Package 'TensorPreAve'

April 14, 2023

Type Package

Title Rank and Factor Loadings Estimation in Time Series Tensor Factor Models

Version 1.1.0

Author Weilin Chen [aut, cre]

Description A set of functions to estimate rank and factor loadings of time series tensor factor models. A tensor is a multidimensional array. To analyze high-dimensional tensor time series, factor model is a major dimension reduction tool. 'TensorPreAve' provides functions to estimate the rank of core tensors and factor loading spaces of tensor time series. More specifically, a pre-averaging method that accumulates information from tensor fibres is used to estimate the factor loading spaces. The estimated directions corresponding to the strongest factors are then used for projecting the data for a potentially improved re-estimation of the factor loading spaces themselves. A new rank estimation method is also implemented to utilizes correlation information from the projected data.
See Chen and Lam (2023) <arXiv:2208.04012> for more details.

License GPL-3

Encoding UTF-8

LazyData true

URL https://github.com/William-Chenwl/TensorPreAve

RoxygenNote 7.2.1

Imports rTensor,MASS,stats,pracma

Depends R (>= 2.10)

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Maintainer Weilin Chen <w.chen56@lse.ac.uk>

Repository CRAN

Date/Publication 2023-04-14 13:20:02 UTC

R topics documented:

`bs_cor_rank` *Bootstrap Rank Estimation.*

Description

Function to estimate the rank of the core tensor by Bootstrapped Correlation Thresholding.

Usage

```
bs_cor_rank(X, initial_direction, r_range = NULL, C_range = NULL, B = 50)
```

Arguments

`X`	A 'Tensor' object defined in package **rTensor** with $K + 1$ modes. Mode-1 should correspond to the time mode.
`initial_direction`	Direction corresponds to the strongest factors, written in a list of K vectors. This can be obtained from the iterative projection procedure by using function `iter_proj`.
`r_range`	Approximate range of r_k (number of factors) to search from, written in a list of K vectors (e.g. `z = list(c(1,10),c(1,10))` for $K = 2$). Default range is 1 to 10 for all modes.
`C_range`	The range of constant C for calculating threshold. Default is `seq(0,100,0.1)`, and set to be automatically tuned as data-driven.
`B`	Number of bootstrap samples. Default is 50. Can be set as 10 to save time when dimension is large.

Details

Input a tensor time series and estimated directions corresponding to the strongest factors, return the estimated rank of core tensor.

Value

A vector of length K, indicating estimated number of factors in each mode.

Examples

```
# Example of real data set
set.seed(10)
Q_PRE = pre_est(value_weight_tensor)
Q_PROJ = iter_proj(value_weight_tensor, initial_direction = Q_PRE)
bs_rank = bs_cor_rank(value_weight_tensor, Q_PROJ)
bs_rank


# Example using generated data
K = 2
T = 100
d = c(40,40)
r = c(2,2)
re = c(2,2)
eta = list(c(0,0),c(0,0))
u = list(c(-2,2),c(-2,2))
set.seed(10)
Data_test = tensor_data_gen(K,T,d,r,re,eta,u)
X = Data_test$X
Q_PRE = pre_est(X)
Q_PROJ = iter_proj(X, initial_direction = Q_PRE)
bs_rank = bs_cor_rank(X, Q_PROJ)
bs_rank
```

---

equal_weight_tensor *Equal weight Fama-French portfolio returns data.*

Description

Equal weight Fama-French portfolio returns data formed on size and operating profitability of Chen and Lam (2023).

Format

A $576 \times 10 \times 10$ 'Tensor' object defined in package **rTensor**, where mode-1,2,3 correspond to time, OP levels and size levels, respectively.

Details

Stocks are categorized into 10 different sizes (market equity, using NYSE market equity deciles) and 10 different operating profitability (OP) levels (using NYSE OP deciles. OP is annual revenues minus cost of goods sold, interest expense, and selling, general, and administrative expenses divided by book equity for the last fiscal year end). The stocks in each of the 10×10 categories form a portfolio by equal weight. We use monthly data from July 1973 to June 2021, so that T = 576, and each data tensor we have thus has size $10 \times 10 \times 576$. Since the market factor is certainly pervasive in financial returns, we use the CAPM to remove its effects and facilitate detection of potentially weaker factors.

References

Chen, W. and Lam, C. (2023). Rank and Factor Loadings Estimation in Time Series Tensor Factor Model by Pre-averaging. Manuscript.

`iter_proj` *Iterative Projection Estimator.*

Description

Function for Iterative Projection Direction Refinement to re-estimate the factor loading matrices.

Usage

```
iter_proj(X, initial_direction, proj_N = 30, z = rep(1, X@num_modes - 1))
```

Arguments

`X`	A 'Tensor' object defined in package **rTensor** with $K + 1$ modes. Mode-1 should correspond to the time mode.
`initial_direction`	Initial direction for projection, written in a list of K vectors. This can be obtained from the pre-averaging procedure by using function `pre_est`.
`proj_N`	Number of iterations, should be a positive integer. Default is 30.
`z`	(Estimated) Rank of the core tensor, written as a vector of length K. Can be set as 1's when we only need to do rank estimation based on projected data. Default is 1's.

