Package 'sparseDFM'

March 23, 2023

Title Estimate Dynamic Factor Models with Sparse Loadings

Version 1.0

Description Implementation of various estimation methods for dynamic factor models (DFMs) including principal components analysis (PCA) Stock and Watson (2002) <doi:10.1198/016214502388618960>, 2Stage Giannone et al. (2008) <doi:10.1016/j.jmoneco.2008.05.010>, expectation-maximisation (EM) Banbura and Modugno (2014) <doi:10.1002/jae.2306>, and the novel EM-sparse approach for sparse DFMs Mosley et al. (2023) <arXiv:2303.11892>. Options to use classic multivariate Kalman filter and smoother (KFS) equations from Shumway and Stoffer (1982) <doi:10.1111/j.1467-9892.1982.tb00349.x> or fast univariate KFS equations from Koopman and Durbin (2000) <doi:10.1111/1467-9892.00186>, and options for independent and identically distributed (IID) white noise or auto-regressive (AR(1)) idiosyncratic errors. Algorithms coded in 'C++' and linked to R via 'RcppArmadillo'.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.3

Imports Rcpp (>= 1.0.9), Matrix, ggplot2

LinkingTo Rcpp, RcppArmadillo

Suggests knitr, rmarkdown, gridExtra

VignetteBuilder knitr

Depends R (>= 3.3.0)

LazyData true

NeedsCompilation yes

Author Luke Mosley [aut],
Tak-Shing Chan [aut],
Alex Gibberd [aut, cre]

Maintainer Alex Gibberd <a.gibberd@lancaster.ac.uk>

Repository CRAN

Date/Publication 2023-03-23 19:40:02 UTC

R topics documented:

`exports` *UK Trade in Goods (Exports) Dataset*

Description

A full dataset used for nowcasting UK trade in goods (Exports) including the 9 export target series and 436 monthly indicator series.

Usage

```
exports
```

Format

`exports`**:**

A data frame with 226 observations and 445 variables:

columns Export target series (9) and monthly indicators (436).

rows Monthly values from Jan 2004 to Oct 2022. ...

Source

https://www.ons.gov.uk/

Examples

```
# load exports data
data = exports
```

fillNA *Interpolation of missing data*

Description

Internal missing data is filled in using a cubic spline. Start and end of sample missing data is filled in using the median of the series and then smoothed with an MA(3) process.

Usage

```
fillNA(X)
```

Arguments

X	n x p numeric matrix of stationary and standardized time series

Value

X $n \times p$ numeric matrix with missing data interpolated

idx.na $n \times p$ logical matrix with TRUE if missing and FALSE otherwise.

inflation *UK Inflation Dataset*

Description

A subset of quarterly CPI Index data from the ONS Inflation data (Q4 2022 release).

A subset of quarterly CPI Index data from the ONS Inflation data (Q4 2022 release).

Usage

```
inflation

inflation
```

Format

`inflation`:
A data frame with 135 observations and 36 variables:
columns Different classes of inflation index
rows Quarterly values of the relevant CPI index, benchmarked to 2015=100 ...

`inflation`:
A data frame with 135 observations and 36 variables:
columns Different classes of inflation index
rows Quarterly values of the relevant CPI index, benchmarked to 2015=100 ...

Source

https://www.ons.gov.uk/economy/inflationandpriceindices/datasets/consumerpriceindices

https://www.ons.gov.uk/economy/inflationandpriceindices/datasets/consumerpriceindices

Examples

```
data = inflation

# load inflation data
data = inflation
```

`kalmanMultivariate` *Classic Multivariate KFS Equations*

Description

Implementation of the classic multivariate Kalman filter and smoother equations of Shumway and Stoffer (1982).

Usage

```
kalmanMultivariate(X, a0_0, P0_0, A, Lambda, Sig_e, Sig_u)
```

Arguments

`X`	n x p, numeric matrix of (stationary) time series
`a0_0`	k x 1, initial state mean vector
`P0_0`	k x k, initial state covariance matrix
`A`	k x k, state transition matrix
`Lambda`	p x k, measurement matrix
`Sig_e`	p x p, measurement equation residuals covariance matrix (diagonal)
`Sig_u`	k x k, state equation residuals covariance matrix

Details

For full details of the classic multivariate KFS approach, please refer to Mosley et al. (2023). Note that n is the number of observations, p is the number of time series, and k is the number of states.

