Package 'GLASSOO'

July 31, 2018

Type Package

Title Lasso Penalized Precision Matrix Estimation

Version 1.0

Date 2018-05-17

Description Estimates a lasso penalized precision matrix via the blockwise coordinate descent algorithm (BCD) algorithm. This package is an alternative to the 'glasso' package.
See Friedman et al (2008) <doi:10.1093/biostatistics/kxm045> for details regarding the estimation method.

URL https://github.com/MGallow/GLASSOO

BugReports https://github.com/MGallow/GLASSOO/issues

License GPL (>= 2)

ByteCompile TRUE

NeedsCompilation yes

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Imports stats,
parallel,
foreach,
ggplot2,
dplyr

Depends Rcpp (>= 0.12.10),
RcppProgress (>= 0.1),
doParallel

LinkingTo Rcpp,
RcppArmadillo,
RcppProgress

Suggests testthat,
knitr,
rmarkdown,
pkgdown

SystemRequirements GNU make

VignetteBuilder knitr

R topics documented:

GLASSO *Penalized precision matrix estimation*

Description

Penalized precision matrix estimation using the graphical lasso (glasso) algorithm. Consider the case where $X_1, ..., X_n$ are iid $N_p(\mu, \Sigma)$ and we are tasked with estimating the precision matrix, denoted $\Omega \equiv \Sigma^{-1}$. This function solves the following optimization problem:

Objective: $\hat{\Omega}_\lambda = \arg\min_{\Omega \in S_+^p} \{Tr(S\Omega) - \log\det(\Omega) + \lambda \|\Omega\|_1\}$

where $\lambda > 0$ and we define $\|A\|_1 = \sum_{i,j} |A_{ij}|$.

Usage

```
GLASSO(X = NULL, S = NULL, nlam = 10, lam.min.ratio = 0.01,
  lam = NULL, diagonal = FALSE, path = FALSE, crit.out = c("avg",
  "max"), crit.in = c("loss", "avg", "max"), tol.out = 1e-04,
  tol.in = 1e-04, maxit.out = 10000, maxit.in = 10000,
  adjmaxit.out = NULL, K = 5, crit.cv = c("loglik", "AIC", "BIC"),
  start = c("warm", "cold"), cores = 1, trace = c("progress", "print",
  "none"))
```

Arguments

X	option to provide a nxp data matrix. Each row corresponds to a single observation and each column contains n observations of a single feature/variable.
S	option to provide a pxp sample covariance matrix (denominator n). If argument is NULL and X is provided instead then S will be computed automatically.
nlam	number of lam tuning parameters for penalty term generated from lam.min.ratio and lam.max (automatically generated). Defaults to 10.
lam.min.ratio	smallest lam value provided as a fraction of lam.max. The function will automatically generate nlam tuning parameters from lam.min.ratio*lam.max to lam.max in log10 scale. lam.max is calculated to be the smallest lam such that all off-diagonal entries in Omega are equal to zero (alpha = 1). Defaults to 1e-2.
lam	option to provide positive tuning parameters for penalty term. This will cause nlam and lam.min.ratio to be disregarded. If a vector of parameters is provided, they should be in increasing order. Defaults to NULL.
diagonal	option to penalize the diagonal elements of the estimated precision matrix (Ω). Defaults to FALSE.
path	option to return the regularization path. This option should be used with extreme care if the dimension is large. If set to TRUE, cores must be set to 1 and errors and optimal tuning parameters will based on the full sample. Defaults to FALSE.

crit.out	criterion for convergence in outer (blockwise) loop. Criterion avg will loop until the average absolute parameter change is less than tol.out times tolerance multiple. Criterion max will loop until the maximum change in the estimated Sigma after an iteration over the parameter set is less than tol.out. Defaults to avg.
crit.in	criterion for convergence in inner (lasso) loop. Criterion for convergence. Criterion loss will loop until the relative change in the objective for each response after an iteration is less than tol.in. Criterion avg will loop until the average absolute change for each response is less than tol.in times tolerance multiple. Similary, criterion max will loop until the maximum absolute change is less than tol.in times tolerance multiple. Defaults to loss.
tol.out	convergence tolerance for outer (blockwise) loop. Defaults to 1e-4.
tol.in	convergence tolerance for inner (lasso) loop. Defaults to 1e-4.
maxit.out	maximum number of iterations for outer (blockwise) loop. Defaults to 1e4.
maxit.in	maximum number of iterations for inner (lasso) loop. Defaults to 1e4.
adjmaxit.out	adjusted maximum number of iterations. During cross validation this option allows the user to adjust the maximum number of iterations after the first lam tuning parameter has converged. This option is intended to be paired with warm starts and allows for 'one-step' estimators. Defaults to NULL.
K	specify the number of folds for cross validation.
crit.cv	cross validation criterion (loglik, AIC, or BIC). Defaults to loglik.
start	specify warm or cold start for cross validation. Default is warm.
cores	option to run CV in parallel. Defaults to cores = 1.
trace	option to display progress of CV. Choose one of progress to print a progress bar, print to print completed tuning parameters, or none.

