Package 'PCA4TS'

October 12, 2022

Title Segmenting Multiple Time Series by Contemporaneous Linear Transformation

Version 0.1

Author Jinyuan Chang, Bin Guo and Qiwei Yao

Maintainer Bin Guo <guobin1987@pku.edu.cn>

Description To seek for a contemporaneous linear transformation for a multivariate time series such that the transformed series is segmented into several lower-dimensional subseries, and those subseries are uncorrelated with each other both contemporaneously and serially.

Depends R (>= 3.1.1)

Imports tseries

License GPL-2

LazyData true

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-08-05 17:21:32

R topics documented:

permutationFDR *Permutation Using the FDR Method*

Description

The permutation is determined by grouping the components of a multivariate series X into q groups, where q and the cardinal numbers of those groups are also unknown.

Usage

```
permutationFDR(X, Vol = FALSE, Beta, m = NULL)
```

Arguments

X	a data matrix used to find the grouping mechanism with n rows and p columns, where n is the sample size and p is the dimension of the time series.
Vol	logical. If FALSE (the default), then prewhiten each series by fitting a univariate AR model with the order between 0 and 5 determined by AIC. If TRUE, then prewhiten each volatility process using GARCH(1,1) model.
Beta	the error rate in FDR
m	a positive constant used to calculate the maximum cross correlation over the lags between $-m$ and m. If m is not specified, the default constant $10 * log10(n/p)$ will be used.

Details

See Chang et al. (2014) for the permutation step and more information.

Value

An object of class "permutationFDR" is a list containing the following components:

NoGroups	number of groups with at least two components series
Nos_of_Members	number of members in each of groups with at least two members
Groups	indices of components in each of groups with at least two members
Pvalues	Pvalue for multiple test H_0 for each of $p(p-1)/2$ pairs in ascending order
NoConnectedPairs	number of connected pairs
Xpre	the prewhitened data with $n - R$ rows and p columns

Note

This is the second step for segmentation by grouping the transformed time series. The first step is to seek for a contemporaneous linear transformation of the original series, see segmentTS.

Author(s)

Jinyuan Chang, Bin Guo and Qiwei Yao

References

Chang, J., Guo, B. and Yao, Q. (2014). Segmenting Multiple Time Series by Contemporaneous Linear Transformation: PCA for Time Series. Available at http://arxiv.org/abs/1410.2323

See Also

segmentTS, permutationMax

Examples

```
## Example 1 (Example 5 of Chang et al.(2014)).
## p=6, x_t consists of 3 independent subseries with 3, 2 and 1 components.

p=6;n=1500
# Generate x_t
X=mat.or.vec(p,n)
x=arima.sim(model=list(ar=c(0.5, 0.3), ma=c(-0.9, 0.3, 1.2,1.3)),n=n+2,sd=1)
for(i in 1:3) X[i,]=x[i:(n+i-1)]
x=arima.sim(model=list(ar=c(0.8,-0.5),ma=c(1,0.8,1.8) ),n=n+1,sd=1)
for(i in 4:5) X[i,]=x[(i-3):(n+i-4)]
x=arima.sim(model=list(ar=c(-0.7, -0.5), ma=c(-1, -0.8)),n=n,sd=1)
X[6,]=x
# Generate y_t
A=matrix(runif(p*p, -3, 3), ncol=p)
Y=A%*%X
Y=t(Y)
Trans=segmentTS(Y, k0=5)
# The transformed series z_t
Z=Trans$X
# Plot the cross correlogram of x_t and y_t
Z=data.frame(Z)
names(Z)=c("Z1","Z2","Z3","Z4","Z5","Z6")
# The cross correlogram of z_t shows 3-2-1 block pattern
acfZ=acf(Z, plot=FALSE)
plot(acfZ, max.mfrow=6, xlab='', ylab='',  mar=c(1.8,1.3,1.6,0.5),
    oma=c(1,1.2,1.2,1), mgp=c(0.8,0.4,0),cex.main=1)
# Identify the permutation mechanism
permutation=permutationFDR(Z,Beta=10^(-8))
permutation$Groups


## Example 2 (Example 6 of Chang et al.(2014)).
## p=20, x_t consists of 5 independent subseries with 6, 5, 4, 3 and 2 components.

p=20;n=3000
# Generate x_t
X=mat.or.vec(p,n)
x=arima.sim(model=list(ar=c(0.5, 0.3), ma=c(-0.9, 0.3, 1.2,1.3)),n.start=500,n=n+5,sd=1)
```

```
for(i in 1:6) X[i,]=x[i:(n+i-1)]
x=arima.sim(model=list(ar=c(-0.4,0.5),ma=c(1,0.8,1.5,1.8)),n.start=500,n=n+4,sd=1)
for(i in 7:11) X[i,]=x[(i-6):(n+i-7)]
x=arima.sim(model=list(ar=c(0.85,-0.3),ma=c(1,0.5,1.2)), n.start=500,n=n+3,sd=1)
for(i in 12:15) X[i,]=x[(i-11):(n+i-12)]
x=arima.sim(model=list(ar=c(0.8,-0.5),ma=c(1,0.8,1.8)),n.start=500,n=n+2,sd=1)
for(i in 16:18) X[i,]=x[(i-15):(n+i-16)]
x=arima.sim(model=list(ar=c(-0.7, -0.5), ma=c(-1, -0.8)),n.start=500,n=n+1,sd=1)
for(i in 19:20) X[i,]=x[(i-18):(n+i-19)]
# Generate y_t
A=matrix(runif(p*p, -3, 3), ncol=p)
Y=A%*%X
Y=t(Y)
Trans=segmentTS(Y, k0=5)
# The transformed series z_t
Z=Trans$X
# Identify the permutation mechanism
permutation=permutationFDR(Z,Beta=10^(-200))
permutation$Groups
```