Value

logl log-likelihood of the innovations from the Kalman filter

at_t $k \times n$, filtered state mean vectors

Pt_t $k \times k \times n$, filtered state covariance matrices

at_n $k \times n$, smoothed state mean vectors

Pt_n $k \times k \times n$, smoothed state covariance matrices

Pt_tlag_n $k \times k \times n$, smoothed state covariance with lag

References

Mosley, L., Chan, TS., & Gibberd, A. (2023). sparseDFM: An R Package to Estimate Dynamic Factor Models with Sparse Loadings.

Shumway, R. H., & Stoffer, D. S. (1982). An approach to time series smoothing and forecasting using the EM algorithm. *Journal of time series analysis, 3*(4), 253-264.

`kalmanUnivariate` *Univariate filtering (sequential processing) for fast KFS*

Description

Univariate treatment (sequential processing) of the multivariate Kalman filter and smoother equations for fast implementation. Refer to Koopman and Durbin (2000).

Usage

```
kalmanUnivariate(X, a0_0, P0_0, A, Lambda, Sig_e, Sig_u)
```

Arguments

`X`	n x p, numeric matrix of (stationary) time series
`a0_0`	k x 1, initial state mean vector
`P0_0`	k x k, initial state covariance matrix
`A`	k x k, state transition matrix
`Lambda`	p x k, measurement matrix
`Sig_e`	p x p, measurement equation residuals covariance matrix (diagonal)
`Sig_u`	k x k, state equation residuals covariance matrix

Details

For full details of the univariate filtering approach, please refer to Mosley et al. (2023). Note that n is the number of observations, p is the number of time series, and k is the number of states.

Value

logl log-likelihood of the innovations from the Kalman filter

at_t $k \times n$, filtered state mean vectors

Pt_t $k \times k \times n$, filtered state covariance matrices

at_n $k \times n$, smoothed state mean vectors

Pt_n $k \times k \times n$, smoothed state covariance matrices

Pt_tlag_n $k \times k \times n$, smoothed state covariance with lag

References

Koopman, S. J., & Durbin, J. (2000). Fast filtering and smoothing for multivariate state space models. *Journal of Time Series Analysis, 21*(3), 281-296.

Mosley, L., Chan, TS., & Gibberd, A. (2023). sparseDFM: An R Package to Estimate Dynamic Factor Models with Sparse Loadings.

`logspace` *logspace*

Description

Produce a vector of log10 space values

Usage

```
logspace(x1, x2, n = 50)
```

Arguments

`x1`	lower bound
`x2`	upper bound
`n`	length

Value

Vector of log10 spaced values of length n

`missing_data_plot` *Plot the missing data in a data matrix/frame*

Description

Visualise the amount of missing data in a data matrix or data frame.

Usage

```
missing_data_plot(
  data,
  present.colour = "grey80",
  missing.colour = "grey20",
  use.names = TRUE
)
```

Arguments

data	Numeric matrix or data frame with NA for missing values.
present.colour	The colour for data that is present. Default is 'grey80'.
missing.colour	The colour for data that is missing. Default is 'grey20'.
use.names	Logical. Label the axis with data variables names. Default is TRUE. Set to FALSE to remove.

Value

A matrix plot showing where missing data is present.

---

plot.sparseDFM *sparseDFM Plot Outputs*

Description

Make plots for the output of sparseDFM(). Options include:

- `factor` - plot factor estimate series on top of the original standardized stationary data
- `loading.heatmap` - make a heatmap of the loadings matrix
- `loading.lineplot` - make a lineplot of variable loadings for a given factor
- `loading.grouplineplot` - separate variable groups into colours for better visualisation
- `residual` - boxplot or scatterplot of residuals
- `lasso.bic` - BIC values for the LASSO tuning parameter
- `em.convergence` - log-likelihood convergence of EM iterations

Usage

```
## S3 method for class 'sparseDFM'
plot(
  x,
  type = "factor",
  which.factors = 1:(dim(x$state$factors)[2]),
  scale.factors = TRUE,
  which.series = 1:(dim(x$params$Lambda)[1]),
  loading.factor = 1,
  series.col = "grey",
  factor.col = "black",
  factor.lwd = 2,
  factor.lab = NULL,
  use.series.names = FALSE,
  series.lab = NULL,
  series.labpos = NULL,
  colorkey = TRUE,
```

```
  col.regions = NULL,
  group.names = NULL,
  group.cols = NULL,
  group.legend = TRUE,
  residual.type = "boxplot",
  scatter.series = 1,
  min.bic.col = "red",
  alpha_index = "best",
  ...
)
```

