An Introduction to the 'spls' Package, Version 1.0

Dongjun Chung[1], Hyonho Chun[1] and Sündüz Keleş[1,2]
[1]Department of Statistics, University of Wisconsin
Madison, WI 53706.
[2]Department of Biostatistics and Medical Informatics, University of Wisconsin
Madison, WI 53706.

August 28, 2024

1 Overview

This vignette provides basic information about the 'spls' package. SPLS stands for "Sparse Partial Least Squares". The SPLS regression methodology is developed in [1]. The main principle of this methodology is to impose sparsity within the context of partial least squares and thereby carry out dimension reduction and variable selection simultaneously. SPLS regression exhibits good performance even when (1) the sample size is much smaller than the total number of variables; and (2) the covariates are highly correlated. One additional advantage of SPLS regression is its ability to handle both univariate and multivariate responses.

The package can be loaded with the command:

```
R> library("spls")
```

2 Input Data

The package requires that the response is given in the form of a either vector or matrix, and the predictors in the form of a matrix. The response can be either univariate or multivariate. The responses and the predictors are assumed to be numerical and should not contain missing values. As part of pre-processing, the predictors are centered and scaled and the responses are centered automatically as default by the package 'spls'.

We provide the Yeast Cell Cycle dataset as an example application for the 'spls' package. The responses are the cell cycle gene expression data of 542 genes [5] from an α factor based experiment. In this experiment, mRNA levels were measured at every 7 minute during 119 minutes. Hence, the response has a total of 18 measurements covering two cell cycle periods. These 18 measurements correspond to 18 columns of the response matrix in the dataset. The predictors are chromatin immunoprecipitation on chip (ChIP-chip) data of [4] and they contain the binding information for 106 transcription factors (TF). This application is concerned with identifying cell cycle related TFs, i.e., TFs whose binding events contribute to explaining the variability in gene expression, as well as inferring their activities. See [1] for more details.

The yeast cell cycle dataset with the 'y' matrix as the cell cycle gene expression (responses) and the 'x' matrix as the ChIP-chip data (predictors) can be loaded as follows:

```
R> data(yeast)
R> yeast$x[1:5,1:5]
```

```
      ABF1_YPD    ACE2_YPD      ADR1_YPD  ARG80_YPD   ARG81_YPD
21  -0.2722730  0.21932294  0.9238359567 -0.4755756 -0.10389318
41   0.1691280  0.53831198  0.0097604993 -0.3219534 -0.19750606
71  -0.1388962  0.02636382  0.0877516229 -0.2234093  0.10307741
78  -0.2865169 -0.31409427 -0.0454998435  0.3262217  0.27757502
102 -0.4950561 -0.14827419  0.0002987512 -0.2179458 -0.02539585
```

```
R> yeast$y[1:5,1:5]
```

```
  alpha0 alpha7 alpha14 alpha21 alpha28
1  -0.36  -0.42    0.29   -0.14   -0.19
2   1.04   0.19    0.47   -1.03   -0.63
5  -0.30  -0.45    0.75    0.37    0.27
8  -0.46   0.12   -0.06   -0.76   -0.70
9  -1.35  -0.86   -0.22   -0.38   -0.65
```

3 Tuning Parameters

SPLS regression has two main tuning parameters: 'eta' represents the sparsity tuning parameter and 'K' is the number of hidden (latent) components. Parameters can be chosen by (v-fold) cross-validation using the function 'cv.spls'. The user specifies the range for these parameters and the cross-validation procedure searches within these ranges. 'eta' should have a value between 0 and 1. 'K' is integer valued and can range between 1 and $min\{p, (v-1)n/v\}$, where p is the number of predictors and n is the sample size. For example, if 10-fold cross-validation is used (default), 'K' should be smaller than $min\{p, 0.9n\}$. For the yeast data, we search for 'K' between 5 and 10 and for 'eta' between 0.1 and 0.9 with the following command:

```
R> set.seed(1)
R> cv <- cv.spls( yeast$x, yeast$y, eta = seq(0.1,0.9,0.1), K = c(5:10) )
```

'cv.spls' returns a heatmap-type plot of mean squared prediction error (MSPE) and the optimal values for 'eta' and 'K'. MSPE plot is given in Figure 1 and 'cv.spls' recommends to use 'eta=0.7' and 'K=8'.

4 SPLS Fit

Using the parameters obtained from 'cv.spls', SPLS can be fitted by the function 'spls'. 'spls' also prints out the variables that join the set of selected variables at each iteration step of SPLS fit. 'print.spls' displays the parameters used, the number of selected predictors, and the list of the selected predictors. 'coef.spls' prints out the coefficient estimates of SPLS fits.

```
R> f <- spls( yeast$x, yeast$y, eta = cv$eta.opt, K = cv$K.opt )
R> print(f)

Sparse Partial Least Squares for multivariate responses
----
Parameters: eta = 0.6, K = 8, kappa = 0.5
PLS algorithm:
```