`permutationMax` *Permutation Using the Maximum Cross Correlation Method*

Description

The permutation is determined by grouping the components of a multivariate series X into q groups, where q and the cardinal numbers of those groups are also unknown.

Usage

```
permutationMax(X, Vol = FALSE, m = NULL)
```

Arguments

`X`	a data matrix used to find the grouping mechanism with n rows and p columns, where n is the sample size and p is the dimension of the time series.
`Vol`	logical. If `FALSE` (the default), then prewhiten each series by fitting a univariate AR model with the order between 0 and 5 determined by AIC. If `TRUE`, then prewhiten each volatility process using GARCH(1,1) model.
`m`	a positive constant used to calculate the maximum cross correlation over the lags between $-m$ and m. If m is not specified, the default constant $10 * log10(n/p)$ will be used.

Details

See Chang et al. (2014) for the permutation step and more information.

Value

An object of class "permutationMax" is a list containing the following components:

`NoGroups`	number of groups with at least two components series
`Nos_of_Members`	number of members in each of groups with at least two members
`Groups`	indices of components in each of groups with at least two members
`maxcorr`	maximum correlation (over lags) of $p(p-1)/2$ pairs in descending order
`corrRatio`	ratios of successive values from maxcorr
`NoConnectedPairs`	number of connected pairs
`Xpre`	the prewhitened data with $n - R$ rows and p columns

Note

This is the second step for segmentation by grouping the transformed time series. The first step is to seek for a contemporaneous linear transformation of the original series, see `segmentTS`.

Author(s)

Jinyuan Chang, Bin Guo and Qiwei Yao

References

Chang, J., Guo, B. and Yao, Q. (2014). Segmenting Multiple Time Series by Contemporaneous Linear Transformation: PCA for Time Series. Available at http://arxiv.org/abs/1410.2323

