Package: HDMFA (via r-universe)

September 17, 2024

Type Package

Title High-Dimensional Matrix Factor Analysis

Version 0.1.1

Author Yong He [aut], Changwei Zhao [aut], Ran Zhao [aut, cre]

Maintainer Ran Zhao <Zhaoran@mail.sdu.edu.cn>

Description High-dimensional matrix factor models have drawn much attention in view of the fact that observations are usually well structured to be an array such as in macroeconomics and finance. In addition, data often exhibit heavy-tails and thus it is also important to develop robust procedures. We aim to address this issue by replacing the least square loss with Huber loss function. We propose two algorithms to do robust factor analysis by considering the Huber loss. One is based on minimizing the Huber loss of the idiosyncratic error's Frobenius norm, which leads to a weighted iterative projection approach to compute and learn the parameters and thereby named as Robust-Matrix-Factor-Analysis (RMFA), see the details in He et al. (2023)<doi:10.1080/07350015.2023.2191676>. The other one is based on minimizing the element-wise Huber loss, which can be solved by an iterative Huber regression algorithm (IHR), see the details in He et al. (2023) <arXiv:2306.03317>. In this package, we also provide the algorithm for alpha-PCA by Chen & Fan (2021) <doi:10.1080/01621459.2021.1970569>, the Projected estimation (PE) method by Yu et al. (2022)<doi:10.1016/j.jeconom.2021.04.001>. In addition, the methods for determining the pair of factor numbers are also given.

License GPL-2 | GPL-3

Encoding UTF-8

Depends R (>= 4.0)

Imports MASS, RSpectra

NeedsCompilation no

Repository CRAN

Date/Publication 2024-01-20 07:32:46 UTC

Contents

`alpha_PCA` *Statistical Inference for High-Dimensional Matrix-Variate Factor Model*

Description

This function is to fit the matrix factor model via the α-PCA method by conducting eigen-analysis of a weighted average of the sample mean and the column (row) sample covariance matrix through a hyper-parameter α.

Usage

```
alpha_PCA(X, m1, m2, alpha = 0)
```

Arguments

`X`	Input an array with $T \times p_1 \times p_2$, where T is the sample size, p_1 is the the row dimension of each matrix observation and p_2 is the the column dimension of each matrix observation.
`m1`	A positive integer indicating the row factor numbers.
`m2`	A positive integer indicating the column factor numbers.
`alpha`	A hyper-parameter balancing the information of the first and second moments ($\alpha \geq -1$). The default is 0.

Details

For the matrix factor models, Chen & Fan (2021) propose an estimation procedure, i.e. α-PCA. The method aggregates the information in both first and second moments and extract it via a spectral method. In detail, for observations $\boldsymbol{X}_t, t = 1, 2, \cdots, T$, define

$$\hat{\boldsymbol{M}}_R = \frac{1}{p_1 p_2}\left((1+\alpha)\bar{\boldsymbol{X}}\bar{\boldsymbol{X}}^\top + \frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{X}_t - \bar{\boldsymbol{X}})(\boldsymbol{X}_t - \bar{\boldsymbol{X}})^\top\right),$$

$$\hat{\boldsymbol{M}}_C = \frac{1}{p_1 p_2}\left((1+\alpha)\bar{\boldsymbol{X}}^\top\bar{\boldsymbol{X}} + \frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{X}_t - \bar{\boldsymbol{X}})^\top(\boldsymbol{X}_t - \bar{\boldsymbol{X}})\right),$$

where $\alpha \in$ [-1,+∞], $\bar{\boldsymbol{X}} = \frac{1}{T}\sum_{t=1}^{T}\boldsymbol{X}_t$, $\frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{X}_t - \bar{\boldsymbol{X}})(\boldsymbol{X}_t - \bar{\boldsymbol{X}})^\top$ and $\frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{X}_t - \bar{\boldsymbol{X}})^\top(\boldsymbol{X}_t - \bar{\boldsymbol{X}})$ are the sample row and column covariance matrix, respectively. The loading matrices $\boldsymbol{R}$ and $\boldsymbol{C}$ are estimated as $\sqrt{p_1}$ times the top k_1 eigenvectors of $\hat{\boldsymbol{M}}_R$ and $\sqrt{p_2}$ times the top k_2 eigenvectors of $\hat{\boldsymbol{M}}_C$, respectively. For details, see Chen & Fan (2021).