Details

For details on the implementation of 'GLASSOO', see the vignette https://mgallow.github.io/GLASSOO/.

Value

returns class object GLASSOO which includes:

Call	function call.
Iterations	number of iterations/
Tuning	optimal tuning parameters.
Lambdas	grid of lambda values for CV.
maxit.out	maximum number of iterations for outer (blockwise) loop.
maxit.in	maximum number of iterations for inner (lasso) loop.
Omega	estimated penalized precision matrix.
Sigma	estimated covariance matrix from the penalized precision matrix (inverse of Omega).
Path	array containing the solution path. Solutions will be ordered by ascending lambda values.
MIN.error	minimum average cross validation error (cv.crit) for optimal parameters.
AVG.error	average cross validation error (cv.crit) across all folds.
CV.error	cross validation errors (cv.crit).

Author(s)

Matt Galloway <gall0441@umn.edu>

References

- Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. 'Sparse inverse covariance estimation with the graphical lasso.' *Biostatistics* 9.3 (2008): 432-441.
- Banerjee, Onureen, Ghauoui, Laurent El, and d'Aspremont, Alexandre. 2008. 'Model Selection through Sparse Maximum Likelihood Estimation for Multivariate Gaussian or Binary Data.' *Journal of Machine Learning Research* 9: 485-516.
- Tibshirani, Robert. 1996. 'Regression Shrinkage and Selection via the Lasso.' *Journal of the Royal Statistical Society. Series B (Methodological)*. JSTOR: 267-288.
- Meinshausen, Nicolai and Buhlmann, Peter. 2006. 'High-Dimensional Graphs and Variable Selection with the Lasso.' *The Annals of Statistics*. JSTOR: 1436-1462.
- Witten, Daniela M, Friedman, Jerome H, and Simon, Noah. 2011. 'New Insights and Faster computations for the Graphical Lasso.' *Journal of Computation and Graphical Statistics*. Taylor and Francis: 892-900.
- Tibshirani, Robert, Bien, Jacob, Friedman, Jerome, Hastie, Trevor, Simon, Noah, Jonathan, Taylor, and Tibshirani, Ryan J. 'Strong Rules for Discarding Predictors in Lasso-Type Problems.' *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*. Wiley Online Library 74 (2): 245-266.
- Ghaoui, Laurent El, Viallon, Vivian, and Rabbani, Tarek. 2010. 'Safe Feature Elimination for the Lasso and Sparse Supervised Learning Problems.' *arXiv preprint arXiv: 1009.4219*.
- Osborne, Michael R, Presnell, Brett, and Turlach, Berwin A. 'On the Lasso and its Dual.' *Journal of Computational and Graphical Statistics*. Taylor and Francis 9 (2): 319-337.
- Rothman, Adam. 2017. 'STAT 8931 notes on an algorithm to compute the Lasso-penalized Gausssian likelihood precision matrix estimator.'

See Also

plot.GLASSO

Examples

```
# generate data from a sparse matrix
# first compute covariance matrix
S = matrix(0.7, nrow = 5, ncol = 5)
for (i in 1:5){
 for (j in 1:5){
   S[i, j] = S[i, j]^abs(i - j)
 }
 }

# generate 100 x 5 matrix with rows drawn from iid N_p
set.seed(123)
Z = matrix(rnorm(100*5), nrow = 100, ncol = 5)
out = eigen(S, symmetric = TRUE)
S.sqrt = out$vectors %*% diag(out$values^0.5)
S.sqrt = S.sqrt %*% t(out$vectors)
X = Z %*% S.sqrt

# lasso penalty CV
GLASSO(X)
```

plot.GLASSO *Plot GLASSO object*

Description

Produces a plot for the cross validation errors, if available.

Usage

```
## S3 method for class 'GLASSO'
plot(x, type = c("line", "heatmap"), footnote = TRUE, ...)
```

Arguments

x	class object GLASSO
type	produce either 'heatmap' or 'line' graph
footnote	option to print footnote of optimal values. Defaults to TRUE.
...	additional arguments.

Examples

```
# generate data from a sparse matrix
# first compute covariance matrix
S = matrix(0.7, nrow = 5, ncol = 5)
for (i in 1:5){
 for (j in 1:5){
   S[i, j] = S[i, j]^abs(i - j)
 }
 }

# generate 100 x 5 matrix with rows drawn from iid N_p(0, S)
set.seed(123)
Z = matrix(rnorm(100*5), nrow = 100, ncol = 5)
out = eigen(S, symmetric = TRUE)
S.sqrt = out$vectors %*% diag(out$values^0.5)
S.sqrt = S.sqrt %*% t(out$vectors)
X = Z %*% S.sqrt

# produce line graph for GLASSO
plot(GLASSO(X))

# produce CV heat map for GLASSO
plot(GLASSO(X), type = 'heatmap')
```

Index