# Package ‘lol’

April 10, 2015

**Type** Package

**Title** Lots Of Lasso

**Version** 1.14.0

**Date** 2011-04-02

**Author** Yinyin Yuan

**Maintainer** Yinyin Yuan <yy341@cam.ac.uk>

**Description** Various optimization methods for Lasso inference with matrix warpper

**Depends** penalized, Matrix

**Imports** Matrix, penalized, graphics, grDevices, stats

**License** GPL-2

**LazyLoad** yes

**biocViews** StatisticalMethod

## R topics documented:

`lol-package` *Lots of Lasso*

### Description

Various optimization methods for Lasso inference with matrix wrapper.

### Details

| | |
|---|---|
| Package: | lol |
| Type: | Package |
| Version: | 0.99.0 |
| Date: | 2011-04-02 |
| License: | GPL-2 |
| LazyLoad: | yes |

### Author(s)

Yinyin Yuan Maintainer: Yinyin Yuan <yy341@cam.ac.uk>

### References

Goeman, J. J. (2009), L1 penalized estimation in the cox proportional hazards model. Biometrical Journal. N. Meinshausen and P. Buehlmann (2010), Stability Selection (with discussion), Journal of the Royal Statistical Society, Series B, 72, 417-473.

### See Also

lasso, matrixLasso

### Examples

```
data(chin07)
data <- list(y=t(chin07$ge), x=t(chin07$cn))
res <- matrixLasso(data, method=cv, nFold=5)
res
```

chin07 *Breast cancer data set of genome-wide copy number merged data and expression of some important genes*

### Description

A subset of breast cancer data as used in Yuan et al. (to be submitted).

### Usage

```
data(chin07)
```

### Format

A list object of two named data matrices, cn: DNA copy number, ge: RNA expression. The matrices columns are samples and rows are probes/variables.

### Details

Genome-wide copy number data was merged using CGHregions resulting in 339 regions across 106 samples. Expression data are 7 probes mapped to important breast cancer genes such as CCNE2, MYC, etc, also of 106 samples.

### References

Chin SF, Teschendorff AE, Marioni JC, Wang Y, Barbosa-Morais NL, et al. (2007) High-resolution arraycgh and expression profiling identifies a novel genomic subtype of er negative breast cancer. Genome Biology 8: R215+. Yuan et al. (2011) Discovery and functional annotation of cis- and trans-acting DNA copy number hotspots in breast cancer, to be submitted.

### Examples

```
data(chin07)
gain <- rowSums(chin07$cn >= .2)
loss <- -rowSums(chin07$cn <= -.2)
plotGW(data=cbind(gain, loss), pos=attr(chin07$cn, chrome), legend=c(gain, loss))
```

getLambdaNcoef *get the lambda value that yield certain number of non-zero coefficients*

### Description

get the lambda value that yield certain number of non-zero coefficients

### Usage

```
getLambdaNcoef(y, x, lambda1, nCoef, track=FALSE, model=linear, standardize=FALSE)
```

## Arguments

| | |
|---|---|
| `y` | A vector of expressions |
| `x` | a matrix of CN variables |
| `lambda1` | minimum lambda to use |
| `nCoef` | the number of coefficients to get |
| `track` | logical value for tracking the progress |
| `model` | which model to use, default to 'linear' |
| `standardize` | standardize the data or not |

## Value

| | |
|---|---|
| `lambda` | The lambda value that gives approximate same number of non-zero coefficients as required |

## Author(s)

Yinyin Yuan

## See Also

lasso

## Examples

```
data(chin07)
data <- list(y=chin07$ge[1,], x=t(chin07$cn))
getLambdaNcoef(data$y, data$x, lambda1=.1, nCoef=10, track=TRUE)
```

---

`lasso` *lasso*

---

## Description

Lasso penalized linear regression with different optimizers

## Usage

```
lasso(y, ...)
```

## Arguments

| | |
|---|---|
| `y` | A list object of one of the four classes: 'cv', 'stability', 'multiSplit', and 'simultaneous'. If x is NULL then y should a list of two components y and x, y is a vector of expression and x is a matrix containing copy number variables |
| `...` | other parameters |

### Details

The function contains various optimization methods for Lasso inference, such as cross-validation, randomised lasso, simultaneous lasso etc. It is specifically designed for multicollinear predictor variables.