See Also

`segmentTS`, `permutationFDR`

Examples

```
## Example 1 (Example 5 of Chang et al.(2014)).
## p=6, x_t consists of 3 independent subseries with 3, 2 and 1 components.

p=6;n=1500
# Generate x_t
X=mat.or.vec(p,n)
x=arima.sim(model=list(ar=c(0.5, 0.3), ma=c(-0.9, 0.3, 1.2,1.3)),n=n+2,sd=1)
for(i in 1:3) X[i,]=x[i:(n+i-1)]
x=arima.sim(model=list(ar=c(0.8,-0.5),ma=c(1,0.8,1.8) ),n=n+1,sd=1)
for(i in 4:5) X[i,]=x[(i-3):(n+i-4)]
x=arima.sim(model=list(ar=c(-0.7, -0.5), ma=c(-1, -0.8)),n=n,sd=1)
X[6,]=x
# Generate y_t
A=matrix(runif(p*p, -3, 3), ncol=p)
Y=A%*%X
Y=t(Y)
Trans=segmentTS(Y, k0=5)
```

```
# The transformed series z_t
Z=Trans$X
# Plot the cross correlogram of x_t and y_t
Z=data.frame(Z)
names(Z)=c("Z1","Z2","Z3","Z4","Z5","Z6")
# The cross correlogram of z_t shows 3-2-1 block pattern
acfZ=acf(Z, plot=FALSE)
plot(acfZ, max.mfrow=6, xlab='', ylab='',  mar=c(1.8,1.3,1.6,0.5),
     oma=c(1,1.2,1.2,1), mgp=c(0.8,0.4,0),cex.main=1)
# Identify the permutation mechanism
permutation=permutationMax(Z)
permutation$Groups


## Example 2 (Example 6 of Chang et al.(2014)).
## p=20, x_t consists of 5 independent subseries with 6, 5, 4, 3 and 2 components.

p=20;n=3000
# Generate x_t
X=mat.or.vec(p,n)
x=arima.sim(model=list(ar=c(0.5, 0.3), ma=c(-0.9, 0.3, 1.2,1.3)),n.start=500,n=n+5,sd=1)
for(i in 1:6) X[i,]=x[i:(n+i-1)]
x=arima.sim(model=list(ar=c(-0.4,0.5),ma=c(1,0.8,1.5,1.8)),n.start=500,n=n+4,sd=1)
for(i in 7:11) X[i,]=x[(i-6):(n+i-7)]
x=arima.sim(model=list(ar=c(0.85,-0.3),ma=c(1,0.5,1.2)), n.start=500,n=n+3,sd=1)
for(i in 12:15) X[i,]=x[(i-11):(n+i-12)]
x=arima.sim(model=list(ar=c(0.8,-0.5),ma=c(1,0.8,1.8)),n.start=500,n=n+2,sd=1)
for(i in 16:18) X[i,]=x[(i-15):(n+i-16)]
x=arima.sim(model=list(ar=c(-0.7, -0.5), ma=c(-1, -0.8)),n.start=500,n=n+1,sd=1)
for(i in 19:20) X[i,]=x[(i-18):(n+i-19)]
# Generate y_t
A=matrix(runif(p*p, -3, 3), ncol=p)
Y=A%*%X
Y=t(Y)
Trans=segmentTS(Y, k0=5)
# The transformed series z_t
Z=Trans$X
# Identify the permutation mechanism
permutation=permutationMax(Z)
permutation$Groups
```

`returns` *Daily returns of six stocks*

Description

The data contains the daily returns of the stocks of Bank of America Corporation, Dell Inc., JP-Morgan Chase Co., FedEx Corporation, McDonald's Corp. and American International Group in 2 January 2002 – 10 July 2008.

Usage

```
data("returns")
```

Examples

```
data(returns)

# Time series plots of the returns

Y=returns
names(Y)=c("Y1","Y2","Y3","Y4","Y5","Y6")
location=c(1,253,505,757,1009,1260,1511)
year=c(2002,2003,2004,2005,2006,2007,2008)

par(mfrow=c(6,1),mar=c(2,2,0.5,1),cex=0.5, mgp = c(1.4, 0.6, 0))
plot(Y[,1],type='l',lty=1,xlab='',ylab='',col="blue",xaxt="n")
axis(1, at=location, labels=year, col.axis='brown')
plot(Y[,2],type='l',lty=1,xlab='',ylab='',col="blue",xaxt="n")
axis(1, at=location, labels=year, col.axis='brown')
plot(Y[,3],type='l',lty=1,xlab='',ylab='',col="blue",xaxt="n")
axis(1, at=location, labels=year, col.axis='brown')
plot(Y[,4],type='l',lty=1,xlab='',ylab='',col="blue",xaxt="n")
axis(1, at=location, labels=year, col.axis='brown')
plot(Y[,5],type='l',lty=1,xlab='',ylab='',col="blue",xaxt="n")
axis(1, at=location, labels=year, col.axis='brown')
plot(Y[,6],type='l',lty=1,xlab='',ylab='',col="blue",xaxt="n")
axis(1, at=location, labels=year, col.axis='brown')
```

---

segmentTS *Segment Multivariate Time Series*

Description

Calculate linear transformation of the p-variate time series y_t such that the transformed series x_t=By_t is segmented into several lower-dimensional subseries, and those subseries are uncorrelated with each other both contemporaneously and serially.