Details

Input a tensor time series and initial estimated directions corresponding to the strongest factors, return the estimated factor loading matrices (or directions) using the Algorithm for Iterative Projection Direction Refinement.

Value

A list of K estimated factor loading matrices.

Examples

```
# Example of a real data set
set.seed(10)
Q_PRE = pre_est(value_weight_tensor)
Q_PROJ = iter_proj(value_weight_tensor, initial_direction = Q_PRE)
Q_PROJ

set.seed(10)
Q_PRE = pre_est(value_weight_tensor)
```

```
Q_PROJ_2 = iter_proj(value_weight_tensor, initial_direction = Q_PRE, z = c(2,2))
Q_PROJ_2


# Example using generated data
K = 2
T = 100
d = c(40,40)
r = c(2,2)
re = c(2,2)
eta = list(c(0,0),c(0,0))
u = list(c(-2,2),c(-2,2))
set.seed(10)
Data_test = tensor_data_gen(K,T,d,r,re,eta,u)
X = Data_test$X
Q_PRE = pre_est(X)
Q_PROJ = iter_proj(X, initial_direction = Q_PRE, z = r)
Q_PROJ
```

`pre_eigenplot` *Eigenvalue Plot of a Random Sample*

Description

Function to plot the eigenvalues of the sample covariance matrix of a randomly chosen sample.

Usage

```
pre_eigenplot(X, k)
```

Arguments

`X`	A 'Tensor' object defined in package **rTensor** with $K + 1$ modes. Mode-1 should correspond to the time mode.
`k`	The mode to plot the eigenvalues for.

Details

Input a tensor time series and a mode index, output the plot of eigenvalues of the sample covariance matrix of the given mode, with a randomly chosen sample of the mode-k fibres. This helps users to choose the parameter `eigen_j` in function `pre_est`. A large dip should be observed at the (r_k+1)-th position of the plot, and user can choose `eigen_j` to be a bit larger than the position of dip observed to avoid missing potential weak factors. If such a dip is not observed, try to run the function for a few times until it can be observed.

Examples

```
# Example of a real data set
set.seed(800)
pre_eigenplot(value_weight_tensor, k = 2)


# Example using generated data
K = 2
T = 100
d = c(40,40)
r = c(2,2)
re = c(2,2)
eta = list(c(0,0),c(0,0))
u = list(c(-2,2),c(-2,2))
set.seed(10)
Data_test = tensor_data_gen(K,T,d,r,re,eta,u)
X = Data_test$X
pre_eigenplot(X, k = 1)
```

`pre_est` *Pre-Averaging Estimator*

Description

Function for the initial Pre-Averaging Procedure.

Usage

```
pre_est(X, z = rep(1, X@num_modes - 1), M0 = 200, M = 5, eigen_j = NULL)
```

Arguments

`X`	A 'Tensor' object defined in package **rTensor** with $K + 1$ modes. Mode-1 should correspond to the time mode.
`z`	(Estimated) Rank of the core tensor, written as a vector of length K. For iterative projection purpose, we only need this to be 1's. Default is 1's.
`M0`	Number of random samples to generate, should be a positive integer. Default is 200.
`M`	Number of chosen samples for pre-averaging, should be a positive integer. Usually can be set as constants (5 or 10) or 2.5 percents of `M0`. Default is 5.
`eigen_j`	The j-th eigenvalue to calculate eigenvalue-ratio for a randomly chosen sample, written as a vector of length K. Default is $d_k/2$ for all modes. Can be manually tuned using function `pre_eigenplot`.

Details

Input a tensor time series and return the estimated factor loading matrices (or directions) using pre-averaging method.

Value

A list of K estimated factor loading matrices.

Examples

```
# Example of a real data set
set.seed(10)
Q_PRE = pre_est(value_weight_tensor)
Q_PRE

set.seed(10)
Q_PRE_2 = pre_est(value_weight_tensor, z = c(2,2))
Q_PRE_2


# Example using generated data
K = 2
T = 100
d = c(40,40)
r = c(2,2)
re = c(2,2)
eta = list(c(0,0),c(0,0))
u = list(c(-2,2),c(-2,2))
set.seed(10)
Data_test = tensor_data_gen(K,T,d,r,re,eta,u)
X = Data_test$X
Q_PRE = pre_est(X, z = r)
Q_PRE
```

---

rank_factors_est *Rank and Factor Loadings Estimation*

Description

The complete procedure to estimate both rank and factor loading matrices simultaneously for a tensor time series.