Arguments

x	an object of class 'sparseDFM'.
type	character. The type of plot: "factor", "loading.heatmap", "loading.lineplot", "loading.grouplineplot" or "residual". Default is "factor".
which.factors	numeric vector of integers representing which factors should be plotted in "factor" and "loading.heatmap". Default is which.factors=1:(dim(x$state$factors)[2]), plotting them all. Accepts a single integer if just one factor required.
scale.factors	logical. Standardize the factor estimates when plotting in "factor". Default is TRUE.
which.series	numeric vector of integers representing which series should be plotted in "loading.heatmap", "loading.lineplot", "loading.grouplineplot" and "residual". Default is which.series = 1:(dim(x$params$Lambda)[1]), plotting them all.
loading.factor	integer. The factor to use in "loading.lineplot" and "loading.grouplineplot". Default is 1.
series.col	character. The colour of the background series plotted in "factor". Default is series.col = "grey".
factor.col	character. The colour of the factor estimate line in "factor". Default is factor.col = "black".
factor.lwd	integer. The line width of the factor estimate line in "factor". Default is factor.lwd = 2.
factor.lab	vector of characters to label each factor in "loading.heatmap". Default is NULL for standard labeling.
use.series.names	logical. Set to TRUE if plot should display series names in the data matrix X. Default is FALSE for numbered series.
series.lab	vector of characters to label each data series in "loading.heatmap". Default is NULL for standard labeling.
series.labpos	numeric vector of integers representing which series are labeled by series.lab. Default is NULL for standard labeling.
colorkey	logical. Display the colour key of the heatmap in "loading.heatmap". Default is TRUE.
col.regions	vector of gradually varying colors for "loading.heatmap", see levelplot package. Default is NULL for standard colours.

group.names	vector of characters of the same dimension as which.series to represent the name of the group for each series in "loading.grouplineplot".
group.cols	vector of characters of the same dimension as the number of different groups in "loading.grouplineplot" to represent the colours of the groups.
group.legend	logical. Display the legend. Default is TRUE.
residual.type	character. The type of residual plot: "boxplot" or "scatterplot". Default is "boxplot".
scatter.series	integer. The series to plot when residual.type = "scatterplot". Default is series 1.
min.bic.col	character. Colour for the best α point. Default is 'red'.
alpha_index	Choose which L1 penalty parameter to display the results for. Default is 'best'. Otherwise, input a number between 1:length(alpha_grid) that indicates the required alpha parameter.
...	for plot.sparseDFM. Further plot arguments.

Value

Plots for the output of sparseDFM().

`predict.sparseDFM` *Forecasting factor estimates and data series.*

Description

Predict the next h steps ahead for the factor estimates and the data series. Given information up to time t, a h-step ahead forecast is $\boldsymbol{X}_{t+h} = \boldsymbol{\Lambda}\boldsymbol{A}^h\boldsymbol{F}_t + \boldsymbol{\Phi}^h\boldsymbol{\epsilon}_t$, where $\boldsymbol{\Phi} = 0$ for the IID idiosyncratic error case.

Usage

```
## S3 method for class 'sparseDFM'
predict(object, h = 1, standardize = FALSE, alpha_index = "best", ...)

## S3 method for class 'sparseDFM_forecast'
print(x, ...)
```

Arguments

object	an object of class 'sparseDFM'.
h	integer. The number of steps ahead to compute the forecast for. Default is $h = 1$.
standardize	logical. Returns data series forecasts in the original data scale if set to FALSE. Default is FALSE.
alpha_index	Choose which L1 penalty parameter to display the results for. Default is 'best'. Otherwise, input a number between 1:length(alpha_grid) that indicates the required alpha parameter.
...	Further print arguments.
x	an object of class 'sparseDFM_forecast' from predict.sparseDFM.

Value

X_hat $h \times p$ numeric matrix of data series forecasts.

F_hat $h \times r$ numeric matrix of factor forecasts.

e_hat $h \times p$ numeric matrix of AR(1) idiosyncratic error forecasts if `err=AR1` in `sparseDFM`.

h forecasts produced for h steps ahead.

err the type of idiosyncratic errors used in `sparseDFM`.

Prints out the h-step ahead forecast from `predict.sparseDFM`.

