Package 'fnets'

November 16, 2022

Type Package

Title Factor-adjusted Network Estimation and Forecasting for High-dimensional Time Series

Version 0.1.1

Maintainer Haeran Cho <haeran.cho@bristol.ac.uk>

Description Implements methods for network estimation and forecasting of high-dimensional time series exhibiting strong serial and cross-sectional correlations under a factor-adjusted vector autoregressive model.

Depends R (>= 4.1.0)

Imports lpSolve,
parallel,
doParallel,
foreach,
MASS,
fields,
igraph,
RColorBrewer

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.2

R topics documented:

`common.predict` *Forecasting the factor-driven common component*

Description

Produces forecasts of the common component for a given forecasting horizon by estimating the best linear predictors

Usage

```
common.predict(object, x, h = 1, forecast.restricted = TRUE, r = c("ic", "er"))
```

Arguments

`object`	`fnets` object
`x`	input time series matrix, with each row representing a variable
`h`	forecasting horizon
`forecast.restricted`	whether to forecast using a restricted or unrestricted, blockwise VAR representation of the common component
`r`	number of restricted factors, or a string specifying the factor number selection method when `forecast.restricted = TRUE`; possible values are: • `"ic"` information criteria of Bai and Ng (2002) • `"er"` eigenvalue ratio

Value

a list containing

`is`	in-sample estimator of the common component
`fc`	forecasts of the common component for a given forecasting horizon `h`
`r`	restricted factor number
`h`	forecast horizon

References

Barigozzi, M., Cho, H. & Owens, D. (2021) FNETS: Factor-adjusted network estimation and forecasting for high-dimensional time series. arXiv preprint arXiv:2201.06110.

Ahn, S. C. & Horenstein, A. R. (2013) Eigenvalue ratio test for the number of factors. Econometrica, 81(3), 1203–1227.

Forni, M., Hallin, M., Lippi, M. & Reichlin, L. (2005). The generalized dynamic factor model: one-sided estimation and forecasting. Journal of the American Statistical Association, 100(471), 830–840.

Forni, M., Hallin, M., Lippi, M. & Zaffaroni, P. (2017). Dynamic factor models with infinite-dimensional factor space: Asymptotic analysis. Journal of Econometrics, 199(1), 74–92.

Examples

```
set.seed(123)
n <- 500
p <- 50
common <- sim.unrestricted(n, p)
idio <- sim.var(n, p)
x <- common$data + idio$data
out <- fnets(x, q = NULL, var.order = 1, var.method = "lasso", do.lrpc = FALSE)
cpre <- common.predict(out, x, h = 1, r = NULL)
ipre <- idio.predict(out, x, cpre, h = 1)
```

---

factor.number — *Factor number estimators of Hallin and Liška (2007) and Bai and Ng (2002)*

Description

Estimates the number of factors by minimising an information criterion over sub-samples of the data. For `restricted = FALSE`, the three information criteria proposed in Hallin and Liška (2007) (`pen.op = 1, 2` or `3`) and their variations with logarithm taken on the cost (`pen.op = 4, 5` or `6`) are implemented, with `pen.op = 5` recommended as a default choice based on numerical experiments. For `restricted = TRUE`, the three information criteria in Owens, Cho, and Barigozzi (2022) are implemented, with `pen.op = 2` recommended by default.

Usage

```
factor.number(
  x,
  restricted = FALSE,
  q.max = NULL,
  mm = NULL,
  w = NULL,
  covx = NULL,
  do.plot = FALSE,
  center = TRUE
)
```

Arguments

x	input time series matrix, with each row representing a variable
restricted	whether to estimate the number of restricted or unrestricted factors
q.max	maximum number of factors; if q.max = NULL, a default value is selected as min(50, floor(sqrt(min(dim(x)[2] - 1, dim(x)[1]))))
mm	bandwidth; defaults to floor(4 * (dim(x)[2]/log(dim(x)[2]))^(1/3)))
w	vector of length 2 * mm + 1 containing symmetric weights; if w = NULL, default weights are generated using the Bartlett kernel and mm
covx	covariance matrix of x
do.plot	whether to plot the information criteria values
center	whether to de-mean the input x row-wise

Details

For further details, see Hallin and Liška (2007), Bai and Ng (2002), or Alessi, Barigozzi, and Capasso (2010).