Value

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

F	The estimated factor matrix of dimension $T \times m_1 \times m_2$.
R	The estimated row loading matrix of dimension $p_1 \times m_1$, satisfying $\boldsymbol{R}^\top\boldsymbol{R} = p_1\boldsymbol{I}_{m_1}$.
C	The estimated column loading matrix of dimension $p_2 \times m_2$, satisfying $\boldsymbol{C}^\top\boldsymbol{C} = p_2\boldsymbol{I}_{m_2}$.

Author(s)

Yong He, Changwei Zhao, Ran Zhao.

References

Chen, E. Y., & Fan, J. (2021). Statistical inference for high-dimensional matrix-variate factor models. Journal of the American Statistical Association, 1-18.

Examples

```
set.seed(11111)
T=20;p1=20;p2=20;k1=3;k2=3
R=matrix(runif(p1*k1,min=-1,max=1),p1,k1)
C=matrix(runif(p2*k2,min=-1,max=1),p2,k2)
X=array(0,c(T,p1,p2))
Y=X;E=Y
F=array(0,c(T,k1,k2))
for(t in 1:T){
  F[t,,]=matrix(rnorm(k1*k2),k1,k2)
  E[t,,]=matrix(rnorm(p1*p2),p1,p2)
  Y[t,,]=R%*%F[t,,]%*%t(C)
}
X=Y+E

#Estimate the factor matrices and loadings
fit=alpha_PCA(X, k1, k2, alpha = 0)
Rhat=fit$R
Chat=fit$C
Fhat=fit$F

#Estimate the common component
CC=array(0,c(T,p1,p2))
for (t in 1:T){
CC[t,,]=Rhat%*%Fhat[t,,]%*%t(Chat)
}
```

CC

KMHFA *Estimating the Pair of Factor Numbers via Eigenvalue Ratios or Rank Minimization.*

Description

The function is to estimate the pair of factor numbers via eigenvalue-ratio corresponding to RMFA method or rank minimization and eigenvalue-ratio corresponding to Iterative Huber Regression (IHR).

Usage

```
KMHFA(X, W1 = NULL, W2 = NULL, kmax, method, max_iter = 100, c = 1e-04, ep = 1e-04)
```

Arguments

X	Input an array with $T \times p_1 \times p_2$, where T is the sample size, p_1 is the the row dimension of each matrix observation and p_2 is the the column dimension of each matrix observation.
W1	Only if method="E_RM" or method="E_ER", the inital value of row loadings matrix. The default is NULL, which is randomly chosen and all entries from a standard normal distribution.
W2	Only if method="E_RM" or method="E_ER", the inital value of column loadings matrix. The default is NULL, which is randomly chosen and all entries from a standard normal distribution.
kmax	The user-supplied maximum factor numbers. Here it means the upper bound of the number of row factors and column factors.
method	Character string, specifying the type of the estimation method to be used. "P", the robust iterative eigenvalue-ratio based on RMFA "E_RM", the rank-minimization based on IHR "E_ER", the eigenvalue-ratio based on IHR
max_iter	Only if method="E_RM" or method="E_ER", the maximum number of iterations in the iterative Huber regression algorithm. The default is 100.
c	A constant to avoid vanishing denominators. The default is 10^{-4}.
ep	Only if method="E_RM" or method="E_ER", the stopping critetion parameter in the iterative Huber regression algorithm. The default is $10^{-4} \times Tp_1p_2$.