### Value

Varied depending on the optimizer used. Generally it contains

| | |
|---|---|
| `beta` | coefficients |
| `residuals` | residuals of regression model |
| `fit` | the corresponding fit of regression |

### Author(s)

Yinyin Yuan

### References

Goeman, J. J. (2009), L1 penalized estimation in the cox proportional hazards model, Biometrical Journal. N. Meinshausen and P. Buehlmann (2010), Stability Selection (with discussion), Journal of the Royal Statistical Society, Series B, 72, 417-473. Nicolai Meinshausen, Lukas Meier and Peter Buehlmann (2009), P-values for high-dimensional regression. Journal of the American Statistical Association, 104, 1671-1681.

### See Also

matrixLasso

### Examples

```
data(chin07)
data <- list(y=chin07$ge[1,], x=t(chin07$cn))
class(data) <- cv
res <- lasso(data)
```

---

## `lasso.cv` *Cross validation optimizer for lasso*

---

### Description

Cross validation lasso. This function optimizes the lasso solution for correlated regulators by an algorithm. this algorithm chooses the minimum lambda since the penalized package by default use 0 for the minimum, which sometimes take a long time to compute

### Usage

```
lasso.cv(y, x=NULL, lambda1=NULL, model=linear, steps=15, minsteps=5, log=TRUE, track=FALSE, standardi
```

### Arguments

| | |
|---|---|
| `y` | A vector of gene expression of a probe, or a list object if x is NULL. In the latter case y should a list of two components y and x, y is a vector of expression and x is a matrix containing copy number variables |
| `x` | Either a matrix containing CN variables or NULL |
| `lambda1` | minimum lambda to use |
| `model` | which model to use, one of "cox", "logistic", "linear", or "poisson". Default to 'linear' |
| `steps` | parameter to be passed to penalized |
| `minsteps` | parameter to be passed to penalized |
| `log` | parameter to be passed to penalized |
| `track` | parameter to be passed to penalized |
| `standardize` | parameter to be passed to penalized |
| `unpenalized` | parameter to be passed to penalized |
| `nFold` | parameter to be passed to penalized |
| `nMaxiter` | parameter to be passed to penalized |
| `...` | other parameter to be passed to penalized |

### Value

A list object of class 'lol', consisting of:

| | |
|---|---|
| `fit` | The final sparse regression fit |
| `beta` | the coefficients, non-zero ones are significant |
| `lambda` | the penalty parameter lambda used |
| `residuals` | regression residuals |
| `conv` | logical value indicating whether the optimization has converged |

### Author(s)

Yinyin Yuan

### References

Goeman, J. J. (2009), L1 penalized estimation in the cox proportional hazards model, Biometrical Journal.

### See Also

lasso

### Examples

```
data(chin07)
data <- list(y=chin07$ge[1,], x=t(chin07$cn), nFold=5)
res <- lasso.cv(data)
res
```

`lasso.multiSplit` *Multi-split lasso*

## Description

Multi-split lasso as described in Meinshausen 2009

## Usage

```
lasso.multiSplit(y, x=NULL, lambda1=NULL, nSubsampling=200, model=linear, alpha=0.05, gamma.min=0.05,
```

## Arguments

| | |
|---|---|
| `y` | A vector of gene expression of a probe, or a list object if x is NULL. In the latter case y should a list of two components y and x, y is a vector of expression and x is a matrix containing copy number variables |
| `x` | Either a matrix containing CN variables or NULL |
| `nSubsampling` | number of splits, default to 200 |
| `model` | which model to use, one of "cox", "logistic", "linear", or "poisson". Default to 'linear' |
| `alpha` | specify significant level to determine the non-zero coefficients in the range of 0 and 1, default to 0.05 |
| `gamma.min` | the lower bound of gamma |
| `gamma.max` | the higher bound of gamma |
| `lambda1` | minimum lambda to be used, if known |
| `track` | track progress |
| `...` | other parameters to be passed to lass.cv |

## Details

This function performs the multi-split lasso as proposed by Meinshausen et al. 2009. The samples are first randomly split into two disjoint sets, one of which is used to find non-zero coefficients with a regular lasso regression, then these non-zero coefficients are fitted to another sample set with OLS. The resulting p-values after multiple runs can then be aggregated using quantiles.

