Package 'fnets'

December 6, 2021

Type Package

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

Version 0.1.0

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

Description Used for fitting the factor-adjusted VAR model proposed in Barigozzi, Cho and Owens (2021),where strong cross-sectional dependence is accounted for by factors and the remaining idiosyncratic dependence is modelled by a sparse vector autoregression. Methods for estimating the Granger, Contemporaneous, and Long-Run Partial Correlation networks are provided, as well as forecasting methods for both components.

Depends R (>= 2.10)

Imports igraph,
lpSolve,
foreach,
mvtnorm,
doParallel,
RColorBrewer,
fields

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

R topics documented:

`common.predict` *Prediction for the factor-driven common component*

Description

Predicts common component from a `fnets` object for new data

Usage

```
common.predict(object, x, h = 1, common.method = c("static", "var"), r = NULL)
```

Arguments

`object`	`fnets` object
`x`	input time series matrix, with each row representing a time series
`h`	forecast horizon
`common.method`	which of "static" or "var" to forecast the common component with
`r`	factor number, if r=NULL this is selected using the maximal eigenratio

Value

A list containing

- `'is'` x in-sample estimation
- `'fc'` x forecast
- `'r'` factor number
- `'h'` forecast horizon

References

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

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

```
require(fnets)
model <- fnets(sample.data, q=2, idio.method = "lasso")
pr <- predict(model,sample.data, common.method = "static")
cpre <- common.predict(model,sample.data, common.method = "static")
ip <- idio.predict(model,sample.data, cpre)
```

`dyn.pca` *Dynamic PCA*

Description

Performs principal components analysis of the autocovariance matrices.

Usage

```
dyn.pca(xx, q = NULL, ic.op = 4, kern.bandwidth.const = 4)
```

Arguments

`xx`	centred input time series matrix, with each row representing a time series
`q`	the number of factors, if q=NULL this is selected by the information criterion-based estimator of Hallin and Liska (2007)
`ic.op`	an index number for the information criterion (1 to 6)
`kern.bandwidth.const`	constant to determine bandwidth size

Value

A list containing

- `'q'` number of factors
- `'spec'` Spectral density matrices
- `'acv'` Autocovariance matrices
- `'kern.bandwidth.const'` Constant to determine bandwidth size

Examples

```
dyn.pca(sample.data, q=2)
```

`fnets` *Factor-adjusted network analysis*

Description

This function estimates the spectral density and autocovariance matrices of the common and the idiosyncratic components, impulse response function and common shocks, and (sparse) VAR transition matrix and innovation covariance matrix.

Usage

```
fnets(
  x,
  q = NULL,
  ic.op = 4,
  kern.bandwidth.const = 4,
  common.var.args = list(var.order = 1, max.var.order = NULL, trunc.lags = 20, n.perm =
    10),
  idio.var.order = 1,
  idio.method = c("ds", "lasso"),
  idio.cv.args = list(n.folds = 1, path.length = 10, symmetric = "min", cv.plot = TRUE),
  center = TRUE
)
```

Arguments

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

`q` the number of factors, if q=NULL this is selected by the information criterion-based estimator of Hallin and Liska (2007)

`ic.op` an index number for the information criterion

`kern.bandwidth.const` constant to determine bandwidth size

`common.var.args` A list specifying the estimator for the common component. This contains:

- `'var.order'` the order of the VAR model, if NULL then selected blockwise by BIC
- `'max.var.order'` the maximum order of the VAR model for the BIC to consider
- `'trunc.lags'` the order of the MA representation
- `'n.perm'` number of cross-sectional permutations

`idio.var.order` order of idiosyncratic VAR model `idio.method` A string specifying the type of l1-regularised estimator for the idiosyncratic VAR matrix, possible values are:

- `'lasso'` Lasso estimator
- `'ds'` Dantzig Selector

`idio.cv.args` A list specifying arguments to the cross-validation (CV) procedure for the idiosyncratic VAR. This contains:

- `'n.folds'` number of folds
- `'path.length'` number of lambda values to consider
- `'symmetric'` symmetrisation method for Gamma matrix
- `'cv.plot'` Boolean selecting whether to plot the CV curve

`center` demean the input `x`

Details

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

Value

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

- `'q'` Number of factors
- `'spec'` Spectral density matrices
- `'acv'` Autocovariance matrices
- `'common.var'` Estimated common component
- `'idio.var'` Estimated idiosyncratic component
- `'mean.x'` Removed means of `x`
- `'kern.bandwidth.const'` Constant to determine bandwidth size

