R package lol

Lots Of Lasso: Stable network inference for integrative genomic analysis

Yinyin Yuan *
http://www.markowetzlab.org/software/lol
yy341@cam.ac.uk

24th Feb 2011

1 Introduction

lol is a package providing various optimization methods for Lasso inference. As popular tools in genomics, Lasso has been used in eQTL, GWAS, and studies of similar kind. It provides a prominent basis in high-dimensional studies, because of its efficiency in performing statistical inference on thousands of variables.

Yet despite its popularity, Lasso inference can be problematic if not optimized properly. Solving the optimisation problem for L1 regression often involves cross-validation. With noisy predictors, cross-validation tends to overfit and produce unstable result. Hence, in the context of genomic marker selection using microarray data, we implemented different optimization methods, i.e, simultanous lasso, stability selection, multisplit lasso, and cross-validation, for increased robustness and stable outcome.

We exemplify the use of this package using a breast cancer dataset comprising CNA and mRNA [1]. The data set includes (i) genome-wide DNA variation, (ii) expression as an intermediate trait.

Load the package 'lol' and import the breast cancer data set.

```
> library(lol)
> data(chin07)
> dim(chin07$cn)

[1] 339 106

> dim(chin07$ge)

[1]   7 106
```

We can visualize in an approximate manner what is the copy number alterations across the genome in this data set, using the plotGW function.

```
> gain <- rowSums(chin07$cn >= .2)
> loss <- -rowSums(chin07$cn <= -.2)
> plotGW(data=cbind(gain, loss), pos=attr(chin07$cn, 'chrome'), file='plotGWCN',
+ fileType='pdf', legend=c('gain', 'loss'), col=c('darkred', 'darkblue'))

null device
          1
```

2 Genomic marker selection for individual response/gene expression

For a genes's expression profile probed by microarray, we search for the genomic markers whose copy number alterations influences this gene. The 'lasso' function incorporates four different optimizers, each can be accessed by specifying the class of input object as one of the optimizers. If cross-validation is prefered, we can use the 'lasso' function with a data object of class 'cv', Because we will be using a lot of resampling, here we set the seed first.

*Cambridge Research Institute - CRUK, Li Ka Shing Centre, Robinson Way Cambridge, CB2 0RE, UK.

```
> Data <- list(y=chin07$ge[1,], x=t(chin07$cn))
> class(Data) <- 'cv'
> set.seed(10)
> res.cv <- lasso(Data)
> res.cv

 Non-zero coefficients: 0
 from a total of 339 predictors
```

This function optimizes lasso solution for correlated regulators by first choosing the minimum lambda, since the penalized package by default use 0 for the minimum which sometimes take a long time to compute due to co-linearity.

Alternatively, we can use the stability lasso [2] that achieves stable output by sub-sampling. The function first selects lambda that approximately give maximum $\sqrt{0.8 \times p}$ 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.

```
> class(Data) <- 'stability'
> res.stability <- lasso(Data)
> res.stability

 Non-zero coefficients: 0
 from a total of 339 predictors
```

Another optimzer performs the multi-split lasso as proposed in [3]. 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.

```
> class(Data) <- 'multiSplit'
> res.multiSplit <- lasso(Data)
> res.multiSplit

 Non-zero coefficients: 1
 from a total of 339 predictors
```

The fourth optimizer, simultanous lasso, performs multiple runs of lasso, each time splitting samples randomly to two equal sets, run lasso on both sets, then select those coefficients that are simultaneously non-zero across two sets.

```
> class(Data) <- 'simultaneous'
> res.simultaneous <- lasso(Data)
> res.simultaneous

 Non-zero coefficients: 2
 from a total of 339 predictors
```

Using 'plotGW', we can also visualize results from different optimizers. While any non-zero coefficients from cross-validation can be deemed significant, results from subsampling-based optimizers such as stability lasso and simultaneous lasso have to pass a significance level such as 60% selection frequency in order to be considered significant. For more details see [2].

```
> plotGW(data=cbind(res.cv$beta, res.stability$beta, res.multiSplit$beta, res.simultaneous$beta),
+ pos=attr(chin07$cn, 'chrome'), file='plotGWLassoOptimizers', fileType='pdf', width=8,
+ height=5, legend=c('cv', 'stability', 'multiSplit', 'simultaneous'), legend.pos='topleft')

null device
          1
```

3 Genomic marker selection for multiple responses/gene expression

Now if the goal is to infer a network of copy number driving expression, with multiple expression profiles, we can use the function matrixLasso. This wrapper function can use 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.

```
> Data <- list(y=t(chin07$ge), x=t(chin07$cn))
> res1 <- matrixLasso(Data, method='cv')
> res1

 Non-zero coefficients in total: 46
 from a total of 339 predictors
 and 7 responses

> res2 <- matrixLasso(Data, method='stability')
> res2

 Non-zero coefficients in total: 2
 from a total of 339 predictors
 and 7 responses
```

At the end, we recommend refitting regression models with the selected predictors using lm() so that the coefficients are not shrinked.

```
> res.lm <- lmMatrixFit(y=Data, mat=abs(res2$coefMat), th=0.01)
> res.lm

 Non-zero coefficients in total: 2
 from a total of 339 predictors
 and 7 responses
```

4 Session Information

```
R version 3.4.0 (2017-04-21)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 16.04.2 LTS

Matrix products: default
BLAS: /home/biocbuild/bbs-3.5-bioc/R/lib/libRblas.so
LAPACK: /home/biocbuild/bbs-3.5-bioc/R/lib/libRlapack.so

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] lol_1.24.0       Matrix_1.2-9     penalized_0.9-50 survival_2.41-3

loaded via a namespace (and not attached):
[1] compiler_3.4.0  tools_3.4.0     Rcpp_0.12.10    splines_3.4.0
[5] grid_3.4.0      lattice_0.20-35
```

References

[1] Suet Chin, Andrew Teschendorff, John Marioni, Yanzhong Wang, Nuno Barbosa-Morais, Natalie Thorne, Jose Costa, Sarah Pinder, Mark van de Wiel, Andrew Green, Ian Ellis, Peggy Porter, Simon Tavare, James Brenton, Bauke Ylstra, and Carlos Caldas. High-resolution acgh and expression profiling identifies a novel genomic subtype of er negative breast cancer. *Genome Biology*, 8(10):R215, 2007.

[2] Nicolai Meinshausen and Peter Buehlmann. Stability selection. *Journal of the Royal Statistical Society Series B-Statistical Methodology*, 72:417–473, 2010.

[3] Nicolai Meinshausen, Lukas Meier, and Peter Buehlmann. p-values for high-dimensional regression. *Journal of the American Statistical Association*, 104(488):1671–1681, December 2009.