## Value

A list object of class 'lol', consisting of:

| | |
|---|---|
| `beta` | coefficients |
| `mat` | the Q_gamma matrix as described in the paper |
| `residuals` | residuals, here is only the input y |
| `pmat` | the adjusted p matrix as described in the paper |

### Author(s)

Yinyin Yuan

### References

Nicolai Meinshausen, Lukas Meier and Peter Buehlmann (2009), P-values for high-dimensional regression. Journal of the American Statistical Association, 104, 1671-1681.

### See Also

lasso

### Examples

```
data(chin07)
data <- list(y=chin07$ge[1,], x=t(chin07$cn))
res <- lasso.multiSplit(data, nSubsampling=50)
res
```

---

## lasso.simultaneous *Simultaneous lasso*

---

### Description

The function performs lasso with multiple random sample splits, selecting coefficients that are simultaneously non-zero in both subsets of samples.

### Usage

```
lasso.simultaneous(y, x=NULL, model=linear, nSubsampling=200, alpha=.5, lambda1=NULL, track=FALSE, ...
```

### Arguments

| | |
|---|---|
| `y` | A vector of gene expression of a probe, or a list object if x is NULL. In the latter case y should a list of two components y and x, y is a vector of expression and x is a matrix containing copy number variables |
| `x` | Either a matrix containing CN variables or NULL |
| `model` | which model to use, one of "cox", "logistic", "linear", or "poisson". Default to 'linear' |
| `nSubsampling` | The number of random permutations, both on sample spliting and on variable scaling, default to 200. |
| `alpha` | weakness parameter: control the shrinkage of regulators. The lower alpha is, the bigger the vanishing effect on small coefficients. |
| `lambda1` | minimum lambda, default to NULL |
| `track` | logical value, whether to track the progress |
| `...` | Other parameters to be passed to the penalized function |

### Details

In each run the function splits samples randomly to two equal sets, run lasso on both sets, then select those coefficients that are simultaneously non-zero across two sets. Finally the results across many runs are summarized as the frequency of selected predictors - the higher the frequency the more confidence that the corresponding predictors are significant.

### Value

A list object of class 'lol', consisting of:

| | |
|---|---|
| `beta` | Coefficient vector |
| `n` | Number of actual subsampling, should be equal or smaller than nSubsampling in case of failing. |
| `mat` | result matrix of the subsampling |

### Author(s)

Yinyin Yuan

### References

N. Meinshausen and P. Buehlmann (2010), Stability Selection (with discussion), Journal of the Royal Statistical Society, Series B, 72, 417-473.

### See Also

lasso

### Examples

```
data(chin07)
data <- list(y=chin07$ge[1,], x=t(chin07$cn))
res <- lasso.simultaneous(data, nSubsampling=50)
res
```

---

## `lasso.stability` *Stability and randomised lasso*

---

### Description

point-wise controled lasso stability selection

### Usage

```
lasso.stability(y, x=NULL, alpha=.5, subsampling=.5, nSubsampling=200, model=linear, pi_th=.6,  alpha.
```