Value

a list containing

q.hat	a vector containing minimisers of the six information criteria
sv	a list containing the singular value decomposition of Sigma_x

and if restricted = FALSE

Gamma_x	an array containing the estimates of the autocovariance matrices of x at 2 * mm + 1 lags
Sigma_x	an array containing the estimates of the spectral density matrices of x at 2 * mm + 1 Fourier frequencies

References

Hallin, M. & Liška, R. (2007) Determining the number of factors in the general dynamic factor model. Journal of the American Statistical Association, 102(478), 603–617.

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

Alessi, L., Barigozzi, M., and Capasso, M. (2010) Improved penalization for determining the number of factors in approximate factor models. Statistics & Probability Letters, 80(23-24):1806–1813.

Examples

```
library(fnets)

set.seed(123)
n <- 500
p <- 50
common <- sim.unrestricted(n, p)
idio <- sim.var(n, p)
x <- common$data * apply(idio$data, 1, sd) / apply(common$data, 1, sd) + idio$data

hl <- factor.number(x, restricted = FALSE, do.plot = TRUE)
hl$q.hat
library(fnets)

set.seed(123)
n <- 500
p <- 50
common <- sim.restricted(n, p)
idio <- sim.var(n, p)
x <- common$data * apply(idio$data, 1, sd) / apply(common$data, 1, sd) + idio$data

bn <- factor.number(x, restricted = TRUE, do.plot = TRUE)
bn$q.hat
```

`fnets` *Factor-adjusted network estimation*

Description

Operating under a factor-adjusted vector autoregressive (VAR) model, the function estimates the spectral density and autocovariance matrices of the factor-driven common component and the idiosyncratic VAR process, the impulse response functions and common shocks for the common component, and VAR parameters, innovation covariance matrix and long-run partial correlations for the idiosyncratic component.

Usage

```
fnets(
  x,
  center = TRUE,
  fm.restricted = FALSE,
  q = c("ic", "er"),
  pen.op = NULL,
  kern.bw = NULL,
 common.args = list(factor.var.order = NULL, max.var.order = NULL, trunc.lags = 20,
    n.perm = 10),
  var.order = 1,
  var.method = c("lasso", "ds"),
```

```
  var.args = list(tuning = c("cv", "ic"), n.iter = 100, tol = 0, n.cores =
    min(parallel::detectCores() - 1, 3)),
  var.threshold = FALSE,
  do.lrpc = TRUE,
  lrpc.adaptive = FALSE,
 tuning.args = list(n.folds = 1, penalty = NULL, path.length = 10, do.plot = FALSE)
)
```

Arguments

`x` input time series matrix, with each row representing a variable

`center` whether to de-mean the input `x` row-wise

`fm.restricted` whether to estimate a restricted factor model using static PCA

`q` Either a string specifying the factor number selection method when `fm.restricted = TRUE`; possible values are:

- `"ic"` information criteria of Hallin and Liška (2007) or Bai and Ng (2002), see factor.number
- `"er"` eigenvalue ratio

; or the number of unrestricted factors.

`pen.op` choice of the information criterion penalty, see factor.number for further details

`kern.bw` kernel bandwidth for dynamic PCA; defaults to `floor(4 *(dim(x)[2]/log(dim(x)[2]))^(1/3)))`

`common.args` a list specifying the tuning parameters required for estimating the impulse response functions and common shocks. It contains:

- `factor.var.order` order of the blockwise VAR representation of the common component. If `factor.var.order = NULL`, it is selected blockwise by Schwarz criterion
- `max.var.order` maximum blockwise VAR order for the Schwarz criterion
- `trunc.lags` truncation lag for impulse response function estimation
- `n.perm` number of cross-sectional permutations involved in impulse response function estimation

`var.order` order of the idiosyncratic VAR process; if a vector of integers is supplied, the order is chosen via `tuning`

`var.method` a string specifying the method to be adopted for idiosyncratic VAR process estimation; possible values are:

- `"lasso"` Lasso-type `l1`-regularised `M`-estimation
- `"ds"` Dantzig Selector-type constrained `l1`-minimisation

`var.args` a list specifying the tuning parameters required for estimating the idiosyncratic VAR process. It contains:

