Package 'SCPME'

July 30, 2018

Type Package

Title Shrinking Characteristics of Precision Matrix Estimators

Version 1.0

Date 2018-07-30

Description An implementation of the methods described in ``Shrinking Characteristics of Precision Matrix Estimators" by Aaron J. Molstad, Ph.D and Adam J. Rothman, Ph.D. The manuscript can be found here: https://doi.org/10.1093/biomet/asy023 .

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

BugReports https://github.com/MGallow/SCPME/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,
microbenchmark,
glasso,
pkgdown

SystemRequirements GNU make

VignetteBuilder knitr

R topics documented:

`data_gen` *Normal regression data generator*

Description

True beta values are generated from p*r independent draws from N(0, 1/p) distribution. X are n independent draws from p multivariate normal N(0, SigmaX). Y is then generated using X and true beta values with an iid error term that follows r multivariate normal distribution N(0, Sigma).

Usage

```
data_gen(n, p, r = 1, sparsity = 0.5, Sigma = c("tridiag", "dense",
  "denseQR", "compound"), s = NULL, SigmaX = c("tridiag", "dense",
  "denseQR", "compound"), sx = NULL, ...)
```

Arguments

`n`	desired sample size
`p`	desired dimension
`r`	number of responses
`sparsity`	desired sparsity for beta
`Sigma`	covariance matrix structure used to generate Y | X
`s`	option to specify diagonal elements in Sigma
`SigmaX`	covariance matrix structure used to generate data X
`sx`	option to specify diagonal elements in SigmaX
`...`	additional arguments to pass to data generating functions

Value

Y, X, betas, Sigma, SigmaX

Author(s)

Matt Galloway <gall0441@umn.edu>

Examples

```
# generate 100 observations with predictor dimension equal to 10 and response dimension equal to 5
data = data_gen(n = 100, p = 10, r = 5)
```

plot.shrink *Plot shrink object*

Description

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

Usage

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

Arguments

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

Examples

```
# generate some data
data = data_gen(n = 100, p = 5, r = 1)

# lasso penalized beta (print estimated omega)
(shrink = shrink(X = data$X, Y = data$Y, B = cov(data$X, data$Y), lam.max = max(abs(t(data$X) %*% data$Y))))

# print estimated beta
shrink$Z

# create heatmap of CV errors
plot(shrink, type = 'heatmap')

# create line graph of CV errors
plot(shrink)
```

shrink *Shrinking characteristics of precision matrix estimators*

Description

Shrinking characteristics of precision matrix estimators. Penalized precision matrix estimation using the ADMM 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: $\hat{\Omega}_\lambda = \arg\min_{\Omega \in S_+^p} \{Tr(S\Omega) - \log\det(\Omega) + \lambda \|A\Omega B - C\|_1\}$

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

Usage

```
shrink(X = NULL, Y = NULL, S = NULL, A = diag(ncol(S)),
  B = diag(ncol(S)), C = matrix(0, ncol = ncol(B), nrow = ncol(A)),
  nlam = 10, lam.max = NULL, lam.min.ratio = 0.001, lam = NULL,
  alpha = 1, path = FALSE, rho = 2, mu = 10, tau.rho = 2,
  iter.rho = 10, crit = c("ADMM", "loglik"), tol.abs = 1e-04,
  tol.rel = 1e-04, maxit = 10000, adjmaxit = NULL, K = 5,
  crit.cv = c("MSE", "loglik", "penloglik", "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.
Y	option to provide nxr response matrix. Each row corresponds to a single response and each column contains n response of a single feature/response.
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.
A	option to provide user-specified matrix for penalty term. This matrix must have p columns. Defaults to identity matrix.
B	option to provide user-specified matrix for penalty term. This matrix must have p rows. Defaults to identity matrix.
C	option to provide user-specified matrix for penalty term. This matrix must have nrow(A) rows and ncol(B) columns. Defaults to zero matrix.
nlam	number of lam tuning parameters for penalty term generated from lam.min.ratio and lam.max (automatically generated). Defaults to 10.
lam.max	option to specify the maximum lam tuning parameter. Defaults to NULL.
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. If lam.max = NULL, lam.max is calculated to be the smallest lam such that all off-diagonal entries in Omega are equal to zero. Defaults to 1e-3.
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.
alpha	elastic net mixing parameter contained in [0, 1]. 0 = ridge, 1 = lasso. Alpha must be a single value (cross validation across alpha not supported).
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.
rho	initial step size for ADMM algorithm.
mu	factor for primal and residual norms in the ADMM algorithm. This will be used to adjust the step size rho after each iteration.
tau.rho	factor in which to increase/decrease step size rho
iter.rho	step size rho will be updated every iter.rho steps
crit	criterion for convergence (ADMM or loglik). If crit = loglik then iterations will stop when the relative change in log-likelihood is less than tol.abs. Default is ADMM and follows the procedure outlined in Boyd, et al.

tol.abs	absolute convergence tolerance. Defaults to 1e-4.
tol.rel	relative convergence tolerance. Defaults to 1e-4.
maxit	maximum number of iterations. Defaults to 1e4.
adjmaxit	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 (for each alpha). 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 (MSE, loglik, penloglik, AIC, or BIC). Defaults to MSE.
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 'shrink', see the vignette https://mgallow.github.io/SCPME/.

Value

returns class object ADMMsigma which includes:

Call	function call.
Iterations	number of iterations.
Tuning	optimal tuning parameter.
Lambdas	grid of lambda values for CV.
maxit	maximum number of iterations.
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 in ascending alpha values for each lambda.
Z	final sparse update of estimated penalized precision matrix.
Y	final dual update.
rho	final step size.
Loglik	penalized log-likelihood for Omega
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

- Boyd, Stephen, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, and others. 2011. 'Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers.' *Foundations and Trends in Machine Learning* 3 (1). Now Publishers, Inc.: 1-122. https://web.stanford.edu/~boyd/papers/pdf/admm_distr_stats.pdf
- Hu, Yue, Chi, Eric C, amd Allen, Genevera I. 2016. 'ADMM Algorithmic Regularization Paths for Sparse Statistical Machine Learning.' *Splitting Methods in Communication, Imaging, Science, and Engineering*. Springer: 433-459.
- Molstad, Aaron J., and Adam J. Rothman. (2017). 'Shrinking Characteristics of Precision Matrix Estimators. *Biometrika.*. https://doi.org/10.1093/biomet/asy023
- Rothman, Adam. 2017. 'STAT 8931 notes on an algorithm to compute the Lasso-penalized Gaussian likelihood precision matrix estimator.'

See Also

plot.shrink

Examples

```
# generate some data
data = data_gen(n = 100, p = 5, r = 1)

# lasso penalized omega (glasso)
shrink(X = data$X, Y = data$Y)

# lasso penalized beta (print estimated omega)
(shrink = shrink(X = data$X, Y = data$Y, B = cov(data$X, data$Y), lam.max = max(abs(t(data$X) %*% data$Y))))

# print estimated beta
shrink$Z
```

Index