### Arguments

| | |
|---|---|
| `y` | A vector of gene expression of a probe, or a list object if x is NULL. In the latter case y should a list of two components y and x, y is a vector of expression and x is a matrix containing copy number variables |
| `x` | Either a matrix containing CN variables or NULL |
| `alpha` | weakness parameter: control the shrinkage of regulators, if alpha = 1 then no randomisation, if NULL then a randomly generated vector is used |
| `subsampling` | fraction of samples to use in the sampling process, default to 0.5 |
| `nSubsampling` | The number of subsampling to do, default to 200 |
| `model` | which model to use, one of "cox", "logistic", "linear", or "poisson". Default to 'linear' |
| `pi_th` | The threshold of the stability probablity for selecting a regulator. It is to determine whether a coefficient is non-zero based on the frequency it is subsampled to be non-zero, default to 0.6 |
| `alpha.fwer` | Parameter to control for the FWER, choosing alpha.fwer and alpha control the E(V), V being the number of noise variables, eg. when alpha=0.9, alpha.fwer = 1 control the E(V)<=1 |
| `lambda1` | minimum lambda to use |
| `steps` | parameter to be passed on to penalized |
| `track` | track the progress, 0 none tracking, 1 minimum amount of information and 2 full information |
| `standardize` | standardize the data or not? |
| `...` | |

### Details

The function first selects lambda that approximately give maximum sqrt(.8*p) predictors, while p is the number of total predictors. Then it runs lasso a number of times keeping lambda fixed. These runs are randomised with scaled predictors and subsamples. At the end, the non-zero coefficients are determined by their frequencies of selections.

### Value

A list object of class 'lol', consisting of:

| | |
|---|---|
| `beta` | coefficients |
| `beta.bin` | binary beta vector as thresholded by pi_th |
| `mat` | the sampling matrix, each column is the result of one sampling |
| `residuals` | residuals of regression model |

### Author(s)

Yinyin Yuan

### References

N. Meinshausen and P. Buehlmann (2010), Stability Selection (with discussion), Journal of the Royal Statistical Society, Series B, 72, 417-473.

### See Also

lasso

### Examples

```
data(chin07)
data <- list(y=chin07$ge[1,], x=t(chin07$cn))
res <- lasso.stability(data, nSubsampling=50)
res
```

---

## lmMatrixFit *Multiple lm fit for penalized regressions*

---

### Description

Refit the regressions given matrices of responses, predictors, and the coefficients/interactions matrix. This is typically used after the lasso, since the coefficients were shrinked.

### Usage

```
lmMatrixFit(y, x = NULL, mat, th = NULL)
```

### Arguments

| | |
|---|---|
| `y` | Input response matrix, typically expression data with genes/variables in columns and samples/measurements in rows. Or when input x is NULL, y should be an object of two lists: y: expression data and x: copy number data |
| `x` | Input predictor matrix, typically copy number data, genes/predictors in columns and samples/measurements in rows. Can be NULL |
| `mat` | Coefficient matrix, number of columns is the number of predictors (y) and number of rows is the number of responses (x) |
| `th` | The threshold to use in order to determine which coefficients are non-zero, so the corresponding predictors are used |

### Value

| | |
|---|---|
| `coefMat` | A coefficient matrix, rows are responses and columns are predictors |
| `resMat` | A residual matrix, each row is the residuals of a response. |
| `pvalMat` | Matrix of p-values for each coefficients |

### Author(s)

Yinyin Yuan

### See Also

lm, matrixLasso

### Examples

```
data(chin07)
data <- list(y=t(chin07$ge), x=t(chin07$cn))
res <- matrixLasso(data, method=cv, nFold=5)
res
res.lm <- lmMatrixFit(y=data, mat=abs(res$coefMat), th=0.01)
res.lm
```

---

## matrixLasso *A wrapper function for matrix-to-matrix Lasso regressions*

---

### Description

This function wraps up different types of lasso optimizers and perform multiple, independent lasso inference on matrix responses. If the dimensionality of the input is small, the function converts the matrix of input response into a vector and solves the problem with one lasso inference. Otherwise, lasso regression is performed independently for each variables in the response matrix.