- `tuning` a string specifying the selection procedure for `var.order` and `lambda`; possible values are:
  - `"cv"` cross validation
  - `"ic"` information criterion
- `n.iter` maximum number of descent steps; applicable when `var.method = "lasso"`

- `tol` numerical tolerance for increases in the loss function; applicable when `var.method = "lasso"`
- `n.cores` number of cores to use for parallel computing, see makePSOCKcluster; applicable when `var.method = "ds"`

`var.threshold` whether to perform adaptive thresholding of VAR parameter estimator with threshold

`do.lrpc` whether to estimate the long-run partial correlation

`lrpc.adaptive` whether to use the adaptive estimation procedure

`tuning.args` a list specifying arguments for `tuning` for selecting the tuning parameters involved in VAR parameter and (long-run) partial correlation matrix estimation. It contains:

- `n.folds` if `tuning = "cv"`, number of folds
- `penalty` if `tuning = "ic"`, penalty multiplier between 0 and 1; if `penalty = NULL`, defaults to `1/(1+exp(dim(x)[1])/dim(x)[2]))`
- `path.length` number of regularisation parameter values to consider; a sequence is generated automatically based in this value
- `do.plot` whether to plot the output of the cross validation step

Details

See Barigozzi, Cho and Owens (2021) for further details. List arguments do not need to be specified with all list components; any missing entries will be filled in with the default argument.

Value

an S3 object of class `fnets`, which contains the following fields:

`q` number of factors

`spec` if `fm.restricted = FALSE` a list containing estimates of the spectral density matrices for `x`, common and idiosyncratic components

`acv` a list containing estimates of the autocovariance matrices for `x`, common and idiosyncratic components

`loadings` if `fm.restricted = TRUE`, factor loadings; if `fm.restricted = FALSE` and `q >= 1`, a list containing estimators of the impulse response functions (as an array of dimension `(p, q, trunc.lags + 2)`)

`factors` if `fm.restricted = TRUE`, factor series; else, common shocks (an array of dimension `(q, n)`)

`idio.var` a list containing the following fields:

- `beta` estimate of VAR parameter matrix; each column contains parameter estimates for the regression model for a given variable
- `Gamma` estimate of the innovation covariance matrix
- `lambda` regularisation parameter
- `convergence` returned when `var.method = "lasso"`; indicates whether a convergence criterion is met
- `var.order` VAR order

`lrpc`	see the output of par.lrpc
`mean.x`	if `center = TRUE`, returns a vector containing row-wise sample means of `x`; if `center = FALSE`, returns a vector of zeros
`var.method`	input parameter
`do.lrpc`	input parameter
`kern.bw`	input parameter

References

Barigozzi, M., Cho, H. & Owens, D. (2021) FNETS: Factor-adjusted network estimation and forecasting for high-dimensional time series. arXiv preprint arXiv:2201.06110.

Hallin, M. & Liška, R. (2007) Determining the number of factors in the general dynamic factor model. Journal of the American Statistical Association, 102(478), 603–617.

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

See Also

predict.fnets, plot.fnets

Examples

```
## Not run:
set.seed(123)
n <- 500
p <- 50
common <- sim.unrestricted(n, p)
idio <- sim.var(n, p)
x <- common$data + idio$data
out <- fnets(x,
  q = NULL, var.order = 1, var.method = "lasso", var.threshold = TRUE,
  do.lrpc = TRUE, tuning.args = list(n.folds = 1, path.length = 10, do.plot = TRUE)
)
pre <- predict(out, x, h = 1, common.method = "unrestricted")
plot(out, type = "granger", display = "network")
plot(out, type = "lrpc", display = "heatmap")

## End(Not run)
```

`fnets.factor.model` *Factor model estimation*

Description

Unrestricted and restricted factor model estimation

Usage

```
fnets.factor.model(
  x,
  center = TRUE,
  fm.restricted = FALSE,
  q = c("ic", "er"),
  pen.op = NULL,
  kern.bw = NULL,
 common.args = list(factor.var.order = NULL, max.var.order = NULL, trunc.lags = 20,
    n.perm = 10)
)
```