`raggedEdge` *Generate a ragged edge structure for a data matrix*

Description

Generate a ragged edge structure for a data matrix

Usage

```
raggedEdge(X, lags)
```

Arguments

`X`	numeric data matrix
`lags`	vector of integers representing publication lag of each variable

Value

ragged edge version of X

Examples

```
data = matrix(rnorm(100),ncol=10)
pub_lags = c(rep(2,5),rep(1,3),rep(0,2))
new_data = raggedEdge(data, pub_lags)
```

residuals.sparseDFM *sparseDFM Residuals and Fitted Values*

Description

Obtain the residuals or fitted values of the sparseDFM fit.

Usage

```
## S3 method for class 'sparseDFM'
fitted(object, standardize = FALSE, alpha_index = "best", ...)

## S3 method for class 'sparseDFM'
residuals(object, standardize = FALSE, alpha_index = "best", ...)
```

Arguments

object	an object of class 'sparseDFM'.
standardize	logical. The residuals and fitted values should be standardized. Default is FALSE, values returned in the original data $\boldsymbol{X}$ scale.
alpha_index	Choose which L1 penalty parameter to display the results for. Default is 'best'. Otherwise, input a number between 1:length(alpha_grid) that indicates the required alpha parameter.
...	Further residuals arguments.

Value

Residuals or fitted values of sparseDFM.

sparseDFM *Estimate a Sparse Dynamic Factor Model*

Description

Main function to allow estimation of a DFM or a sparse DFM (with sparse loadings) on stationary data that may have arbitrary patterns of missing data. We allow the user:

- an option for estimation method - "PCA", "2Stage", "EM" or "EM-sparse"
- an option for IID or AR1 idiosyncratic errors
- an option for Kalman Filter/Smoother estimation using standard multivariate equations or fast univariate filtering equations

Usage

```
sparseDFM(
  X,
  r,
  q = 0,
  alphas = logspace(-2, 3, 100),
  alg = "EM-sparse",
  err = "IID",
  kalman = "univariate",
  store.parameters = FALSE,
  standardize = TRUE,
  max_iter = 100,
  threshold = 1e-04
)
```

Arguments

X	n x p numeric data matrix or data frame of (stationary) time series.
r	Integer. Number of factors.
q	Integer. The first q series (columns of X) should not be made sparse. Default q = 0.
alphas	Numeric vector or value of LASSO regularisation parameters. Default is alphas = logspace(-2,3,100).
alg	Character. Option for estimation algorithm. Default is "EM-sparse". Options are:
"PCA"	principle components analysis (PCA) for static factors seen in Stock and Watson (2002).
"2Stage"	the two-stage framework of PCA plus Kalman filter/smoother seen in Giannone et al. (2008) and Doz et al
"EM"	the quasi-maximum likelihood approach using the EM algorithm to handle arbitrary patterns of missing da
"EM-sparse"	the novel sparse EM approach allowing LASSO regularisation on factor loadings seen in (cite our paper).
err	Character. Option for idiosyncratic errors. Default is "IID". Options are:
"IID"	errors are IID white noise.
"AR1"	errors follow an AR(1) process.
kalman	Character. Option for Kalman filter and smoother equations. Default is "univariate". Options are:

"multivariate"	classic Kalman filter and smoother equations seen in Shumway and Stoffer (1982).
"univaraite"	univariate treatment (sequential processing) of the multivariate equations for fast Kalman filter and smo

store.parameters	Logical. Store outputs for every alpha L1 penalty parameter. Default is FALSE.
standardize	Logical. Standardize the data before estimating the model. Default is TRUE.
max_iter	Integer. Maximum number of EM iterations. Default is 100.
threshold	Numeric. Tolerance on EM iterates. Default is 1e-4.

Details

For full details of the model please refer to Mosley et al. (2023).

Value

A list-of-lists-like S3 object of class 'sparseDFM' with the following elements:

data	A list containing information about the data with the following elements:
X	is the original $n \times p$ numeric data matrix of (stationary) time series.
standardize	is a logical value indicating whether the original data was standardized.
X.mean	is a p-dimensional numeric vector of column means of X.
X.sd	is a p-dimensional numeric vector of column standard deviations of X.
X.bal	is a $n \times p$ numeric data matrix of the original X with missing data interpolated using fillNA().
eigen	is the eigen decomposition of X.bal.
fitted	is the $n \times p$ predicted data matrix using the estimated parameters: $\hat{\Lambda}\hat{F}$.
fitted.unscaled	is the $n \times p$ predicted data matrix using the estimated parameters: $\hat{\Lambda}\hat{F}$ that has been unscaled back to
method	the estimation algorithm used (alg).
err	the type of idiosyncratic errors assumed. Either IID or AR1.
call	call object obtained from match.call().