Details

If `method="P"`, the number of factors k_1 and k_2 are estimated by

$$\hat{k}_1 = \arg\max_{j \le k_{max}} \frac{\lambda_j(\boldsymbol{M}_c^w)}{\lambda_{j+1}(\boldsymbol{M}_c^w)}, \hat{k}_2 = \arg\max_{j \le k_{max}} \frac{\lambda_j(\boldsymbol{M}_r^w)}{\lambda_{j+1}(\boldsymbol{M}_r^w)},$$

where k_{max} is a predetermined value larger than k_1 and k_2. $\lambda_j(\cdot)$ is the j-th largest eigenvalue of a nonnegative definitive matrix. See the function `MHFA` for the definition of $\boldsymbol{M}_c^w$ and $\boldsymbol{M}_r^w$. For details, see He et al. (2023).

Define $D = \min(\sqrt{Tp_1}, \sqrt{Tp_2}, \sqrt{p_1 p_2})$,

$$\hat{\boldsymbol{\Sigma}}_1 = \frac{1}{T}\sum_{t=1}^{T} \hat{\boldsymbol{F}}_t \hat{\boldsymbol{F}}_t^\top, \hat{\boldsymbol{\Sigma}}_2 = \frac{1}{T}\sum_{t=1}^{T} \hat{\boldsymbol{F}}_t^\top \hat{\boldsymbol{F}}_t,$$

where $\hat{\boldsymbol{F}}_t, t = 1, \ldots, T$ is estimated by IHR under the number of factor is k_{max}.

If `method="E_RM"`, the number of factors k_1 and k_2 are estimated by

$$\hat{k}_1 = \sum_{i=1}^{k_{max}} I\left(\mathrm{diag}(\hat{\boldsymbol{\Sigma}}_1) > P_1\right), \hat{k}_2 = \sum_{j=1}^{k_{max}} I\left(\mathrm{diag}(\hat{\boldsymbol{\Sigma}}_2) > P_2\right),$$

where I is the indicator function. In practice, P_1 is set as $\max\left(\mathrm{diag}(\hat{\boldsymbol{\Sigma}}_1)\right) \cdot D^{-2/3}$, P_2 is set as $\max\left(\mathrm{diag}(\hat{\boldsymbol{\Sigma}}_2)\right) \cdot D^{-2/3}$.

If `method="E_ER"`, the number of factors k_1 and k_2 are estimated by

$$\hat{k}_1 = \arg\max_{i \le k_{max}} \frac{\lambda_i(\hat{\boldsymbol{\Sigma}}_1)}{\lambda_{i+1}(\hat{\boldsymbol{\Sigma}}_1) + cD^{-2}}, \hat{k}_2 = \arg\max_{j \le k_{max}} \frac{\lambda_j(\hat{\boldsymbol{\Sigma}}_2)}{\lambda_{j+1}(\hat{\boldsymbol{\Sigma}}_2) + cD^{-2}}.$$

Value

`\eqn{k_1}`	The estimated row factor number.
`\eqn{k_2}`	The estimated column factor number.

Author(s)

Yong He, Changwei Zhao, Ran Zhao.

References

He, Y., Kong, X., Yu, L., Zhang, X., & Zhao, C. (2023). Matrix factor analysis: From least squares to iterative projection. Journal of Business & Economic Statistics, 1-26.

He, Y., Kong, X. B., Liu, D., & Zhao, R. (2023). Robust Statistical Inference for Large-dimensional Matrix-valued Time Series via Iterative Huber Regression. <arXiv:2306.03317>.

Examples

```
set.seed(11111)
   T=20;p1=20;p2=20;k1=3;k2=3
   R=matrix(runif(p1*k1,min=-1,max=1),p1,k1)
   C=matrix(runif(p2*k2,min=-1,max=1),p2,k2)
   X=array(0,c(T,p1,p2))
   Y=X;E=Y
   F=array(0,c(T,k1,k2))
   for(t in 1:T){
     F[t,,]=matrix(rnorm(k1*k2),k1,k2)
     E[t,,]=matrix(rnorm(p1*p2),p1,p2)
     Y[t,,]=R%*%F[t,,]%*%t(C)
   }
   X=Y+E

   KMHFA(X, kmax=6, method="P")

   KMHFA(X, W1 = NULL, W2 = NULL, 6, "E_RM")
   KMHFA(X, W1 = NULL, W2 = NULL, 6, "E_ER")
```

KPCA *Estimating the Pair of Factor Numbers via Eigenvalue Ratios Corresponding to α-PCA*

Description

The function is to estimate the pair of factor numbers via eigenvalue ratios corresponding to α-PCA.