Arguments

`x`	input time series matrix, with each row representing a variable
`center`	whether to de-mean the input `x` row-wise
`fm.restricted`	whether to estimate a restricted factor model using static PCA
`q`	Either a string specifying the factor number selection method when `fm.restricted = TRUE`; possible values are: • `"ic"` information criteria of Hallin and Liška (2007) or Bai and Ng (2002), see factor.number • `"er"` eigenvalue ratio ; or the number of unrestricted factors.
`pen.op`	choice of the information criterion penalty, see hl.factor.number or bn.factor.number for further details
`kern.bw`	kernel bandwidth for dynamic PCA; defaults to `4 * floor((dim(x)[2]/log(dim(x)[2]))^(1/3)))`
`common.args`	a list specifying the tuning parameters required for estimating the impulse response functions and common shocks. It contains: • `factor.var.order` order of the blockwise VAR representation of the common component. If `factor.var.order = NULL`, it is selected blockwise by Schwarz criterion • `max.var.order` maximum blockwise VAR order for the Schwarz criterion • `trunc.lags` truncation lag for impulse response function estimation • `n.perm` number of cross-sectional permutations involved in impulse response function estimation

Details

See Barigozzi, Cho and Owens (2021) for further details.

Value

an S3 object of class `fm`, which contains the following fields:

`q`	number of factors

spec — if fm.restricted = FALSE a list containing estimates of the spectral density matrices for x, common and idiosyncratic components

acv — a list containing estimates of the autocovariance matrices for x, common and idiosyncratic components

loadings — if fm.restricted = TRUE, factor loadings; if fm.restricted = FALSE and q >= 1, a list containing estimators of the impulse response functions (as an array of dimension (p, q, trunc.lags + 2))

factors — if fm.restricted = TRUE, factor series; else, common shocks (an array of dimension (q, n))

mean.x — if center = TRUE, returns a vector containing row-wise sample means of x; if center = FALSE, returns a vector of zeros

References

Barigozzi, M., Cho, H. & Owens, D. (2021) Factor-adjusted network estimation and forecasting for high-dimensional time series. arXiv preprint arXiv:2201.06110.

Hallin, M. & Liška, R. (2007) Determining the number of factors in the general dynamic factor model. Journal of the American Statistical Association, 102(478), 603–617.

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

Examples

```
## Not run:
set.seed(123)
n <- 500
p <- 50
common <- sim.restricted(n, p)
x <- common$data
out <- fnets.factor.model(x, fm.restricted = TRUE)

## End(Not run)
```

---

fnets.var — *l1-regularised Yule-Walker estimation for VAR processes*

Description

Estimates the VAR parameter matrices via l1-regularised Yule-Walker estimation and innovation covariance matrix via constrained l1-minimisation.

Usage

```
fnets.var(
  x,
  center = TRUE,
  method = c("lasso", "ds"),
  lambda = NULL,
  var.order = 1,
 tuning.args = list(tuning = c("cv", "ic"), n.folds = 1, path.length = 10, do.plot =
    FALSE),
  var.threshold = FALSE,
  n.iter = 100,
  tol = 0,
  n.cores = min(parallel::detectCores() - 1, 3)
)
```

Arguments

`x`	input time series matrix, with each row representing a variable
`center`	whether to de-mean the input `x` row-wise
`method`	a string specifying the method to be adopted for VAR process estimation; possible values are: • `"lasso"` Lasso-type `l1`-regularised `M`-estimation • `"ds"` Dantzig Selector-type constrained `l1`-minimisation
`lambda`	regularisation parameter; if `lambda = NULL`, `tuning` is employed to select the parameter
`var.order`	order of the VAR process; if a vector of integers is supplied, the order is chosen via `tuning`

`tuning.args`	a list specifying arguments for `tuning` for selecting the regularisation parameter (and VAR order). It contains:
	• `tuning` a string specifying the selection procedure for `var.order` and `lambda`; possible values are:
	  – `"cv"` cross validation
	  – `"ic"` information criterion
	• `n.folds` if `tuning = "cv"`, number of folds
	• `penalty` if `tuning = "ic"`, penalty multiplier between 0 and 1; if `penalty = NULL`, defaults to `1/(1+exp(dim(x)[1])/dim(x)[2]))`
	• `path.length` number of regularisation parameter values to consider; a sequence is generated automatically based in this value
	• `do.plot` whether to plot the output of the cross validation step

`var.threshold`	whether to perform adaptive thresholding of VAR parameter estimator with threshold
`n.iter`	maximum number of descent steps; applicable when `method = "lasso"`
`tol`	numerical tolerance for increases in the loss function; applicable when `method = "lasso"`
`n.cores`	number of cores to use for parallel computing, see makePSOCKcluster; applicable when `method = "ds"`

Details

Further information can be found in Barigozzi, Cho and Owens (2021).