params	A list containing the estimated parameters of the model with the following elements:

A	the $r \times r$ factor transition matrix.
Phi	the p-dimensional vector of AR(1) coefficients for the idiosyncratic errors.
Lambda	the $p \times r$ loadings matrix.
Sigma_u	the $r \times r$ factor transition error covariance matrix.
Sigma_epsilon	the p-dimensional vector of idiosyncratic error variances. As $\mathbf{\Sigma}_\epsilon$ is assumed to be diagonal.

state	A list containing the estimated states and state covariances with the following elements:
factors	the $n \times r$ matrix of factor estimates.
errors	the $n \times p$ matrix of AR(1) idiosyncratic error estimates. For err = AR1 only.
factors.cov	the $r \times r \times n$ covariance matrices of the factor estimates.
errors.cov	the $p \times p \times n$ covariance matrices of the AR(1) idiosyncratic error estimates. For err = AR1 only.

em	A list containing information about the EM algorithm with the following elements:
converged	a logical value indicating whether the EM algorithm converged.
alpha_grid	a numerical vector containing the LASSO tuning parameters considered in BIC evaluation before stopping.
alpha_opt	the optimal LASSO tuning parameter used.
bic	a numerical vector containing BIC values for the corresponding LASSO tuning parameter in alpha_grid.
loglik	the log-likelihood of the innovations from the Kalman filter in the final model.
num_iter	number of iterations taken by the EM algorithm.
tol	tolerance for EM convergence. Matches threshold in the input.
max_iter	maximum number of iterations allowed for the EM algorithm. Matches max_iter in the input.
em_time	time taken for EM convergence

`alpha.output` Parameter and state outputs for each L1-norm penalty parameter in `alphas` if `store.parameters = TRUE`.

References

Banbura, M., & Modugno, M. (2014). Maximum likelihood estimation of factor models on datasets with arbitrary pattern of missing data. *Journal of Applied Econometrics, 29*(1), 133-160.

Doz, C., Giannone, D., & Reichlin, L. (2011). A two-step estimator for large approximate dynamic factor models based on Kalman filtering. *Journal of Econometrics, 164*(1), 188-205.

Giannone, D., Reichlin, L., & Small, D. (2008). Nowcasting: The real-time informational content of macroeconomic data. *Journal of monetary economics, 55*(4), 665-676.

Koopman, S. J., & Durbin, J. (2000). Fast filtering and smoothing for multivariate state space models. *Journal of Time Series Analysis, 21*(3), 281-296.

Mosley, L., Chan, TS., & Gibberd, A. (2023). sparseDFM: An R Package to Estimate Dynamic Factor Models with Sparse Loadings.

Shumway, R. H., & Stoffer, D. S. (1982). An approach to time series smoothing and forecasting using the EM algorithm. *Journal of time series analysis, 3*(4), 253-264.

Stock, J. H., & Watson, M. W. (2002). Forecasting using principal components from a large number of predictors. *Journal of the American statistical association, 97*(460), 1167-1179.

Examples

```
# load inflation data set
data = inflation

# reduce the size for these examples - full data found in vignette
data = data[1:60,]

# make stationary by taking first differences
new_data = transformData(data, rep(2,ncol(data)))

# tune for the number of factors to use
tuneFactors(new_data, type = 2)

# fit a PCA using 3 PC's
fit.pca <- sparseDFM(new_data, r = 3, alg = 'PCA')

# fit a DFM using the two-stage approach
fit.2stage <- sparseDFM(new_data, r = 3, alg = '2Stage')

# fit a DFM using EM algorithm with 3 factors
fit.dfm <- sparseDFM(new_data, r = 3, alg = 'EM')

# fit a Sparse DFM with 3 factors
fit.sdfm <- sparseDFM(new_data, r = 3, alg = 'EM-sparse')

# observe the factor loadings of the sparse DFM
plot(fit.sdfm, type = 'loading.heatmap')
```

```
# observe the factors
plot(fit.sdfm, type = 'factor')

# observe the residuals
plot(fit.sdfm, type = 'residual')

# observe the LASSO parameter selected and BIC values
plot(fit.sdfm, type = 'lasso.bic')

# predict 3 steps ahead
predict(fit.sdfm, h = 3)
```

`summary.sparseDFM` *sparseDFM Summary Outputs*

Description

Summary and print outputs for class 'sparseDFM'.