Usage

```
KPCA(X, kmax, alpha = 0)
```

Arguments

X	Input an array with $T \times p_1 \times p_2$, where T is the sample size, p_1 is the the row dimension of each matrix observation and p_2 is the the column dimension of each matrix observation.
kmax	The user-supplied maximum factor numbers. Here it means the upper bound of the number of row factors and column factors.
alpha	A hyper-parameter balancing the information of the first and second moments ($\alpha \geq -1$). The default is 0.

Details

The function KPCA uses the eigenvalue-ratio idea to estimate the number of factors. In details, the number of factors k_1 is estimated by

$$\hat{k}_1 = \arg \max_{j \leq k_{max}} \frac{\lambda_j(\hat{\boldsymbol{M}}_R)}{\lambda_{j+1}(\hat{\boldsymbol{M}}_R)},$$

where k_{max} is a given upper bound. k_2 is defined similarly with respect to $\hat{\boldsymbol{M}}_C$. See the function alpha_PCA for the definition of $\hat{\boldsymbol{M}}_R$ and $\hat{\boldsymbol{M}}_C$. For more details, see Chen & Fan (2021).

Value

\eqn{k_1}	The estimated row factor number.
\eqn{k_2}	The estimated column factor number.

Author(s)

Yong He, Changwei Zhao, Ran Zhao.

References

Chen, E. Y., & Fan, J. (2021). Statistical inference for high-dimensional matrix-variate factor models. Journal of the American Statistical Association, 1-18.

Examples

```
set.seed(11111)
T=20;p1=20;p2=20;k1=3;k2=3
R=matrix(runif(p1*k1,min=-1,max=1),p1,k1)
C=matrix(runif(p2*k2,min=-1,max=1),p2,k2)
X=array(0,c(T,p1,p2))
Y=X;E=Y
F=array(0,c(T,k1,k2))
for(t in 1:T){
  F[t,,]=matrix(rnorm(k1*k2),k1,k2)
  E[t,,]=matrix(rnorm(p1*p2),p1,p2)
  Y[t,,]=R%*%F[t,,]%*%t(C)
}
X=Y+E

KPCA(X, 8, alpha = 0)
```

KPE *Estimating the Pair of Factor Numbers via Eigenvalue Ratios Corresponding to PE*

Description

The function is to estimate the pair of factor numbers via eigenvalue ratios corresponding to PE method.

Usage

```
KPE(X, kmax, c = 0)
```

Arguments

X	Input an array with $T \times p_1 \times p_2$, where T is the sample size, p_1 is the the row dimension of each matrix observation and p_2 is the the column dimension of each matrix observation.
kmax	The user-supplied maximum factor numbers. Here it means the upper bound of the number of row factors and column factors.
c	A constant to avoid vanishing denominators. The default is 0.

Details

The function KPE uses the eigenvalue-ratio idea to estimate the number of factors. First, obtain the initial estimators $\hat{\boldsymbol{R}}$ and $\hat{\boldsymbol{C}}$. Second, define

$$\hat{\boldsymbol{Y}}_t = \frac{1}{p_2}\boldsymbol{X}_t\hat{\boldsymbol{C}}, \hat{\boldsymbol{Z}}_t = \frac{1}{p_1}\boldsymbol{X}_t^\top\hat{\boldsymbol{R}},$$

and

$$\tilde{\boldsymbol{M}}_1 = \frac{1}{Tp_1}\hat{\boldsymbol{Y}}_t\hat{\boldsymbol{Y}}_t^\top, \tilde{\boldsymbol{M}}_2 = \frac{1}{Tp_2}\sum_{t=1}^{T}\hat{\boldsymbol{Z}}_t\hat{\boldsymbol{Z}}_t^\top,$$

the number of factors k_1 is estimated by

$$\hat{k}_1 = \arg\max_{j\leq k_{max}} \frac{\lambda_j(\tilde{\boldsymbol{M}}_1)}{\lambda_{j+1}(\tilde{\boldsymbol{M}}_1)},$$

where k_{max} is a predetermined upper bound for k_1. The estimation of k_2 is defined similarly with respect to $\tilde{\boldsymbol{M}}_2$. For details, see Yu et al. (2022).