Value

a list which contains the following fields:

`beta`	estimate of VAR parameter matrix; each column contains parameter estimates for the regression model for a given variable
`Gamma`	estimate of the innovation covariance matrix
`lambda`	regularisation parameter
`convergence`	returned when `method = "lasso"`; indicates whether a convergence criterion is met
`var.order`	VAR order
`mean.x`	if `center = TRUE`, returns a vector containing row-wise sample means of `x`; if `center = FALSE`, returns a vector of zeros

References

Barigozzi, M., Cho, H. & Owens, D. (2021) FNETS: Factor-adjusted network estimation and forecasting for high-dimensional time series. arXiv preprint arXiv:2201.06110.

Examples

```
library(fnets)

set.seed(123)
n <- 500
p <- 50
idio <- sim.var(n, p)
x <- idio$data

fv <- fnets.var(x,
  center = TRUE, method = "lasso", var.order = 1,
  tuning.args = list(tuning = "cv", n.folds = 1, path.length = 10, do.plot = TRUE)
)
norm(fv$beta - t(idio$A), "F") / norm(t(idio$A), "F")
```

`idio.predict` *Forecasting idiosyncratic VAR process*

Description

Produces forecasts of the idiosyncratic VAR process for a given forecasting horizon by estimating the best linear predictors

Usage

```
idio.predict(object, x, cpre, h = 1)
```

Arguments

object	`fnets` object
x	input time series matrix, with each row representing a variable
cpre	output of common.predict
h	forecast horizon

Value

a list containing

is	in-sample estimator of the idiosyncratic component
fc	forecasts of the idiosyncratic component for a given forecasting horizon `h`
h	forecast horizon

References

Barigozzi, M., Cho, H. & Owens, D. (2021) FNETS: Factor-adjusted network estimation and forecasting for high-dimensional time series. arXiv preprint arXiv:2201.06110.

Examples

```
set.seed(123)
n <- 500
p <- 50
common <- sim.unrestricted(n, p)
idio <- sim.var(n, p)
x <- common$data + idio$data
out <- fnets(x, q = NULL, var.order = 1, var.method = "lasso", do.lrpc = FALSE)
cpre <- common.predict(out, x, h = 1, r = NULL)
ipre <- idio.predict(out, x, cpre, h = 1)
```

---

par.lrpc *Parametric estimation of long-run partial correlations of factor-adjusted VAR processes*

Description

Returns a parametric estimate of long-run partial correlations of the VAR process from the VAR parameter estimates and the inverse of innovation covariance matrix obtained via constrained `l1`-minimisation.

Usage

```
par.lrpc(
  object,
  x,
  eta = NULL,
  tuning.args = list(n.folds = 1, path.length = 10, do.plot = FALSE),
  lrpc.adaptive = FALSE,
  eta.adaptive = NULL,
  do.correct = TRUE,
  n.cores = min(parallel::detectCores() - 1, 3)
)
```

Arguments

`object`	`fnets` object
`x`	input time series matrix; with each row representing a variable
`eta`	regularisation parameter; if `eta = NULL`, it is selected by cross validation
`tuning.args`	a list specifying arguments for the cross validation procedure for selecting the tuning parameter involved in long-run partial correlation matrix estimation. It contains: • `n.folds` number of folds • `path.length` number of regularisation parameter values to consider; a sequence is generated automatically based in this value • `do.plot` whether to plot the output of the cross validation step
`lrpc.adaptive`	whether to use the adaptive estimation procedure
`eta.adaptive`	regularisation parameter for Step 1 of the adaptive estimation procedure; if `eta.adaptive = NULL`, defaults to `2 * sqrt(log(dim(x)[1])/dim(x)[2])`
`do.correct`	whether to correct for any negative entries in the diagonals of the inverse of long-run covariance matrix
`n.cores`	number of cores to use for parallel computing, see makePSOCKcluster

Details

See Barigozzi, Cho and Owens (2021) for further details, and Cai, Liu and Zhou (2016) for further details on the adaptive estimation procedure.

