long Yu

References

Horváth L, Hušková M, Kokoszka P, et al (2004). Monitoring changes in linear models. Journal of statistical Planning and Inference, 126(1): 225-251.

He Y, Kong X, Trapani L, & Yu L(2021). Online change-point detection for matrix-valued time series with latent two-way factor structure. *arXiv preprint*, arXiv:2112.13479.

Examples

```
k1=3
k2=3
epsilon=0.05
Sample_T=50
p1=40
p2=20
kmax=8
p=8
m=p2

# generate data
Y=gen.data(Sample_T,p1,p2,k1,k2,tau=0.5,change=1,pp=0.3)

# calculate delta
temp=log(p1)/log(m*p2)
delta=epsilon*(temp<=0.5)+(epsilon+1-1/(2*temp))*(temp>0.5)


# calculate cv for eta=0.5 and "wc"
Tm=Sample_T-m
cv1=getcv(Tm,0.05,method="ps")
```

```
cv2=getcv(Tm,0.05,method="wc")
print(c(cv1,cv2))


## test with Y, without projection
re1=test.matrix.flat(Y,k1+1,m,epsilon,p,0,method="ps",eta=0.25)
print(re1)

re2=test.matrix.flat(Y,k1+1,m,epsilon,p,0,method="ps",eta=0.75)
print(re2)

re3=test.matrix.flat(Y,k1+1,m,epsilon,p,0,method="ps",eta=0.5,cv=cv1)
print(re3)

re4=test.matrix.flat(Y,k1+1,m,epsilon,p,0,method="wc",eta=0.5,cv=cv2)
print(re4)
```

`test.matrix.proj` *test change point for matrix-valued online time series-projected version*

Description

This function tests change point for matrix-valued online time series, under a two-way factor structure, using projected sample covariance matrix

Usage

```
test.matrix.proj(Y, r, m, epsilon, p, kmax, decrease = 0, method = "ps",

 eta = 0.25, cv = 2.3860)
```

Arguments

`Y`	data, a $T \times p_1 \times p_2$ array.
`r`	a positive integer indicating which eigenvalue to monitor. $r = 1$ for the largest eigenvalue.
`m`	a positive integer (> 1) indicating the bandwidth of the rolling windom.
`epsilon`	the rescaling parameter taking value in $(0, 1)$; see He et al. (2021).
`p`	a positive integer indicating the order of the transformation function $g(x) = \|x\|^p$. Motivated by He et al. (2021), p should be chosen according to the moments of the data; see more details in He et al. (2021).
`kmax`	a positive number determining the column number of the projection matrix, should be larger than 0 but smaller than p_2.
`decrease`	a logical value. If `decrease=1`, testing the decrease of factor number.

method	indicating the test statistic, "ps" for the partial-sum method; others for the worst-case method.
eta	a number between $[0, 1)$, indicating the parameter η used in the partial-sum statistic.
cv	critical value, related to the significance level and test statistic. For worst-case statistic, calculate cv by function get.cv. For partial-sum statistic, the default cv only works for $\eta = 0.25$ or $\eta = 0.75$. For other η, see Horváth et al. (2004).

Details

See He et al. (2021).

Value

The return value is a list. In this list, it contains the following:

test	a logical value. 1 indicating the existence of change point, 0 indicating no change point.
loc	an integer larger than m, indicating the location of change point; or NA when no change point is reported

Author(s)

Yong He, Xinbing Kong, Lorenzo Trapani, Long Yu

References

Horváth L, Hušková M, Kokoszka P, et al (2004). Monitoring changes in linear models. Journal of statistical Planning and Inference, 126(1): 225-251.

He Y, Kong X, Trapani L, & Yu L(2021). Online change-point detection for matrix-valued time series with latent two-way factor structure. *arXiv preprint*, arXiv:2112.13479.

