Package: rDecode (via r-universe)

September 18, 2024

Type Package

Title Descent-Based Calibrated Optimal Direct Estimation

Version 0.1.0

Author Chi Seng Pun, Matthew Zakharia Hadimaja

Maintainer Chi Seng Pun <cspun@ntu.edu.sg>

Description Algorithms for solving a self-calibrated l1-regularized quadratic programming problem without parameter tuning. The algorithm, called DECODE, can handle high-dimensional data without cross-validation. It is found useful in high dimensional portfolio selection (see Pun (2018) <https://ssrn.com/abstract=3179569>) and large precision matrix estimation and sparse linear discriminant analysis (see Pun and Hadimaja (2019) <https://ssrn.com/abstract=3422590>).

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.0.0

Depends R (>= 2.10)

Imports stats

NeedsCompilation no

Date/Publication 2019-12-18 14:20:05 UTC

Repository https://cspun.r-universe.dev

RemoteUrl https://github.com/cran/rDecode

RemoteRef HEAD

RemoteSha fd4ff30abc6432f0962b29455a587a8b2fa5aa4d

Contents

`decode` *Descent-based Calibrated Optimal Direct Estimation*

Description

Implement `DECODE` for `sigma` and `beta` to estimate $\Sigma^{-1}\beta$ where `sigma` is an estimator of Σ and `beta` is an estimator of β.

Usage

```
decode(sigma, beta, lambda0, decode.tol = 1e-06, decode.maxit = 100,
  trace = FALSE, solver = c("apg", "homotopy"), solver.tol = 1e-08,
  solver.maxit = 10000, return.sigma = FALSE, return.beta = FALSE,
  return.param = FALSE)
```

Arguments

`sigma`	$p \times p$ positive semidefinite symmetric matrix. `sigma` will be perturbed if needed.
`beta`	p-length vector.
`lambda0`	number between 0 and 1.
`decode.tol`	error tolerance for `DECODE`.
`decode.maxit`	maximum iterations for `DECODE`
`trace`	logical. If `TRUE`, will return η, θ, and λ found during each iteration of `DECODE`
`solver`	solver for ℓ_1-RQP problem inside `DECODE`.
`solver.tol`	tolerance for solver.
`solver.maxit`	maximum iterations for solver (only for APG).
`return.sigma`	logical. If `TRUE` the `sigma` entered is returned.
`return.beta`	logical. If `TRUE` the `beta` entered is returned.
`return.param`	logical. If `TRUE` the parameters used are returned.

Value

An object of class decode containing:

`eta`	`DECODE` of $\Sigma^{-1}\beta$.
`theta`	final θ of the `DECODE`.
`lambda`	final λ of the `DECODE`.
`sigma.mult`	multiplier applied on `sigma` to ensure convergence.

`total.iter`	number of iterations until convergence.
`call`	the matched call.
`method`	the solver used, if requested.
`lambda0`	the `lambda0` entered, if requested.
`decode.tol`	the `decode.tol` used, if requested.
`decode.maxit`	the `decode.maxit` used, if requested.
`trace`	the `trace` used, if requested.
`solver.tol`	the `solver.tol` used, if requested.
`solver.maxit`	the `solver.maxit` used, if requested.
`eta.trace`	matrix of η used in each iteration, if requested.
`theta.trace`	vector of θ used in each iteration, if requested.
`lambda.trace`	vector of λ used in each iteration, if requested.

References

Pun, C. S. (2018). A Sparse Learning Approach to Relative-Volatility-Managed Portfolio Selection. Hadimaja, M. Z., & Pun, C. S. (2018). A Self-Calibrated Regularized Direct Estimation for Graphical Selection and Discriminant Analysis.

Examples

```
# estimate A^(-1) b with a certain lambda0
X <- matrix(rnorm(100), 10, 10)
A <- t(X) %*% X
b <- rnorm(10)
object <- decode(A, b, lambda0 = 0.8)

object
summary(object)

coef(object)
```

`decodeLDA` *Implement* `DECODE` *for simple LDA*

Description

Implement `DECODE` for simple LDA. The LDA assumes both classes have equal prior probabilities. This implementation is used in Hadimaja and Pun (2018).