Usage

```
segmentTS(Y, k0, thresh = FALSE, tuning.vec = 2, K = 5)
```

Arguments

Y	a data matrix with n rows and p columns, where n is the sample size and p is the dimension of the time series.
k0	a positive integer specified to calculate Wy. See (2.5) in Chang et al. (2014).

`thresh` logical. If `FALSE` (the default), no thresholding will be applied. If `TRUE`, a thresholding method will be applied first to estimate Wy, see (3.4) and (3.5) in Chang et al. (2014).

`tuning.vec` the value of thresholding parameter λ. The thresholding level is specified by

$$u = \lambda(logp/n)^(1/2).$$

Default value is 2. If `tuning.vec` is a vector, then a cross validation method proposed in Cai and Liu (2011) will be used to choose the best tuning parameter.

`K` the number of folders used in the cross validation, the default is 5. It is required when `thresh` is `TRUE`.

Details

When p is small, thresholding is not required. However, when p is large, it is necessary to use the thresholding method, see more information in Chang et al. (2014).

Value

An object of class "segmentTS" is a list containing the following components:

`B` the p by p transformation matrix such that x_t=By_t

`X` the transformed series with n rows and p columns

Note

This is the first step to transform the time series. The second step is grouping the transformed time series, see `permutationMax`, `permutationFDR`.

Author(s)

Jinyuan Chang, Bin Guo and Qiwei Yao

References

Chang, J., Guo, B. and Yao, Q. (2014). *Segmenting Multiple Time Series by Contemporaneous Linear Transformation: PCA for Time Series*. Available at `http://arxiv.org/abs/1410.2323`.

Cai, T. and Liu, W. (2011). *Adaptive thresholding for sparse covariance matrix estimation*. Journal of the American Statistical Association 106: 672-684.

See Also

`permutationMax`, `permutationFDR`.

Examples

```
## Example 1 (Example 5 of Chang et al.(2014)).
## p=6, x_t consists of 3 independent subseries with 3, 2 and 1 components.

p=6;n=1500
# Generate x_t
X=mat.or.vec(p,n)
x=arima.sim(model=list(ar=c(0.5, 0.3), ma=c(-0.9, 0.3, 1.2,1.3)),n=n+2,sd=1)
for(i in 1:3) X[i,]=x[i:(n+i-1)]
x=arima.sim(model=list(ar=c(0.8,-0.5),ma=c(1,0.8,1.8) ),n=n+1,sd=1)
for(i in 4:5) X[i,]=x[(i-3):(n+i-4)]
x=arima.sim(model=list(ar=c(-0.7, -0.5), ma=c(-1, -0.8)),n=n,sd=1)
X[6,]=x
# Generate y_t
A=matrix(runif(p*p, -3, 3), ncol=p)
Y=A%*%X
Y=t(Y)
Trans=segmentTS(Y, k0=5)
# The transformed series z_t
Z=Trans$X
# Plot the cross correlogram of z_t and y_t
Y=data.frame(Y);Z=data.frame(Z)
names(Y)=c("Y1","Y2","Y3","Y4","Y5","Y6")
names(Z)=c("Z1","Z2","Z3","Z4","Z5","Z6")
# The cross correlogram of y_t shows no block pattern
acfY=acf(Y)
# The cross correlogram of z_t shows 3-2-1 block pattern
acfZ=acf(Z)

## Example 2 (Example 6 of Chang et al.(2014)).
## p=20, x_t consists of 5 independent subseries with 6, 5, 4, 3 and 2 components.

p=20;n=3000
# Generate x_t
X=mat.or.vec(p,n)
x=arima.sim(model=list(ar=c(0.5, 0.3), ma=c(-0.9, 0.3, 1.2,1.3)),n.start=500,n=n+5,sd=1)
for(i in 1:6) X[i,]=x[i:(n+i-1)]
x=arima.sim(model=list(ar=c(-0.4,0.5),ma=c(1,0.8,1.5,1.8)),n.start=500,n=n+4,sd=1)
for(i in 7:11) X[i,]=x[(i-6):(n+i-7)]
x=arima.sim(model=list(ar=c(0.85,-0.3),ma=c(1,0.5,1.2)), n.start=500,n=n+3,sd=1)
for(i in 12:15) X[i,]=x[(i-11):(n+i-12)]
x=arima.sim(model=list(ar=c(0.8,-0.5),ma=c(1,0.8,1.8)),n.start=500,n=n+2,sd=1)
for(i in 16:18) X[i,]=x[(i-15):(n+i-16)]
x=arima.sim(model=list(ar=c(-0.7, -0.5), ma=c(-1, -0.8)),n.start=500,n=n+1,sd=1)
for(i in 19:20) X[i,]=x[(i-18):(n+i-19)]
# Generate y_t
A=matrix(runif(p*p, -3, 3), ncol=p)
Y=A%*%X
Y=t(Y)
Trans=segmentTS(Y, k0=5)
# The transformed series z_t
Z=Trans$X
```

```
# Plot the cross correlogram of x_t and y_t
Y=data.frame(Y);Z=data.frame(Z)
namesY=NULL;namesZ=NULL
for(i in 1:p)
{
   namesY=c(namesY,paste0("Y",i))
   namesZ=c(namesZ,paste0("Z",i))
}
names(Y)=namesY;names(Z)=namesZ
# The cross correlogram of y_t shows no block pattern
acfY=acf(Y, plot=FALSE)
plot(acfY, max.mfrow=6, xlab='', ylab='',  mar=c(1.8,1.3,1.6,0.5),
     oma=c(1,1.2,1.2,1), mgp=c(0.8,0.4,0),cex.main=1)
# The cross correlogram of z_t shows 6-5-4-3-2 block pattern
acfZ=acf(Z, plot=FALSE)
plot(acfZ, max.mfrow=6, xlab='', ylab='',  mar=c(1.8,1.3,1.6,0.5),
     oma=c(1,1.2,1.2,1), mgp=c(0.8,0.4,0),cex.main=1)
# Identify the permutation mechanism
permutation=permutationMax(Z)
permutation$Groups
```