References

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

Examples

```
fnets(sample.data, q=2, idio.method = "lasso")
```

`hl.factor.number` *Factor number estimator of Hallin and Liska (2011)*

Description

Selects the factor number q based on 6 information criteria

Usage

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

Arguments

`xx`	centred input time series matrix, with each row representing a time series
`q.max`	the maximum number of factors to consider
`mm`	bandwidth scalar
`w`	weight vector, defaults to Bartlett weights determined by `mm`
`do.plot`	return a plot of the information criteria

Value

A list containing

- `'q.hat'` Estimated factor numbers corresponding to each criterion
- `'Gamma_x'` Autocovariance of x
- `'Sigma_x'` Spectral density of x
- `'sv'` singular value decomposition of Sigma_x

hl.factor.number(sample.data,6, 10)

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.

`idio.cv` *Cross-validation for l1-regularised VAR estimation*

Description

Selects the prediction-optimal regularisation parameter for the estimation of the idiosyncratic VAR

Usage

```
idio.cv(
  xx,
  lambda.max = NULL,
  var.order = 1,
  idio.method = c("lasso", "ds"),
  path.length = 10,
  n.folds = 1,
  q = 0,
  kern.bandwidth.const = 4,
  cv.plot = TRUE
)
```

Arguments

`xx`	centred input time series matrix, with each row representing a time series
`lambda.max`	maximum regularisation parameter, if NULL this is set to the smallest which sets all entries to 0
`var.order`	vector of VAR orders to consider
`idio.method`	estimation method, one of "lasso" or "ds"
`path.length`	number of regularisation parameters to consider
`n.folds`	number of CV folds
`q`	factor number
`kern.bandwidth.const`	constant to determine bandwidth size
`cv.plot`	return a plot of the CV error against regularisation parameters, stratified by VAR order

Details

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

Value

A list which contains the following fields:

- 'lambda' minimising argument
- 'var.order' minimising order
- 'cv.error' matrix of errors
- 'lambda.path' candidate lambda values

References

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

Examples

```
idio.cv(sample.data, idio.method = "lasso", q=2)
```

---

idio.predict — *Prediction for the idiosyncratic VAR process*

Description

Predicts idiosyncratic components from a `fnets` object for new data

Usage

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

Arguments

`object`	`fnets` object
`x`	input time series matrix, with each row representing a time series
`cpre`	estimated common component
`h`	forecast horizon

Value

A list containing

- 'is' in-sample estimation
- 'fc' forecast
- 'h' forecast horizon

References

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

Examples

```
require(fnets)
model <- fnets(sample.data, q=2, idio.method = "lasso")
pr <- predict(model,sample.data, common.method = "static")
cpre <- common.predict(model,sample.data, common.method = "static")
ip <- idio.predict(model,sample.data, cpre)
```

---

make.gg — *Construct* gg *and* GG *matrices for Yule-Walker estimation*

Description

Construct gg and GG matrices for Yule-Walker estimation

Usage

```
make.gg(acv, d)
```

Arguments

acv	autocovariance array
d	order of VAR

Examples

```
require(fnets)
require(doParallel)
require(lpSolve)
dpca <- dyn.pca(sample.data, q=2)
mg <- make.gg(dpca$acv$Gamma_i,1)
vl <- var.lasso(mg$GG,mg$gg, .1)
vd <- var.dantzig(mg$GG,mg$gg, .1, n.cores = 1)
```

---

nonpar.lrpc — *Nonparametric partial coherence matrix estimation*

Description

Returns a non-parametric estimate of the partial coherence matrix, possibly using cross-validation

Usage

```
nonpar.lrpc(
  object,
  x,
  eta = NULL,
  symmetric = c("min", "max", "avg", "none"),
  lrpc.cv.args = list(n.folds = 1, path.length = 10),
  n.cores = min(parallel::detectCores() - 1, 3)
)
```

Arguments

`object`	`fnets` object
`x`	input time series matrix, with each row representing a time series
`eta`	regularisation parameter, if NULL this is selected by cross-validation
`symmetric`	type of symmetry to enforce on output, one of 'min', 'max', 'avg', 'none'
`lrpc.cv.args`	A list specifying arguments to the cross-validation (CV) procedure containing: • `'n.folds'` number of folds • `'path.length'` number of lambda values to consider
`n.cores`	number of cores to use for parallel computing

Value

A list containing

- `'Omega'` estimated partial coherence matrix
- `'eta'` regularisation parameter