Usage

```
decodeLDA(X, y, lambda0 = NULL, ...)
```

Arguments

`X` $n \times p$ data matrix.

`y` binary n-length vector containing the class of each observation.

`lambda0` number between 0 and 1. If `NULL`, will use $\sqrt{2\log p/n}$.

`...` additional arguments to be passed to general decode function.

Value

An object of class `decodeLDA` containing:

`eta` `DECODE` of $\Omega\delta$

`X` training data used

`y` training label used

and various outputs from `decode` function.

References

Hadimaja, M. Z., & Pun, C. S. (2018). A Self-Calibrated Regularized Direct Estimation for Graphical Selection and Discriminant Analysis.

Examples

```
# for efficiency, we will only use 500 variables

# load the training data (Lung cancer data, cleaned)
data(lung.train) # 145 x 1578
X.train <- lung.train[,1:500]
y.train <- lung.train[,1578]

# build the DECODE
object <- decodeLDA(X.train, y.train)

object
summary(object)
coef(object)

# test on test data
data(lung.test)
X.test <- lung.test[,1:500]
y.test <- lung.test[,1578]
y.pred <- predict(object, X.test)
table(y.pred, y.test)
```

decodePM *Implement* DECODE *for simple precision matrix estimation*

Description

Implement DECODE to estimate a precision matrix of X. This implementation is used in Hadimaja and Pun (2018).

Usage

```
decodePM(X, lambda0 = NULL, ...)
```

Arguments

X	$n \times p$ data matrix.
lambda0	number between 0 and 1. If NULL, will use $\sqrt{2 \log p/n}$.
...	additional arguments to be passed to general decode function.

Value

An object of class decodePM containing:

Omega	DECODE of Ω.
lambda0	the lambda0 used.
X	data used.
theta	final θ for each column.
lambda	final λ for each column.
total.iter	number of iterations until convergence for each column.

References

Hadimaja, M. Z., & Pun, C. S. (2018). A Self-Calibrated Regularized Direct Estimation for Graphical Selection and Discriminant Analysis.

Examples

```
# estimate the precision matrix of iris data
object <- decodePM(iris[,1:4], lambda0 = 0.01)

object
summary(object)

object$Omega
```

`lung.test` *Lung cancer test data set from Gordon et al. (2002)*

Description

Preprocessed lung cancer test data of 1577 genes from 36 patients with lung cancer. There are 30 patients with adenocarcinoma (AD) and 6 patients with malignant pleural mesothelioma (MPM). The original data was used in Gordon et al. (2002), with this preprocessed version used in Pun and Hadimaja (2018).

Usage

```
data(lung.test)
```

Format

A data frame with 36 observations on 1577 variables.

Source

http://dx.doi.org/10.17632/ynp2tst2hh.4#file-673c9416-39ed-446d-9be9-37ac74353029

References

Gordon, G. J., Jensen, R. V., Hsiao, L. L., Gullans, S. R., Blumenstock, J. E., Ramaswamy, S., ... & Bueno, R. (2002). Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. Cancer research, 62(17), 4963-4967.

`lung.train` *Lung cancer training data set from Gordon et al. (2002)*

Description

Preprocessed lung cancer training data of 1577 genes from 145 patients with lung cancer. There are 120 patients with adenocarcinoma (AD) and 25 patients with malignant pleural mesothelioma (MPM). The original data was used in Gordon et al. (2002), with this preprocessed version used in Pun and Hadimaja (2018).

Usage

```
data(lung.train)
```

Format

A data frame with 145 observations on 1577 variables.

Source

http://dx.doi.org/10.17632/ynp2tst2hh.4#file-673c9416-39ed-446d-9be9-37ac74353029

References

Gordon, G. J., Jensen, R. V., Hsiao, L. L., Gullans, S. R., Blumenstock, J. E., Ramaswamy, S., ... & Bueno, R. (2002). Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. Cancer research, 62(17), 4963-4967.

Index