Value

\eqn{k_1}	The estimated row factor number.
\eqn{k_2}	The estimated column factor number.

Author(s)

Yong He, Changwei Zhao, Ran Zhao.

References

Yu, L., He, Y., Kong, X., & Zhang, X. (2022). Projected estimation for large-dimensional matrix factor models. Journal of Econometrics, 229(1), 201-217.

Examples

```
set.seed(11111)
T=20;p1=20;p2=20;k1=3;k2=3
R=matrix(runif(p1*k1,min=-1,max=1),p1,k1)
C=matrix(runif(p2*k2,min=-1,max=1),p2,k2)
X=array(0,c(T,p1,p2))
Y=X;E=Y
F=array(0,c(T,k1,k2))
for(t in 1:T){
  F[t,,]=matrix(rnorm(k1*k2),k1,k2)
  E[t,,]=matrix(rnorm(p1*p2),p1,p2)
  Y[t,,]=R%*%F[t,,]%*%t(C)
}
X=Y+E

KPE(X, 8, c = 0)
```

MHFA *Matrix Huber Factor Analysis*

Description

This function is to fit the matrix factor models via the Huber loss. We propose two algorithms to do robust factor analysis. One is based on minimizing the Huber loss of the idiosyncratic error's Frobenius norm, which leads to a weighted iterative projection approach to compute and learn the parameters and thereby named as Robust-Matrix-Factor-Analysis (RMFA). The other one is based on minimizing the element-wise Huber loss, which can be solved by an iterative Huber regression algorithm (IHR).

Usage

```
MHFA(X, W1=NULL, W2=NULL, m1, m2, method, max_iter = 100, ep = 1e-04)
```

Arguments

X	Input an array with $T \times p_1 \times p_2$, where T is the sample size, p_1 is the the row dimension of each matrix observation and p_2 is the the column dimension of each matrix observation.

W1	Only if method="E", the inital value of row loadings matrix. The default is NULL, which is randomly chosen and all entries from a standard normal distribution.
W2	Only if method="E", the inital value of column loadings matrix. The default is NULL, which is randomly chosen and all entries from a standard normal distribution.
m1	A positive integer indicating the row factor numbers.
m2	A positive integer indicating the column factor numbers.
method	Character string, specifying the type of the estimation method to be used.
	"P", indicates minimizing the Huber loss of the idiosyncratic error's Frobenius norm. (RMFA)
	"E", indicates minimizing the elementwise Huber loss. (IHR)
max_iter	Only if method="E", the maximum number of iterations in the iterative Huber regression algorithm. The default is 100.
ep	Only if method="E", the stopping critetion parameter in the iterative Huber regression algorithm. The default is $10^{-4} \times Tp_1p_2$.

Details

For the matrix factor models, He et al. (2021) propose a weighted iterative projection approach to compute and learn the parameters by minimizing the Huber loss function of the idiosyncratic error's Frobenius norm. In details, for observations $\boldsymbol{X}_t, t = 1, 2, \cdots, T$, define

$$\boldsymbol{M}_c^w = \frac{1}{Tp_2}\sum_{t=1}^{T} w_t \boldsymbol{X}_t \boldsymbol{C}\boldsymbol{C}^\top \boldsymbol{X}_t^\top, \boldsymbol{M}_r^w = \frac{1}{Tp_1}\sum_{t=1}^{T} w_t \boldsymbol{X}_t^\top \boldsymbol{R}\boldsymbol{R}^\top \boldsymbol{X}_t.$$