References

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

Examples

```
#nonpar.lrpc
require(doParallel)
require(lpSolve)
model <- fnets(sample.data, q=2, idio.method = "lasso")
nonpar.lrpc(model, sample.data, 1, n.cores = 1)
```

param.lrpc *Parametric partial coherence matrix estimation*

Description

Returns a parametric estimate of the partial coherence matrix, possibly using cross-validation

Usage

```
param.lrpc(
  object,
  x,
  eta = NULL,
  symmetric = c("min", "max", "avg", "none"),
  lrpc.cv.args = list(n.folds = 1, path.length = 10),
  n.cores = min(parallel::detectCores() - 1, 3)
)
```

Arguments

`object`	`fnets` object
`x`	input time series matrix, with each row representing a time series
`eta`	regularisation parameter, if NULL this is selected by cross-validation
`symmetric`	type of symmetry to enforce on output, one of 'min', 'max', 'avg', 'none'
`lrpc.cv.args`	A list specifying arguments to the cross-validation (CV) procedure containing: • `'n.folds'` number of folds • `'path.length'` number of lambda values to consider
`n.cores`	number of cores to use for parallel computing

Value

A list containing

- `'Omega'` estimated partial coherence matrix
- `'eta'` regularisation parameter

References

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

Examples

```
#param.lrpc
require(doParallel)
require(lpSolve)
model <- fnets(sample.data, q=2, idio.method = "lasso")
param.lrpc(model, sample.data, 1, n.cores = 1)
```

plot.fnets *Plot fnets object*

Description

plots the idiosyncratic component of the fnets object as a Granger causal network, either as a network graph or a heatmap

Usage

```
## S3 method for class 'fnets'
plot(
  object,
  type = "network",
  names = NULL,
  groups = NULL,
  threshold = 0,
  size = NULL,
  ...
)
```

Arguments

`object`	fnets object
`type`	whether to plot a "network" or "heatmap"
`names`	character vector of node names
`groups`	integer vector denoting groups for "network" plots
`threshold`	sets all elements less than this in absolute value to 0
`size`	which type of degree to use for node size in "network" plots, one of "all", "out", "in", "total"
`...`	additional arguments

References

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

plot.fnets.lrpc   *Plot fnets.lrpc object*

Description

plots the fnets.lrpc object as a Long-Run Partial Correlation network, and if available as a Contemporaneous network, either as a network graph or a heatmap

Usage

```
## S3 method for class 'fnets.lrpc'
plot(
  object,
  type = "network",
  names = NULL,
  groups = NULL,
  threshold = 0,
  size = NULL,
  ...
)
```

Arguments

`object`	fnets.lrpc object
`type`	whether to plot a "network" or "heatmap"
`names`	character vector of node names
`groups`	integer vector denoting groups for "network" plots
`threshold`	sets all elements less than this in absolute value to 0
`size`	which type of degree to use for node size in "network" plots, one of "all", "out", "in", "total"
`...`	additional arguments

References

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

predict.fnets *Prediction by fnets*

Description

Predicts common and idiosyncratic components from a `fnets` object for new data

Usage

```
## S3 method for class 'fnets'
predict(object, x, h = 1, common.method = c("static", "var"), r = NULL)
```

Arguments

`object`	`fnets` object
`x`	input time series matrix, with each row representing a time series
`h`	forecast horizon
`common.method`	which of "static" or "var" to forecast the common component with
`r`	factor number, if r=NULL this is selected using the maximal eigenratio

Value

A list containing

- `'fitted'` `x` in-sample estimation
- `'forecast'` `x` forecast
- `'common.pred'` Prediction for the factor-driven common component
- `'idio.pred'` Prediction for the idiosyncratic component
- `'x.mean'` removed mean of `x`

References

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

Examples

```
require(fnets)
model <- fnets(sample.data, q=2, idio.method = "lasso")
pr <- predict(model,sample.data, common.method = "static")
cpre <- common.predict(model,sample.data, common.method = "static")
ip <- idio.predict(model,sample.data, cpre)
```

`sample.data` *Simulated example common+idiosyncratic time series*

Description

Example ‘p=100‘, ‘n=200‘ time series, where the common component is generated using Model 1 and the idiosyncratic component is generated with a sparse VAR.