Usage

```
rank_factors_est(
  X,
  proj_N = 30,
  r_range = NULL,
```

```
  C_range = NULL,
  M0 = 200,
  M = 5,
  B = 50,
  eigen_j = NULL,
  input_r = NULL
)
```

Arguments

X	A 'Tensor' object defined in package **rTensor** with $K + 1$ modes. Mode-1 should correspond to the time mode.
proj_N	Number of iterations for iterative projection. Default is 30.
r_range	Approximate range of r_k (number of factors) to search from, written in a list of K vectors (e.g. `z = list(c(1,10),c(1,10))` for $K = 2$). Default range is 1 to 10 for all modes.
C_range	The range of constant C for calculating threshold. Default is `seq(0,100,0.1)`, Default is `seq(0,100,0.1)`, and set to be automatically tuned as data-driven.
M0	Number of random samples to generate in pre-averaging procedure. Default is 200.
M	Number of chosen samples for pre-averaging. Usually can be set as constants (5 or 10) or 2.5 percents of `M0`. Default is 5.
B	Number of bootstrap samples for estimating rank of core tensor by bootstrapped correlation thresholding. Default is 50. Can be set as 10 when dimension is large.
eigen_j	The j-th eigenvalue to calculate eigenvalue-ratio for a randomly chosen sample, written as a vector of length K. Default is $d_k/2$ for all modes. Can be manually tuned using function `pre_eigenplot`.
input_r	The rank of core tensor if it is already know, written as a vector of length K. If no input, it will be estimated. Default is `NULL`.

Details

Input a tensor time series and return the estimated factor loading matrices and rank of core tensor.

Value

A list containing the following:
`rank`: A vector of K elements, indicating the estimated number of factors in each mode
`loadings`: A list of K estimated factor loading matrices.

Examples

```
# Example of real data set
set.seed(10)
results = rank_factors_est(value_weight_tensor)
results
```

```
# Example using generated data
K = 2
T = 100
d = c(40,40)
r = c(2,2)
re = c(2,2)
eta = list(c(0,0),c(0,0))
u = list(c(-2,2),c(-2,2))
set.seed(10)
Data_test = tensor_data_gen(K,T,d,r,re,eta,u)
X = Data_test$X
results = rank_factors_est(X)
results
```

`tensor_data_gen` *Tensor time series data generation.*

Description

Function to generate a random sample of time series tensor factor model, based on econometrics assumptions. (See Chen and Lam (2023) for more details on the assumptions.)

Usage

```
tensor_data_gen(K, n, d, r, re, eta, u, heavy_tailed = FALSE, t_df = 3)
```

Arguments

`K`	The number of modes for the tensor time series.
`n`	Length of time series.
`d`	Dimensions of each mode of the tensor, written in a vector of length `K`.
`r`	Rank of the core tensors, written in a vector of length `K`.
`re`	Rank of the cross-sectional common error core tensors, written in a vector of length `K`.
`eta`	Quantities controlling factor strengths in each factor loading matrix, written in a list of `K` vectors.
`u`	Quantities controlling range of elements in each factor loading matrix, written in a list of `K` vectors.
`heavy_tailed`	Whether to generate data from heavy-tailed distribution. If FALSE, generate from N(0,1); if TRUE, generate from t-distribution. Default is FALSE.
`t_df`	The degree of freedom for t-distribution if heavy_tailed = TRUE. Default is 3.

Details

Input tensor dimension and rank of core tensor, return a sample of tensor time series generated by factor model.

Value

A list containing the following:
`X`: the generated tensor time series, stored in a 'Tensor' object defined in **rTensor**, where mode-1 is the time mode
`A`: a list of K factor loading matrices
`F_ts`: time series of core tensor, stored in a 'Tensor' object, where mode-1 is the time mode
`E_ts`: time series of error tensor, stored in a 'Tensor' object, where mode-1 is the time mode

Examples

```
set.seed(10)
K = 2
n = 100
d = c(40,40)
r = c(2,2)
re = c(2,2)
eta = list(c(0,0),c(0,0))
u = list(c(-2,2),c(-2,2))
Data_test = tensor_data_gen(K,n,d,r,re,eta,u)

X = Data_test$X
A = Data_test$A
F_ts = Data_test$F_ts
E_ts = Data_test$E_ts

X@modes
F_ts@modes
E_ts@modes
dim(A[[1]])
```

`value_weight_tensor` *Value weighted Fama-French portfolio returns data.*

Description

Value weighted Fama-French portfolio returns data formed on size and operating profitability of Chen and Lam (2023).

Format

A $576 \times 10 \times 10$ 'Tensor' object defined in package **rTensor**, where mode-1,2,3 correspond to time, OP levels and size levels, respectively.

Details

Stocks are categorized into 10 different sizes (market equity, using NYSE market equity deciles) and 10 different operating profitability (OP) levels (using NYSE OP deciles. OP is annual revenues minus cost of goods sold, interest expense, and selling, general, and administrative expenses divided by book equity for the last fiscal year end). The stocks in each of the 10×10 categories form a portfolio using value weighted. We use monthly data from July 1973 to June 2021, so that T = 576, and each data tensor we have thus has size $10 \times 10 \times 576$. Since the market factor is certainly pervasive in financial returns, we use the CAPM to remove its effects and facilitate detection of potentially weaker factors.

References

Chen, W. and Lam, C. (2023). Rank and Factor Loadings Estimation in Time Series Tensor Factor Model by Pre-averaging. Manuscript.

Index