Package 'fnets'

January 12, 2022

Type Package

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

Version 0.1.0

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 (>= 3.1.2)

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

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

R topics documented:

common.predict 2
fnets 3
fnets.var 6
hl.factor.number 8
idio.predict 9
npar.lrpc 10
par.lrpc 11
plot.fnets 13
predict.fnets 14

sim.common1 15
sim.common2 16
sim.var 17

Index **19**

`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,
  common.method = c("restricted", "unrestricted"),
  r = NULL
)
```

Arguments

`object`	`fnets` object
`x`	input time series matrix, with each row representing a variable
`h`	forecasting horizon
`common.method`	a string specifying the method for common component forecasting; possible values are: • `"restricted"` performs forecasting under a restrictive static factor model • `"unrestricted"` performs forecasting under an unrestrictive, blockwise VAR representation of the common component
`r`	number of static factors; if `common.method = "restricted"` and `r = NULL`, it is estimated as the maximiser of the ratio of the successive eigenvalues of the estimate of the common component covariance matrix, see Ahn and Horenstein (2013)

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`	static factor number
`h`	forecast horizon

References

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

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.common1(n, p)
idio <- sim.var(n, p)
x <- common$data + idio$data
out <- fnets(x, q = NULL, idio.var.order = 1, idio.method = "lasso", lrpc.method = "none")
cpre <- common.predict(out, x, h = 1, common.method = 'restricted', r = NULL)
ipre <- idio.predict(out, x, cpre, h = 1)
```

---

fnets *Factor-adjusted network estimation*

Description

Operating under 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,
  q = NULL,
  ic.op = 5,
  kern.const = 4,
  common.args = list(var.order = NULL, max.var.order = NULL, trunc.lags = 20, n.perm =
    10),
  idio.var.order = 1,
  idio.method = c("lasso", "ds"),
```

```
  idio.args = list(n.iter = 100, tol = 1e-05, n.cores = min(parallel::detectCores() -
    1, 3)),
  lrpc.method = c("par", "npar", "none"),
  cv.args = list(n.folds = 1, 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
- `q` — number of factors. If `q = NULL`, the factor number is estimated by an information criterion-based approach of Hallin and Liška (2007), see hl.factor.number for further details
- `ic.op` — choice of the information criterion, see hl.factor.number for further details
- `kern.const` — constant multiplied to `floor((dim(x)[2]/log(dim(x)[2]))^(1/3)))` which determines the kernel bandwidth for dynamic PCA
- `common.args` — a list specifying the tuning parameters required for estimating the impulse response functions and common shocks. It contains:
  - `var.order` order of the blockwise VAR representation of the common component. If `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 impluse response function estimation
- `idio.var.order` — order of the idiosyncratic VAR process; if a vector of integers is supplied, the order is chosen via cross validation
- `idio.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
- `idio.args` — a list specifying the tuning parameters required for estimating the idiosyncratic VAR process. It contains:
  - `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"`
- `lrpc.method` — a string specifying the type of estimator for long-run partial correlation matrix estimation; possible values are:
  - `"par"` parametric estimator based on the VAR model assumption
  - `"npar"` nonparametric estimator from inverting the long-run covariance matrix of the idiosyncratic component via constrained `l1`-minimisation
  - `"none"` do not estimate the long-run partial correlation matrix

`cv.args` a list specifying arguments for the cross validation procedures for selecting the tuning parameters involved in VAR parameter and (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

Details

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

Value

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

`q` number of factors

`spec` 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

`common.irf` if `q >= 1`, a list containing estimators of the impulse response functions (as an array of dimension `(p,q,trunc.lags + 2)`) and common shocks (an array of dimension `(q,n)`) for the common component

`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
- `var.order` VAR order

`lrpc` see the output of par.lrpc if `lrpc.method = 'par'` and that of npar.lrpc if `lrpc.method = 'npar'`

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

`idio.method` input parameter

`lrpc.method` input parameter

`kern.const` input parameter

References

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

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.

See Also

predict.fnets, plot.fnets

Examples

```
## Not run:
set.seed(123)
n <- 500
p <- 50
common <- sim.common1(n, p)
idio <- sim.var(n, p)
x <- common$data + idio$data
out <- fnets(x, q = NULL, idio.var.order = 1, idio.method = "lasso",
lrpc.method = "par", cv.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', threshold = .05)
plot(out, type = 'lrpc', display = 'heatmap', threshold = .05)

## 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,
  cv.args = list(n.folds = 1, path.length = 10, do.plot = FALSE),
  n.iter = 100,
  tol = 1e-05,
  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, cross validation 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 cross validation
cv.args	a list specifying arguments for the cross validation procedure for selecting the regularisation parameter (and VAR order). 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
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
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 analysis for high-dimensional time series.

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,
                cv.args = list(n.folds = 1, path.length = 10, do.plot = TRUE))
norm(fv$beta - t(idio$A), 'F')/norm(t(idio$A), 'F')
```

`hl.factor.number` *Factor number estimator of Hallin and Liška (2007)*

Description

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

Usage

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

Arguments

`x`	input time series matrix, with each row representing a variable
`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`	integer representing the kernel bandwidth
`w`	vector of length `2 * mm + 1` containing symmetric weights; if `w = NULL`, default weights are generated using the Bartlett kernel and `mm`
`do.plot`	whether to plot the values of six information criteria
`center`	whether to de-mean the input `x` row-wise

Details

See Hallin and Liška (2007) for further details.