Usage

```
## S3 method for class 'sparseDFM'
print(x, ...)

## S3 method for class 'sparseDFM'
summary(object, ...)
```

Arguments

`x`	an object of class 'sparseDFM'
`...`	Further `summary` arguments.
`object`	an object of class 'sparseDFM'

Value

Information on the model fitted.

Summary information on estimation details.

transformData *Transform data to make it stationary*

Description

Methods to transform the data to make it stationary. Input a $n \times p$ numeric data matrix and what transform is required for each data series. Returns a $n \times p$ matrix of the transformed data.

Usage

```
transformData(X, stationary_transform)
```

Arguments

X	n x p numeric data matrix
stationary_transform	p-dimensional vector filled with numbers from $\{1, 2, 3, 4, 5, 6, 7\}$ representing:

1	no change
2	first difference $X_{i,t} - X_{i,t-1}$
3	second difference $(X_{i,t} - X_{i,t-1}) - (X_{i,t-1} - X_{i,t-2})$
4	log first difference $log(X_{i,t}) - log(X_{i,t-1})$
5	log second difference $(log(X_{i,t}) - log(X_{i,t-1})) - (log(X_{i,t-1}) - log(X_{i,t-2}))$
6	growth rate $(X_{i,t} - X_{i,t-1})/X_{i,t-1}$
7	log growth rate $(log(X_{i,t}) - log(X_{i,t-1}))/log(X_{i,t-1})$

Value

Transformed stationary version of $\boldsymbol{X}$.

tuneFactors *Tune for the number of factors to use*

Description

Uses Bai and Ng (2002) information criteria approach. Missing data is interpolated using the `fillNA` function.

Usage

```
tuneFactors(
  X,
  type = 2,
  standardize = TRUE,
  r.max = min(15, ncol(X) - 1),
  plot = TRUE
)
```

Arguments

X	n x p numeric data matrix or data frame of (stationary) time series.
type	Character. Option for which information criteria to use. Default is 2.
standardize	Logical. Standardize the data before estimating the model. Default is TRUE.
r.max	Integer. Maximum number of factors to search for. Default is min(15,ncol(X)-1).
plot	Logical. Make a plot showing the IC value for each of the number of factors considered. Default is TRUE.

Details

To calculate the number of factors to use in the model, the information criteria approach of Bai and Ng (2002) is used. This can be done before sparseDFM is fitted to the data to determine r. Bai and Ng (2002) consider 3 types of information criteria with different penalties of the form:

$$IC_1(r) = log\left(V_r(\hat{\boldsymbol{F}}, \hat{\boldsymbol{\Lambda}})\right) + r\left(\frac{n+p}{np}\right) log\left(\frac{np}{n+p}\right)$$

$$IC_2(r) = log\left(V_r(\hat{\boldsymbol{F}}, \hat{\boldsymbol{\Lambda}})\right) + r\left(\frac{n+p}{np}\right) log\left(min\{n,p\}\right)$$

$$IC_3(r) = log\left(V_r(\hat{\boldsymbol{F}}, \hat{\boldsymbol{\Lambda}})\right) + r\frac{log\left(min\{n,p\}\right)}{min\{n,p\}}$$

The sum of squared residuals for r factors $V_r(\hat{\boldsymbol{F}}, \hat{\boldsymbol{\Lambda}}) = \sum_{i=1}^{p}\sum_{t=1}^{n} E[\hat{\epsilon}_{i,t}^2]/np$ with $\hat{\epsilon}_{i,t} = X_{t,i} - \hat{\boldsymbol{F}}_t\hat{\boldsymbol{\Lambda}}_i$ is found using PCA on the standardized data set $\boldsymbol{X}$. The estimated factors $\hat{\boldsymbol{F}}$ corresponding to the principle components and the estimated loadings $\hat{\boldsymbol{\Lambda}}$ corresponding to the eigenvectors. Should the data contain missing values, then the missing data is interpolated using fillNA.

The number of factors to use will correspond to $argmin_r IC_i(r)$ for $i = 1, 2$ or 3. Type 2 is the highest when working in finite samples and therefore is set to default.

Value

The number of factors to use according to Bai and Ng (2002) information criteria.

References

Bai, J., & Ng, S. (2002). Determining the number of factors in approximate factor models. *Econometrica, 70*(1), 191-221.

Index