Value

a list containing

`Delta`	estimated inverse of the innovation covariance matrix
`Omega`	estimated inverse of the long-run covariance matrix
`pc`	estimated innovation partial correlation matrix
`lrpc`	estimated long-run partial correlation matrix
`eta`	regularisation parameter
`lrpc.adaptive`	was the adaptive procedure used

References

Barigozzi, M., Cho, H. & Owens, D. (2021) FNETS: Factor-adjusted network estimation and forecasting for high-dimensional time series.

Cai, T. T., Liu, W., & Zhou, H. H. (2016). Estimating sparse precision matrix: Optimal rates of convergence and adaptive estimation. The Annals of Statistics, 44(2), 455-488.

Examples

```
## Not run:
set.seed(123)
n <- 500
p <- 50
common <- sim.unrestricted(n, p)
idio <- sim.var(n, p)
x <- common$data + idio$data
out <- fnets(x, q = NULL, var.method = "lasso", do.lrpc = FALSE)
plrpc <- par.lrpc(out, x, tuning.args = list(n.folds = 1, path.length = 10, do.plot = TRUE))
out$lrpc <- plrpc
out$do.lrpc <- TRUE
plot(out, type = "pc", display = "network", threshold = .05)
plot(out, type = "lrpc", display = "heatmap", threshold = .05)

## End(Not run)
```

`plot.fnets` *Plotting the networks estimated by fnets*

Description

Plotting method for S3 objects of class `fnets`. Produces a plot visualising three networks underlying factor-adjusted VAR processes: (i) directed network representing Granger causal linkages, as given by estimated VAR transition matrices summed across the lags, (ii) undirected network representing contemporaneous linkages after accounting for lead-lag dependence, as given by partial correlations of VAR innovations, (iii) undirected network summarising (i) and (ii) as given by long-run partial correlations of VAR processes.

Usage

```
## S3 method for class 'fnets'
plot(
  x,
  type = c("granger", "pc", "lrpc"),
  display = c("network", "heatmap"),
  names = NA,
  groups = NA,
  threshold = 0,
  ...
)
```

Arguments

`x`	`fnets` object
`type`	a string specifying which of the above three networks (i)–(iii) to visualise; possible values are • `"granger"` directed network representing Granger causal linkages • `"pc"` undirected network representing contemporaneous linkages; available when `x$do.lrpc = TRUE` • `"lrpc"` undirected network summarising Granger causal and contemporaneous linkages; available when `x$do.lrpc = TRUE`
`display`	a string specifying how to visualise the network; possible values are: • `"network"` as an `igraph` object, see plot.igraph • `"heatmap"` as a heatmap, see imagePlot
`names`	a character vector containing the names of the vertices
`groups`	an integer vector denoting any group structure of the vertices
`threshold`	if `threshold > 0`, hard thresholding is performed on the matrix giving rise to the network of interest
`...`	additional arguments

Details

See Barigozzi, Cho and Owens (2021) for further details.

References

Barigozzi, M., Cho, H. & Owens, D. (2021) FNETS: Factor-adjusted network estimation and forecasting for high-dimensional time series. arXiv preprint arXiv:2201.06110.

See Also

fnets

---

predict.fm — *Forecasting for factor models*

Description

Produces forecasts of the data for a given forecasting horizon by estimating the best linear predictors of the common component

Usage

```
## S3 method for class 'fm'
predict(object, x, h = 1, forecast.restricted = TRUE, r = c("ic", "er"), ...)
```

Arguments

`object`	`fm` object
`x`	input time series matrix, with each row representing a variable
`h`	forecasting horizon
`forecast.restricted`	whether to forecast using a restricted or unrestricted, blockwise VAR representation of the common component
`r`	number of restricted factors, or a string specifying the factor number selection method when `forecast.restricted = TRUE`; possible values are: • `"ic"` information criteria of Bai and Ng (2002) • `"er"` eigenvalue ratio
`...`	not used

Value

a list containing

`is`	in-sample predictions
`forecast`	forecasts for the given forecasting horizon
`r`	factor number

References

Barigozzi, M., Cho, H. & Owens, D. (2021) Factor-adjusted network estimation and forecasting for high-dimensional time series. arXiv preprint arXiv:2201.06110.