Usage

```
sample.data
```

Format

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

`sim.factor.M1` *Simulate data from a dynamic factor model (Model 1)*

Description

Simulate data from a dynamic factor model (Model 1)

Usage

```
sim.factor.M1(n, p, q = 2, r = 4, do.scale = T, loadings = NULL, K = NULL)
```

Arguments

`n`	sample size
`p`	number of series
`q`	dynamic dimension (default 2)
`r`	factor number (default 4)
`do.scale`	scale the output (default `TRUE`)
`loadings`	loading matrix, dimension p by r (default null)
`K`	transition matrix, dimension r by q (default null)

Value

A list containing

- `'data'` generated series
- `'shocks'` q-dimensional shock series
- `'factors'` r-dimensional factor series
- `'K'` transition matrix
- `'loadings'` factor loadings

References

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

Examples

```
sim.factor.M1(100,10)
```

`sim.factor.M2` *Simulate data from a static factor model (Model 2)*

Description

Simulate data from a static factor model (Model 2)

Usage

```
sim.factor.M2(
  n,
  p,
  trunc.lags = 20,
  do.scale = T,
  a1 = NULL,
  a2 = NULL,
  alpha1 = NULL,
  alpha2 = NULL
)
```

Arguments

`n`	sample size
`p`	number of series
`trunc.lags`	lag for moving average representation
`do.scale`	scale the output (default `TRUE`)
`a1, a2, alpha1, alpha2`	generative parameters (default null, see reference)

Value

A list containing

- `'data'` generated series
- `'shocks'` 2-dimensional shock series
- `'a1','a2','alpha1','alpha2'` generative parameters

References

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

Examples

```
sim.factor.M2(100,10)
```

`sim.idio` *Simulate data from a (sparse) VAR(1) model*

Description

Simulate data from a (sparse) VAR(1) model

Usage

```
sim.idio(
  n,
  p,
  A = NULL,
  cov = diag(1, p),
  prob = 1/p,
  two.norm = NULL,
  do.scale = T
)
```

Arguments

`n`	sample size
`p`	number of series
`A`	transition matrix, dimension p by p (default null)
`cov`	generative covariance matrix (default identity)
`prob`	probability of an edge existing in the transition matrix, if `A` is `NULL`
`two.norm`	target 2-norm to scale A by (default NULL)
`do.scale`	scale the output (default `TRUE`)

Value

A list containing

- `'data'` generated series
- `'A'` transition matrix

References

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

Examples

```
sim.idio(100,10, A=diag(0.3, 10))
```

`var.dantzig` *Dantzig selector-type Yule-Walker estimation for VAR processes*

Description

Returns parameter estimates for the idiosyncratic VAR and the corresponding Gamma matrix

Usage

```
var.dantzig(
  GG,
  gg,
  lambda,
  symmetric = "min",
  n.cores = min(parallel::detectCores() - 1, 3)
)
```

Arguments

`GG, gg`	output from `make.gg`
`lambda`	regularisation parameter
`symmetric`	type of symmetry to enforce on `Gamma`, one of 'min', 'max', 'avg', 'none'
`n.cores`	number of cores to use for parallel computing

Details

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

Value

A list which contains the following fields:

- beta: VAR parameters
- lambda: regularisation parameter
- Gamma: Estimated noise covariance

References

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

Examples

```
require(fnets)
require(doParallel)
require(lpSolve)
dpca <- dyn.pca(sample.data, q=2)
mg <- make.gg(dpca$acv$Gamma_i,1)
vl <- var.lasso(mg$GG,mg$gg, .1)
vd <- var.dantzig(mg$GG,mg$gg, .1, n.cores = 1)
```

`var.lasso` *Lasso-type Yule-Walker estimation for VAR processes*

Description

Returns parameter estimates for the idiosyncratic VAR and the corresponding Gamma matrix

Usage

```
var.lasso(
  GG,
  gg,
  lambda,
  symmetric = "min",
  niter = 100,
  tol = 0,
  do.plot = FALSE
)
```

Arguments

`GG, gg`	output from `make.gg`
`lambda`	regularisation parameter
`symmetric`	type of symmetry to enforce on `Gamma`, one of 'min', 'max', 'avg', 'none'
`niter`	maximum number of descent steps
`tol`	numerical tolerance for increases in the loss function
`do.plot`	return a plot of the loss function against descent steps

Details

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

Value

A list which contains the following fields:

- `beta`VAR parameters
- `lambda` regularisation parameter
- `Gamma` Estimated noise covariance
- `loss` Objective function value

References

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

Examples

```
require(fnets)
require(doParallel)
require(lpSolve)
dpca <- dyn.pca(sample.data, q=2)
mg <- make.gg(dpca$acv$Gamma_i,1)
vl <- var.lasso(mg$GG,mg$gg, .1)
vd <- var.dantzig(mg$GG,mg$gg, .1, n.cores = 1)
```

Index