The estimators of loading matrics $\hat{\boldsymbol{R}}$ and $\hat{\boldsymbol{C}}$ are calculated by $\sqrt{p_1}$ times the leading k_1 eigenvectors of $\boldsymbol{M}_c^w$ and $\sqrt{p_2}$ times the leading k_2 eigenvectors of $\boldsymbol{M}_r^w$. And

$$\hat{\boldsymbol{F}}_t = \frac{1}{p_1p_2}\hat{\boldsymbol{R}}^\top \boldsymbol{X}_t \hat{\boldsymbol{C}}.$$

For details, see He et al. (2023).

The other one is based on minimizing the element-wise Huber loss. Define

$$M_{i,Tp_2}(\boldsymbol{r}, \boldsymbol{F}_t, \boldsymbol{C}) = \frac{1}{Tp_2}\sum_{t=1}^{T}\sum_{j=1}^{p_2} H_\tau\left(x_{t,ij} - \boldsymbol{r}_i^\top \boldsymbol{F}_t \boldsymbol{c}_j\right),$$

$$M_{i,Tp_1}(\boldsymbol{R}, \boldsymbol{F}_t, \boldsymbol{c}) = \frac{1}{Tp_1}\sum_{t=1}^{T}\sum_{i=1}^{p_1} H_\tau\left(x_{t,ij} - \boldsymbol{r}_i^\top \boldsymbol{F}_t \boldsymbol{c}_j\right),$$

$$M_{t,p_1p_2}(\boldsymbol{R}, \mathrm{vec}(\boldsymbol{F}), \boldsymbol{C}) = \frac{1}{p_1p_2}\sum_{i=1}^{p_1}\sum_{j=1}^{p_2} H_\tau\left(x_{t,ij} - (\boldsymbol{c}_j \otimes \boldsymbol{r}_i)^\top \mathrm{vec}(\boldsymbol{F})\right).$$

This can be seen as Huber regression as each time optimizing one argument while keeping the other two fixed.

Value

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

F	The estimated factor matrix of dimension $T \times m_1 \times m_2$.
R	The estimated row loading matrix of dimension $p_1 \times m_1$, satisfying $\boldsymbol{R}^\top \boldsymbol{R} = p_1 \boldsymbol{I}_{m_1}$.
C	The estimated column loading matrix of dimension $p_2 \times m_2$, satisfying $\boldsymbol{C}^\top \boldsymbol{C} = p_2 \boldsymbol{I}_{m_2}$.

Author(s)

Yong He, Changwei Zhao, Ran Zhao.

References

He, Y., Kong, X., Yu, L., Zhang, X., & Zhao, C. (2023). Matrix factor analysis: From least squares to iterative projection. Journal of Business & Economic Statistics, 1-26.

He, Y., Kong, X. B., Liu, D., & Zhao, R. (2023). Robust Statistical Inference for Large-dimensional Matrix-valued Time Series via Iterative Huber Regression. <arXiv:2306.03317>.

Examples

```
set.seed(11111)
   T=20;p1=20;p2=20;k1=3;k2=3
   R=matrix(runif(p1*k1,min=-1,max=1),p1,k1)
   C=matrix(runif(p2*k2,min=-1,max=1),p2,k2)
   X=array(0,c(T,p1,p2))
   Y=X;E=Y
   F=array(0,c(T,k1,k2))
   for(t in 1:T){
     F[t,,]=matrix(rnorm(k1*k2),k1,k2)
     E[t,,]=matrix(rnorm(p1*p2),p1,p2)
     Y[t,,]=R%*%F[t,,]%*%t(C)
   }
   X=Y+E

   #Estimate the factor matrices and loadings by RMFA
   fit1=MHFA(X, m1=3, m2=3, method="P")
   Rhat1=fit1$R
   Chat1=fit1$C
   Fhat1=fit1$F

   #Estimate the factor matrices and loadings by IHR
   fit2=MHFA(X, W1=NULL, W2=NULL, 3, 3, "E")
   Rhat2=fit2$R
   Chat2=fit2$C
   Fhat2=fit2$F

   #Estimate the common component by RMFA
   CC1=array(0,c(T,p1,p2))
```

```
for (t in 1:T){
CC1[t,,]=Rhat1%*%Fhat1[t,,]%*%t(Chat1)
}
CC1

#Estimate the common component by IHR
CC2=array(0,c(T,p1,p2))
for (t in 1:T){
CC2[t,,]=Rhat2%*%Fhat2[t,,]%*%t(Chat2)
}
CC2
```

