Package ‘gclm’

June 4, 2020

Type Package

Title Graphical Continuous Lyapunov Models

Version 0.0.1

Description Estimation of covariance matrices as solutions of
continuous time Lyapunov equations.
Sparse coefficient matrix and diagonal noise are estimated
with a proximal gradient
method for an l1-penalized loss minimization problem.
Varando G, Hansen NR (2020) <arXiv:2005.10483>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

URL https://github.com/gherardovarando/gclm

BugReports https://github.com/gherardovarando/gclm/issues

Suggests testthat

NeedsCompilation yes

Author Gherardo Varando [aut, cre, cph]
(<https://orcid.org/0000-0002-6708-1103>),
Niels Richard Hansen [aut] (<https://orcid.org/0000-0003-3883-365X>)

Maintainer Gherardo Varando <gherardo.varando@gmail.com>

Repository CRAN

Date/Publication 2020-06-04 08:40:07 UTC

R topics documented:

B0 — *Generate a naive stable matrix*

Description

Generate a naive stable matrix

Usage

```
B0(p)
```

Arguments

`p`	dimension of the matrix

Value

a stable matrix with off-diagonal entries equal to 1 and diagonal entries equal to `-p`

clyap — *Solve continuous-time Lyapunov equations*

Description

`clyap` solve the continuous-time Lyapunov equations

$$BX + XB' + C = 0$$

Using the Bartels-Stewart algorithm with Hessenberg–Schur decomposition. Optionally the Hessenberg-Schur decomposition can be returned.

Usage

```
clyap(B, C, Q = NULL, all = FALSE)
```

Arguments

`B`	Square matrix
`C`	Square matrix
`Q`	Square matrix, the orthogonal matrix used to transform the original equation
`all`	logical

Details

If the matrix `Q` is set then the matrix `B` is assumed to be in upper quasi-triangular form (Hessenberg-Schur canonical form), as required by LAPACK subroutine `DTRSYL` and `Q` is the orthogonal matrix associated with the Hessenberg-Schur form of `B`. Usually the matrix `Q` and the appropriate form of `B` are obtained by a first call to `clyap(B,C,all = TRUE)`

`clyap` uses lapack subroutines:

- `DGEES`
- `DTRSYL`
- `DGEMM`

Value

The solution matrix `X` if `all = FALSE`. If `all = TRUE` a list with components `X`, `B` and `Q`. Where `B` and `Q` are the Hessenberg-Schur form of the original matrix `B` and the orthogonal matrix that performed the transformation.

Examples

```
B <- matrix(data = rnorm(9), nrow = 3)
## make B negative diagonally dominant, thus stable:
diag(B) <- - 3 * max(B)
C <- diag(runif(3))
X <- clyap(B, C)
## check X is a solution:
max(abs(B %*% X + X %*% t(B) + C))
```

---

gclm *l1 penalized loss estimation for GCLM*

Description

Estimates a sparse continuous time Lyapunov parametrization of a covariance matrix using a lasso (L1) penalty.

Usage

```
gclm(
  Sigma,
  B = -0.5 * diag(ncol(Sigma)),
  C = rep(1, ncol(Sigma)),
  C0 = rep(1, ncol(Sigma)),
  loss = "loglik",
  eps = 0.01,
  alpha = 0.5,
  maxIter = 100,
  lambda = 0,
```

```
  lambdac = 0,
  job = 0
)

gclm.path(
  Sigma,
  lambdas = NULL,
  B = -0.5 * diag(ncol(Sigma)),
  C = rep(1, ncol(Sigma)),
  ...
)
```

Arguments

Sigma	covariance matrix
B	initial B matrix
C	diagonal of initial C matrix
C0	diagonal of penalization matrix
loss	one of "loglik" (default) or "frobenius"
eps	convergence threshold
alpha	parameter line search
maxIter	maximum number of iterations
lambda	penalization coefficient for B
lambdac	penalization coefficient for C
job	integer 0,1,10 or 11
lambdas	sequence of lambda
...	additional arguments passed to `gclm`

Details

`gclm` performs proximal gradient descent for the optimization problem

$$argminL(\Sigma(B,C)) + \lambda\rho(B) + \lambda_C||C - C0||_F^2$$

subject to B stable and C diagonal, where $\rho(B)$ is the l1 norm of the off-diagonal element of B.

`gclm.path` simply calls iteratively `gclm` with different `lambda` values. Warm start is used, that is in the i-th call to `gclm` the `B` and `C` matrices are initialized as the one obtained in the (i-1)th call.

Value

for `gclm`: a list with the result of the optimization

for `gclm.path`: a list of the same length of `lambdas` with the results of the optimization for the different `lambda` values

Examples

```
x <- matrix(rnorm(50*20),ncol=20)
S <- cov(x)

## l1 penalized log-likelihood
res <- gclm(S, eps = 0, lambda = 0.1, lambdac = 0.01)

## l1 penalized log-likelihood with fixed C
res <- gclm(S, eps = 0, lambda = 0.1, lambdac = -1)

## l1 penalized frobenius loss
res <- gclm(S, eps = 0, lambda = 0.1, loss = "frobenius")
```

`gclm.lowertri` *Recover lower triangular GCLM*

Description

Recover the only lower triangular stable matrix B such that `Sigma` (Σ) is the solution of the associated continuous Lyapunov equation:

$$B\Sigma + \Sigma B' + C = 0$$

Usage

```
gclm.lowertri(Sigma, P = solve(Sigma), C = diag(nrow = nrow(Sigma)))
```

Arguments

`Sigma`	covariance matrix
`P`	the inverse of the covariance matrix
`C`	symmetric positive definite matrix

Value

A stable lower triangular matrix

Index