Package 'LSVAR'

January 20, 2025

Type Package

Title Estimation of Low Rank Plus Sparse Structured Vector Auto-Regressive (VAR) Model

Version 1.2

Author Peiliang Bai [aut, cre]

Maintainer Peiliang Bai <baipl92@ufl.edu>

Description Implementations of estimation algorithm of low rank plus sparse structured VAR model by using Fast Iterative Shrinkage-Thresholding Algorithm (FISTA). It relates to the algorithm in Sumanta, Li, and Michailidis (2019) <doi:10.1109/TSP.2018.2887401>.

License GPL-2

Encoding UTF-8

Imports igraph, mvtnorm, pracma

Suggests rmarkdown, knitr

VignetteBuilder knitr

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2021-05-26 12:00:02 UTC

Contents

`f.func` *Main loss function for quardratic loss*

Description

Main loss function

Usage

```
f.func(x, A, b)
```

Arguments

`x`	Model parameters
`A`	Design matrix with size of n by p
`b`	Correspond vector or matrix

Value

Value of objective function

`fista.LpS` *A function to solve low rank plus sparse model estimation using FISTA algorithm*

Description

A function to solve low rank plus sparse model estimation

Usage

```
fista.LpS(
  data,
  lambda,
  mu,
  alpha_L = 0.25,
  niter = 100,
  backtracking = TRUE,
  x.true = NULL
)
```

Arguments

data	A numeric dataset with size of n by p
lambda	A positive numeric value, indicating the tuning parameter for sparse component
mu	A positive numeric value, indicating the tuning parameter for low rank component
alpha_L	The constraint coefficient of low rank component, default is 0.25
niter	The maximum number of iterations required for FISTA
backtracking	A boolean argument, indicating that use backtracking in the FISTA
x.true	A p by p matrix, the true model parameter. Only available for simulation.

Value

A S3 object of class LSVAR, including

est_phi estimated model parameter

sparse.comp Estimated sparse component

lr.comp Estimated low-rank component

obj.val Values of objective function

rel.err Relative errors compared with the true model parameters if available

Examples

```
n <- 300
p <- 20
try <- testVAR(n, p, struct = "LS", signal = 0.75, rank = 2,
               singular_vals = c(1, 0.8))
data <- as.matrix(try$series)
lambda <- 0.1; mu <- 1
fit <- fista.LpS(data, lambda = lambda, mu = mu, x.true = try$model_param)
summary(fit, threshold = 0.2)
```

---

gradf.func *Gradient function of quardratic loss*

Description

Gradient function of quardratic loss

Usage

```
gradf.func(x, AtA, Atb)
```

Arguments

x	A vector, or matrix, indicating the model parameter
AtA	A p by p Gram matrix for corresponding design matrix A
Atb	An inner product for design matrix A and corresponding matrix (vector) b

Value

Value of gradients

---

nuclear.pen *Nuclear norm penalty for low-rank component*

Description

Nuclear norm penalty for low-rank component

Usage

```
nuclear.pen(x, lambda)
```

Arguments

x	Model parameter
lambda	Tuning parameter

Value

Value of nuclear norm penalty term

---

obj.func *Objective function*

Description

objective function, main loss function and penalties

Usage

```
obj.func(x.lr, x.sparse, A, b, lambda, mu)
```

Arguments

x.lr	low-rank component
x.sparse	sparse component
A	design matrix
b	correspond vector
lambda	a tuning parameter for sparse component
mu	a tuning parameter for low-rank component

Value

value of objective function

---

plot_network *plot sparse component for use igraph and network layout*

Description

Plot a network to illustrate the estimated sparse component

Usage

```
plot_network(mat, threshold = 0.1)
```

Arguments

mat	a p by p matrix, indicating the sparse component
threshold	the threshold for presenting the edges in the network

Value

A network plot for the sparse component

Examples

```
set.seed(1)
est_mats <- matrix(rnorm(400, 0, 1), 20, 20)
plot_network(est_mats, threshold = 0.1)
```

prox.nuclear.func *Proximal function with nuclear norm penalty updating*

Description

Proximal function with nuclear norm

Usage

```
prox.nuclear.func(w1, y, A, b, L, lambda, AtA, Atb)
```

Arguments

w1	previously updated model parameter
y	updated model parameter
A	design matrix
b	correspond vector, or matrix
L	learning rate
lambda	tuning parameter for low-rank component
AtA	Gram matrix of design matrix A
Atb	inner product of design matrix A and correspond vector b