Value

a list containing

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

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.

Examples

```
library(fnets)

set.seed(123)
n <- 500
p <- 50
common <- sim.common2(n, p)
idio <- sim.var(n, p)
x <- common$data * apply(idio$data, 1, sd)/apply(common$data, 1, sd) + idio$data

hl <- hl.factor.number(x, q.max = NULL, mm = floor(4 * (n/log(n))^(1/3)), do.plot = TRUE)
hl$q
```

`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 analysis for high-dimensional time series.

Examples

```
set.seed(123)
n <- 500
p <- 50
common <- sim.common1(n, p)
idio <- sim.var(n, p)
x <- common$data + idio$data
out <- fnets(x, q = NULL, idio.var.order = 1, idio.method = "lasso", lrpc.method = "none")
cpre <- common.predict(out, x, h = 1, common.method = 'restricted', r = NULL)
ipre <- idio.predict(out, x, cpre, h = 1)
```

---

npar.lrpc — *Nonparametric estimation of long-run partial correlations of factor-adjusted VAR processes*

Description

Returns a nonparametric estimate of long-run partial correlations of the VAR process from the inverse of long-run covariance matrix obtained via constrained `l1`-minimisation.

Usage

```
npar.lrpc(
  object,
  x,
  eta = NULL,
  cv.args = list(n.folds = 1, path.length = 10, do.plot = FALSE),
  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
`cv.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

`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

Value

a list containing

`Omega`	estimated inverse of the long-run covariance matrix
`lrpc`	estimated long-run partial correlation matrix
`eta`	regularisation parameter

Examples

```
## Not run:
set.seed(123)
n <- 500
p <- 50
common <- sim.common1(n, p)
idio <- sim.var(n, p)
x <- common$data + idio$data
out <- fnets(x, q = NULL, idio.method = 'lasso', lrpc.method = 'none')
nlrpc <- npar.lrpc(out, x, cv.args = list(n.folds = 1, path.length = 10, do.plot = TRUE))
out$lrpc <- nlrpc
out$lrpc.method <- 'npar'
plot(out, type = 'lrpc', display = 'heatmap', threshold = .05)

## End(Not run)
```

`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,
  cv.args = list(n.folds = 1, path.length = 10, do.plot = FALSE),
  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
`cv.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
`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.

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

References

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

Examples

```
## Not run:
set.seed(123)
n <- 500
p <- 50
common <- sim.common1(n, p)
idio <- sim.var(n, p)
x <- common$data + idio$data
out <- fnets(x, q = NULL, idio.method = 'lasso', lrpc.method = 'none')
plrpc <- par.lrpc(out, x, cv.args = list(n.folds = 1, path.length = 10, do.plot = TRUE))
out$lrpc <- plrpc
out$lrpc.method <- 'par'
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 aggregated 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$lrpc.method = "par"
- "lrpc" undirected network summarising Granger causal and contemporaneous linkages; available when x$lrpc.method = "par" or x$lrpc.method = "npar"

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 analysis for high-dimensional time series.

See Also

fnets

---

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,
  common.method = c("restricted", "unrestricted"),
  r = NULL,
  ...
)
```

Arguments

`object`	`fnets` object
`x`	input time series matrix, with each row representing a variable
`h`	forecasting horizon
`common.method`	a string specifying the method for common component forecasting; possible values are: • `"restricted"` performs forecasting under a restrictive static factor model • `"unrestricted"` performs forecasting under an unrestrictive, blockwise VAR representation of the common component
`r`	number of static factors; if `common.method = "restricted"` and `r = NULL`, it is estimated as the maximiser of the ratio of the successive eigenvalues of the estimate of the common component covariance matrix, see Ahn and Horenstein (2013)
`...`	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 analysis for high-dimensional time series.

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.common1` *Simulate data from a dynamic factor model*

Description

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

Usage

```
sim.common1(n, p, q = 2, heavy = FALSE)
```

Arguments

n	sample size
p	dimension
q	number of dynamic 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 analysis for high-dimensional time series.

Examples

```
common <- sim.common1(500, 50)
```

---

sim.common2 *Simulate data from a static factor model*

Description

Simulate the common component following a dynamic factor model that admits a static representation; see the model (C2) in the reference.

Usage

```
sim.common2(n, p, q = 2, heavy = FALSE)
```

Arguments

n	sample size
p	dimension
q	number of dynamic factors; number of static 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 static factors

References

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

Examples

```
common <- sim.common2(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 analysis for high-dimensional time series.

Examples

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

Index

common.predict, 2, *9*, *15*

fnets, 3, *14, 15*
fnets.var, 6

hl.factor.number, *4*, 8

idio.predict, 9, *15*
imagePlot, *14*

makePSOCKcluster, *4*, *7*, *11, 12*

npar.lrpc, *5*, 10

par.lrpc, *5*, 11
plot.fnets, *6*, 13
plot.igraph, *14*
predict.fnets, *6*, 14

sim.common1, 15
sim.common2, 16
sim.var, 17