PE *Projected Estimation for Large-Dimensional Matrix Factor Models*

Description

This function is to fit the matrix factor model via the PE method by projecting the observation matrix onto the row or column factor space.

Usage

```
PE(X, m1, m2)
```

Arguments

X	Input an array with $T \times p_1 \times p_2$, where T is the sample size, p_1 is the the row dimension of each matrix observation and p_2 is the the column dimension of each matrix observation.
m1	A positive integer indicating the row factor numbers.
m2	A positive integer indicating the column factor numbers.

Details

For the matrix factor models, Yu et al. (2022) propose a projection estimation method to estimate the model parameters. In details, for observations $\boldsymbol{X}_t, t = 1, 2, \cdots, T$, the data matrix is projected to a lower dimensional space by setting

$$\boldsymbol{Y}_t = \frac{1}{p_2}\boldsymbol{X}_t\boldsymbol{C}.$$

Given $\boldsymbol{Y}_t$, define

$$\boldsymbol{M}_1 = \frac{1}{Tp_1}\sum_{t=1}^{T}\boldsymbol{Y}_t\boldsymbol{Y}_t^\top,$$

and then the row factor loading matrix $\boldsymbol{R}$ can be estimated by $\sqrt{p_1}$ times the leading k_1 eigenvectors of $\boldsymbol{M}_1$. However, the projection matrix $\boldsymbol{C}$ is unavailable in practice. A natural solution is to replace it with a consistent initial estimator. The column factor loading matrix $\boldsymbol{C}$ can be similarly estimated by projecting $\boldsymbol{X}_t$ onto the space of $\boldsymbol{C}$ with $\boldsymbol{R}$. See Yu et al. (2022) for the detailed algorithm.

Value

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

F	The estimated factor matrix of dimension $T \times m_1 \times m_2$.
R	The estimated row loading matrix of dimension $p_1 \times m_1$, satisfying $\boldsymbol{R}^\top \boldsymbol{R} = p_1 \boldsymbol{I}_{m_1}$.
C	The estimated column loading matrix of dimension $p_2 \times m_2$, satisfying $\boldsymbol{C}^\top \boldsymbol{C} = p_2 \boldsymbol{I}_{m_2}$.

Author(s)

Yong He, Changwei Zhao, Ran Zhao.

References

Yu, L., He, Y., Kong, X., & Zhang, X. (2022). Projected estimation for large-dimensional matrix factor models. Journal of Econometrics, 229(1), 201-217.

Examples

```
set.seed(11111)
T=20;p1=20;p2=20;k1=3;k2=3
R=matrix(runif(p1*k1,min=-1,max=1),p1,k1)
C=matrix(runif(p2*k2,min=-1,max=1),p2,k2)
X=array(0,c(T,p1,p2))
Y=X;E=Y
F=array(0,c(T,k1,k2))
for(t in 1:T){
  F[t,,]=matrix(rnorm(k1*k2),k1,k2)
  E[t,,]=matrix(rnorm(p1*p2),p1,p2)
  Y[t,,]=R%*%F[t,,]%*%t(C)
}
X=Y+E

#Estimate the factor matrices and loadings
fit=PE(X, k1, k2)
Rhat=fit$R
Chat=fit$C
Fhat=fit$F

#Estimate the common component
CC=array(0,c(T,p1,p2))
for (t in 1:T){
CC[t,,]=Rhat%*%Fhat[t,,]%*%t(Chat)
}
CC
```

Index