Value

Value of proximal function with nuclear norm penalty

prox.sparse.func *Proximal function with l1-norm penalty updating*

Description

Proximal function with l1-norm

Usage

```
prox.sparse.func(w1, y, A, b, L, lambda, AtA, Atb)
```

Arguments

w1	previously updated model parameter
y	updated model parameter
A	design matrix
b	correspond vector, or matrix
L	learning rate
lambda	tuning parameter for sparse component
AtA	Gram matrix of design matrix A
Atb	inner product of design matrix A and correspond vector b

Value

Value of proximal function with l1-norm penalty

Q.func *An auxiliary function in FISTA algorithm*

Description

Auxiliary function for FISTA implementation

Usage

```
Q.func(x, y, A, b, L, AtA, Atb)
```

Arguments

x	Model parameter for previous update
y	Model parameter for updating
A	An n by p design matrix
b	A correspond vector, or matrix with size of n by 1 or n by p
L	Learning rate
AtA	Gram matrix for design matrix A
Atb	Inner product for design matrix A and correspond vector b

Value

Value of function Q

shrinkage *Shrinkage function for sparse soft-thresholding*

Description

Shrinkage function for sparse soft-thresholding

Usage

```
shrinkage(y, tau)
```

Arguments

y	A matrix, or a vector for thresholding
tau	A positive number, threshold

Value

A thresholded matrix, or vector

shrinkage.lr *Shrinkage function for low-rank soft-thresholding*

Description

Shrinkage function for low-rank soft-thresholding

Usage

```
shrinkage.lr(y, tau)
```

Arguments

y	A matrix, or a vector for thresholding
tau	A positive number, threshold

Value

A thresholded matrix, or vector

sparse.pen *L1-norm penalty for sparse component*

Description

L1-norm penalty for sparse component

Usage

```
sparse.pen(x, lambda)
```

Arguments

x	Model parameter
lambda	Tuning parameter

Value

Value of l1-norm penalty term

summary.LSVAR *Summary of LSVAR S3 class*

Description

summary function for S3 class for the fitting result

Usage

```
## S3 method for class 'LSVAR'
summary(object, threshold = 0.2, ...)
```

Arguments

object	the S3 class object of LSVAR
threshold	the threshold for sparse component visualization
...	not in use

Value

A series of summary for the S3 result

Examples

```
n <- 300
p <- 20
try <- testVAR(n, p, struct = "LS", signal = 0.75, rank = 2,
               singular_vals = c(1, 0.8))
data <- as.matrix(try$series)
lambda <- 0.1; mu <- 1
fit <- fista.LpS(data, lambda = lambda, mu = mu, x.true = try$model_param)
summary(fit, threshold = 0.2)
```

`testVAR` *Function to generate a VAR process*

Description

A function to generate synthetic time series process based on the given structure

Usage

```
testVAR(
  n,
  p,
  struct = c("sparse", "low rank", "LS")[1],
  sp_density = 0.1,
  signal = NULL,
  rank = NULL,
  singular_vals,
  spectral_radius = 0.9,
  sigma = NULL,
  skip = 50,
  seed = 1
)
```

Arguments

`n`	the length of time series
`p`	the number of multivariate time series
`struct`	a character string indicating the structure of the transition matrix, here are three options: sparse, low rank and LS (low rank plus sparse)
`sp_density`	a numeric value, indicating the sparsity density of sparse components, default is 0.1
`signal`	a numeric value, indicating the magnitude of transition matrix
`rank`	a positive integer, the rank for low rank component
`singular_vals`	a numeric vector, indicating the singular values for the low rank component, the length of singular value must equal to the rank

spectral_radius
: a numeric value, controlling the stability of the process, default is 0.9

sigma
: a numeric matrix, indicating the covariance matrix of noise term

skip
: a numeric value, indicating the number of skipped time points in the beginning of the process

seed
: an integer, indicating the seed for random seed.

Value

A list object, including

series the generated time series

noise the noise term

model_param true transition matrix

Examples

```
n <- 300; p <- 15
signal <- 0.75
rank <- 3
singular_vals <- c(1, 0.75, 0.5)
try <- testVAR(n, p, struct = "LS", signal = signal, rank = rank,
               singular_vals = singular_vals)
data <- as.matrix(try$series)
```

Index