`segmentVOL` *Segment Multivariate Volatility Processes*

Description

Calculate linear transformation of the p-variate volatility processes y_t such that the transformed volatility process x_t=By_t can be segmented into q lower-dimensional processes, and there exist no *conditional* cross correlations across those q processes.

Usage

```
segmentVOL(Y, k0)
```

Arguments

`Y`	a data matrix with n rows and p columns, where n is the sample size and p is the dimension of the time series.
`k0`	a positive integer specified to calculate Wy.

Value

An object of class "segmentVOL" is a list containing the following components:

`B`	the p by p transformation matrix such that x_t=By_t
`X`	the transformed series with n rows and p columns

Author(s)

Jinyuan Chang, Bin Guo and Qiwei Yao

References

Chang, J., Guo, B. and Yao, Q. (2014). *Segmenting Multiple Time Series by Contemporaneous Linear Transformation: PCA for Time Series*. Available at http://arxiv.org/abs/1410.2323.

See Also

segmentTS

Examples

```
## Example 7 of Chang et al.(2014)
## Segmenting the returns of the 6 stocks

require(tseries)
data(returns)
Y=returns
n=dim(Y)[1]; p=dim(Y)[2]
# Carry out the transformation procedure
Trans=segmentVOL(Y,5)
X_0=data.frame(Trans$X)
X_1=X_0
# The ACF plot of the residuals after prewhitening the transformed data by GARCH(1,1)
nanum=rep(0,p)
for(j in 1:p) {options( warn = -1 )
               t=garch(X_1[,j], order = c(1,1),trace=FALSE)
               X_1[,j]=t$residuals
               a=X_1[,j]
               nanum[j]=length(a[is.na(X_1[,j])]) }
X=X_1[(max(nanum)+1):n,]
colnames(X)=c("X1","X2","X3","X4","X5","X6")
t=acf(X,plot=FALSE)
plot(t, max.mfrow=6, xlab='', ylab='',  mar=c(1.8,1.3,1.6,0.5),
    oma=c(1,1.2,1.2,1), mgp=c(0.8,0.4,0),cex.main=1.0,ylim=c(0,1))
# Identify the permutation mechanism
permutation=permutationMax(X_0,Vol=TRUE)
permutation$Groups
options( warn = 0)
```

Index