Ahn, S. C. & Horenstein, A. R. (2013) Eigenvalue ratio test for the number of factors. Econometrica, 81(3), 1203–1227.

See Also

fnets.factor.model, common.predict

`predict.fnets` *Forecasting by fnets*

Description

Produces forecasts of the data for a given forecasting horizon by separately estimating the best linear predictors of common and idiosyncratic components

Usage

```
## S3 method for class 'fnets'
predict(object, x, h = 1, forecast.restricted = TRUE, r = c("ic", "er"), ...)
```

Arguments

`object`	`fnets` object
`x`	input time series matrix, with each row representing a variable
`h`	forecasting horizon
`forecast.restricted`	whether to forecast using a restricted or unrestricted, blockwise VAR representation of the common component
`r`	number of restricted factors, or a string specifying the factor number selection method when `forecast.restricted = TRUE`; possible values are: • `"ic"` information criteria of Bai and Ng (2002) • `"er"` eigenvalue ratio
`...`	not used

Value

a list containing

`forecast`	forecasts for the given forecasting horizon
`common.pred`	a list containing forecasting results for the common component
`idio.pred`	a list containing forecasting results for the idiosyncratic component
`mean.x`	`mean.x` argument from `object`

References

Barigozzi, M., Cho, H. & Owens, D. (2021) FNETS: Factor-adjusted network estimation and forecasting for high-dimensional time series. arXiv preprint arXiv:2201.06110.

Ahn, S. C. & Horenstein, A. R. (2013) Eigenvalue ratio test for the number of factors. Econometrica, 81(3), 1203–1227.

See Also

fnets, common.predict, idio.predict

`sim.restricted` *Simulate data from a restricted factor model*

Description

Simulate the common component following an unrestricted factor model that admits a restricted representation; see the model (C2) in the reference.

Usage

```
sim.restricted(n, p, q = 2, heavy = FALSE)
```

Arguments

`n`	sample size
`p`	dimension
`q`	number of unrestricted factors; number of restricted factors is given by `2 * q`
`heavy`	if `heavy = FALSE`, common shocks are generated from `rnorm` whereas if `heavy = TRUE`, from `rt` with `df = 5` and then scaled by `sqrt(3 / 5)`

Value

a list containing

`data`	generated series
`q`	number of factors
`r`	number of restricted factors

References

Barigozzi, M., Cho, H. & Owens, D. (2021) FNETS: Factor-adjusted network estimation and forecasting for high-dimensional time series.

Examples

```
common <- sim.restricted(500, 50)
```

`sim.unrestricted` *Simulate data from an unrestricted factor model*

Description

Simulate the common component following an unrestricted factor model that does not admit a restricted representation; see the model (C1) in the reference.

Usage

```
sim.unrestricted(n, p, q = 2, heavy = FALSE)
```

Arguments

`n`	sample size
`p`	dimension
`q`	number of unrestricted factors
`heavy`	if `heavy = FALSE`, common shocks are generated from `rnorm` whereas if `heavy = TRUE`, from `rt` with `df = 5` and then scaled by `sqrt(3 / 5)`

Value

a list containing

`data`	generated series
`q`	number of factors

References

Barigozzi, M., Cho, H. & Owens, D. (2021) FNETS: Factor-adjusted network estimation and forecasting for high-dimensional time series. arXiv preprint arXiv:2201.06110

Examples

```
common <- sim.unrestricted(500, 50)
```

`sim.var` *Simulate a VAR(1) process*

Description

Simulate a VAR(1) process; see the reference for the generation of the transition matrix.

Usage

```
sim.var(n, p, Gamma = diag(1, p), heavy = FALSE)
```

Arguments

`n`	sample size
`p`	dimension
`Gamma`	innovation covariance matrix; ignored if `heavy = TRUE`
`heavy`	if `heavy = FALSE`, common shocks are generated from `rnorm` whereas if `heavy = TRUE`, from `rt` with `df = 5` and then scaled by `sqrt(3 / 5)`

Value

a list containing

`data`	generated series
`A`	transition matrix
`Gamma`	innovation covariance matrix

References

Barigozzi, M., Cho, H. & Owens, D. (2021) FNETS: Factor-adjusted network estimation and forecasting for high-dimensional time series.

Examples

```
idio <- sim.var(500, 50)
```

Index