### Usage

```
matrixLasso(y, x=NULL, method=cv, nameControl=FALSE, standardize=FALSE, track=0, lambda1=NULL, nFold=1
```

### Arguments

| | |
|---|---|
| `y` | Input response matrix, typically expression data with genes/variables in columns and samples/measurements in rows. Or when input x is NULL, y should be an object of two lists: y: expression data and x: copy number data |
| `x` | Input predictor matrix, typically copy number data, genes/predictors in columns and samples/measurements in rows. Can be missing if the data is input to y. |
| `method` | Which optimization method to use for lasso inference, such as 'cv', 'stability', 'simultaneous', and 'multiSplit'. |
| `nameControl` | If the same item appears in both responses and predictors, the regression should remove the one same as the response from the predictors. This happens when for example a single data type is use for inferring gene network from expression data. Enable nameControl in this case. |
| `standardize` | Option to standardize the data, default to TRUE |
| `track` | Option to display progress, default to 0, 1 gives a brief summary of each fit, and 2 gives the full detail. |

`lambda1` The minimum lambda to use, default to NULL for which the program will select it automatically

`nFold` Number of folds for cross-validation, default to 10

`...`

## Value

`coefMat` A coefficient matrix, rows are responses and columns are predictors

`fit` If only a single regression is used for matrix lasso, the fit return.

`resMat` A residual matrix, each row is the residuals of a response.

## Author(s)

Yinyin Yuan

## See Also

lasso

## Examples

```
data(chin07)
data <- list(y=t(chin07$ge), x=t(chin07$cn))
res <- matrixLasso(data, method=cv, nFold=5)
res
```

---

`plotGW` *Plot genome-wide data along the genome*

---

## Description

Plot different measurements across the genome such as copy number amplifications and deletions.

## Usage

```
plotGW(data, pos, marks=NULL, fileType=png, file=plotGW, width=1000, height=500,  autoscale=FALSE, col
```

## Arguments

`data` data matrix to plot, each column is plotted individually across the genome

`pos` the chromosome locations for the data, can be a matrix or data frame with a column named chromosome_name, or a numeric vector

`marks` if there is specific marks to plot on the baselne, eg. to indicate where are the SNPs, should be a vector of numbers indicating where the marks is relative to the input data matrix

`fileType` either png or pdf file type

| | |
|---|---|
| `file` | file name |
| `width` | width of the plot |
| `height` | height of the plot |
| `autoscale` | should the columns of data be scaled? |
| `col` | colors for each of the data columns to be plotted, should be no shorter than the number of columns in 'data' |
| `legend` | legend text in the legend box |
| `ylab` | parameter for par, default to " |
| `pch` | parameter for par, default to 19 |
| `cex.axis` | parameter for par, default to 1.2 |
| `cex.lab` | parameter for par, default to 1.2 |
| `cex` | parameter for par, default to 0.5 |
| `legend.pos` | parameter for legend, default to 'bottomright' |
| `mtext` | parameter for mtext, default to NULL |
| `mtext.side` | parameter for mtext, default to 2 |
| `mtext.at` | parameter for mtext, default to 2 |
| `mtext.line` | parameter for mtext, default to 3 |
| `...` | Other parameters to pass to plot() or legend() |

## Details

This function requires as input data a vector or a matrix with different variables in columns, and a position matrix of chromosome name and start position. The number of rows in the position matrix should be the same as the length of the data vector or the number of rows of the data matrix. The function plots the data according to the position across the genome, providing a genome-wide description.

## Value

Write an image file to disk, either in png or pdf format.

## Author(s)

Yinyin Yuan

## See Also

lasso.cv

## Examples

```
data(chin07)
gain <- rowSums(chin07$cn >= .2)
loss <- -rowSums(chin07$cn <= -.2)
plotGW(data=cbind(gain, loss), pos=attr(chin07$cn, chrome), legend=c(gain, loss))
```

## print.lol — *print function for class lol*

### Description

print function for class lol

### Usage

```
print.lol(x,...)
```

### Arguments

| | |
|---|---|
| x | an object of class lol |
| ... | other parameters for consistency |

### Author(s)

Yinyin Yuan

## print.lolMatrix — *print function for class lolMatrix*

### Description

print function for class lolMatrix

### Usage

```
print.lolMatrix(x, ...)
```

### Arguments

| | |
|---|---|
| x | an object of class lolMatrix |
| ... | other parameters for consistency |

### Author(s)

Yinyin Yuan

# Index