Examples

```
k1=3
k2=3
epsilon=0.05
Sample_T=50
p1=40
p2=20
kmax=8
p=8
m=p2

# generate data
Y=gen.data(Sample_T,p1,p2,k1,k2,tau=0.5,change=1,pp=0.3)

# calculate delta
temp=log(p1)/log(m*p2)
delta=epsilon*(temp<=0.5)+(epsilon+1-1/(2*temp))*(temp>0.5)
```

```
# calculate cv for eta=0.5 and "wc"
Tm=Sample_T-m
cv1=getcv(Tm,0.05,method="ps")
cv2=getcv(Tm,0.05,method="wc")
print(c(cv1,cv2))


## test with Y, projection
re1=test.matrix.proj(Y,k1+1,m,epsilon,p,kmax,0,method="ps",eta=0.25)
print(re1)

re2=test.matrix.proj(Y,k1+1,m,epsilon,p,kmax,0,method="ps",eta=0.75)
print(re2)

re3=test.matrix.proj(Y,k1+1,m,epsilon,p,kmax,0,method="ps",eta=0.5,cv=cv1)
print(re3)

re4=test.matrix.proj(Y,k1+1,m,epsilon,p,kmax,0,method="wc",eta=0.5,cv=cv2)
print(re4)
```

`test.matrix.psi` *test change point for matrix-valued online data with rolling eigenvalue series*

Description

This function tests change point for matrix-valued online time series, under a two-way factor structure, when the transformed eigenvalue series is given.

Usage

```
test.matrix.psi(m, psi, method = "ps", eta = 0.25, cv = 2.3860)
```

Arguments

`m`	a positive integer (> 1) indicating the bandwidth of the rolling windom.
`psi`	the transformed eigenvalue series, produced by gen.psi.tau.flat or gen.psi.tau.proj, with length $T - m$.
`method`	indicating the test statistic, "ps" for the partial-sum method, while others for the worst-case method.
`eta`	a number between $[0, 1)$, indicating the parameter η used in the partial-sum statistic.
`cv`	critical value, related to the significance level and test statistic. For worst-case statistic, calculate cv by function get.cv. For partial-sum statistic, the default cv only works for $\eta = 0.25$ or $\eta = 0.75$. For other η, see Horváth et al. (2004).

Details

See He et al. (2021).

Value

The return value is a list. In this list, it contains the following:

`test`	a logical value. 1 indicating the existence of change point, 0 indicating no change point.
`loc`	an integer larger than m, indicating the location of change point; or NA when no change point is reported

Author(s)

Yong He, Xinbing Kong, Lorenzo Trapani, Long Yu

References

Horváth L, Hušková M, Kokoszka P, et al (2004). Monitoring changes in linear models. Journal of statistical Planning and Inference, 126(1): 225-251.

He Y, Kong X, Trapani L, & Yu L(2021). Online change-point detection for matrix-valued time series with latent two-way factor structure. *arXiv preprint*, arXiv:2112.13479.

Examples

```
k1=3
k2=3
epsilon=0.05
Sample_T=50
p1=40
p2=20
kmax=8
p=8
m=p2

# generate data
Y=gen.data(Sample_T,p1,p2,k1,k2,tau=0.5,change=1,pp=0.3)

# calculate delta
temp=log(p1)/log(m*p2)
delta=epsilon*(temp<=0.5)+(epsilon+1-1/(2*temp))*(temp>0.5)

# calculate psi.tau
psi1=gen.psi.tau.proj(Y,k1+1,m,delta,p,kmax)
psi2=gen.psi.tau.flat(Y,k1+1,m,delta,p)

# calculate cv for eta=0.5 and "wc"
Tm=Sample_T-m
cv1=getcv(Tm,0.05,method="ps")
cv2=getcv(Tm,0.05,method="wc")
print(c(cv1,cv2))
```

```
# test with psi1
re1=test.matrix.psi(m,psi1[,3],method="ps",eta=0.25)
print(re1)

re2=test.matrix.psi(m,psi1[,3],method="ps",eta=0.75)
print(re2)

re3=test.matrix.psi(m,psi1[,3],method="ps",eta=0.5,cv=cv1)
print(re3)

re4=test.matrix.psi(m,psi1[,3],method="wc",cv=cv2)
print(re4)

# test with psi2
re1=test.matrix.psi(m,psi2[,3],method="ps",eta=0.25)
print(re1)

re2=test.matrix.psi(m,psi2[,3],method="ps",eta=0.75)
print(re2)

re3=test.matrix.psi(m,psi2[,3],method="ps",eta=0.5,cv=cv1)
print(re3)

re4=test.matrix.psi(m,psi2[,3],method="wc",cv=cv2)
print(re4)
```

Index