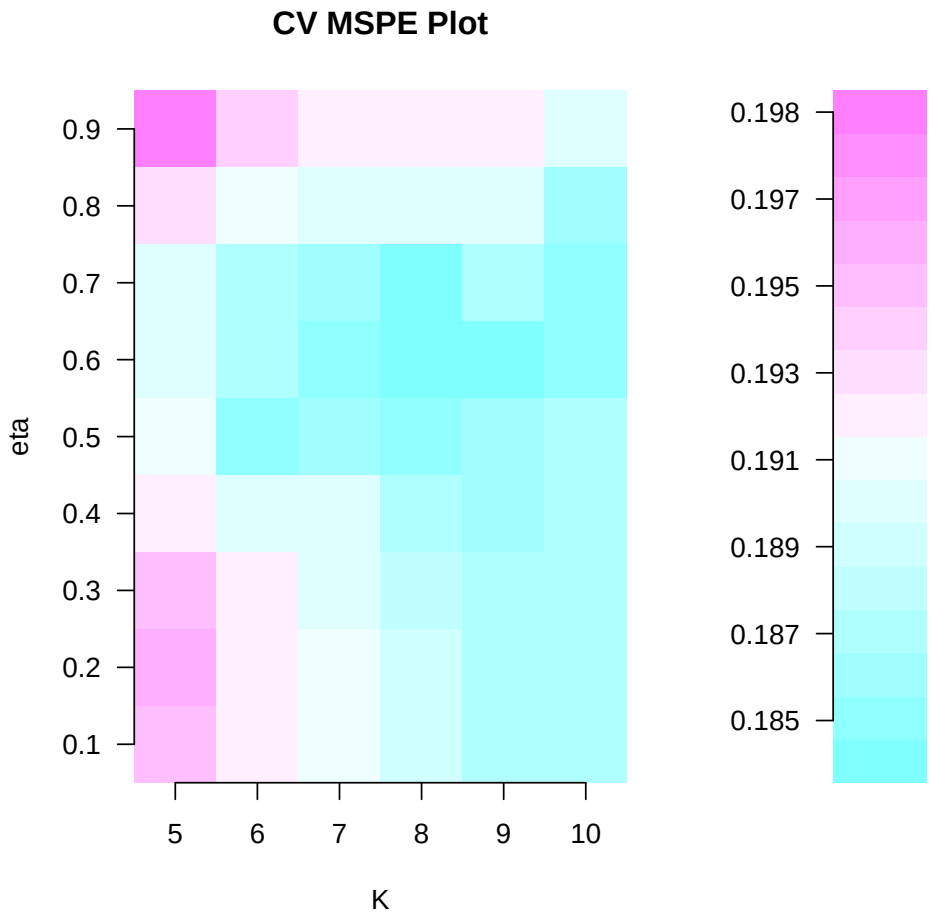

Figure 1: MSPE plot of SPLS when eta = 0.1–0.9 and K = 5–10.

```
pls2 for variable selection, simpls for model fitting

SPLS chose 56 variables among 106 variables

Selected variables:
ACE2_YPD        ARG80_YPD        ARG81_YPD        ASH1_YPD        AZF1_YPD
BAS1_YPD        CBF1_YPD        CHA4_YPD        CRZ1_YPD        FHL1_YPD
FKH1_YPD        FKH2_YPD        FZF1_YPD        GAT1_YPD        GAT3_YPD
GCN4_YPD        GCR2_YPD        GLN3_YPD        HAA1_YPD        HAP2_YPD
HAP5_YPD        HIR1_YPD        HIR2_YPD        IME4_YPD        INO4_YPD
A1..MATA1._YPD        MBP1_YPD        MCM1_YPD        MET4_YPD        MSN2_YPD
NDD1_YPD        NRG1_YPD        PHD1_YPD        PHO2_YPD        PUT3_YPD
RCS1_YPD        REB1_YPD        RFX1_YPD        RIM101_YPD        RME1_YPD
RTG1_YPD        RTG3_YPD        SIP4_YPD        SOK2_YPD        STB1_YPD
STE12_YPD        STP2_YPD        SWI4_YPD        SWI5_YPD        SWI6_YPD
THI2_YPD        YAP1_YPD        YAP6_YPD        YAP7_YPD        YFL044C_YPD
```

```
YJL206C_YPD
```

```
R> coef.f <- coef(f)
R> coef.f[1:5,1:5]
```

```
              alpha0       alpha7    alpha14       alpha21      alpha28
ABF1_YPD   0.0000000  0.000000000 0.00000000  0.0000000000  0.000000000
ACE2_YPD   0.0874325  0.068452293 0.01374781 -0.0002541969 -0.033302624
ADR1_YPD   0.0000000  0.000000000 0.00000000  0.0000000000  0.000000000
ARG80_YPD -0.0486881 -0.019092797 0.02063442  0.0300421634  0.007925553
ARG81_YPD -0.0168849  0.009465868 0.06353825  0.0541704059  0.006978985
```

A plot that illuminates the fitting procedure is the coefficient path plot given in Figure 2. This plot illustrates how the coefficient estimates change as a function of '`K`' for a given '`eta`'. The coefficient path plot for a specific value of '`eta`' can be obtained using the function '`plot.spls`'. When there are many responses, it might not be a good idea to plot them all together. The response of interest can be defined with the option '`yvar`'. If the option '`yvar`' is not specified, then '`plot.spls`' draws the coefficient paths of all responses. The command below generates the coefficient path plot given in Figure 2.

```
R> plot.spls( f, yvar=1 )
```

In the yeast cell cycle data, the responses were repeatedly measured at different time points. In this case, it is useful to visualize how the estimated coefficients change as a function of time. The function '`coefplot.spls`' plots the estimated coefficients of the fit obtained from the '`spls`' function across all the responses. By default, '`coefplot.spls`' displays the estimated coefficients of the selected predictors. However, in the case that too many predictors are chosen, this might lead to a crowded plotting area. We provide two options to avoid this. First, one can choose predictors to be plotted by the option '`xvar`'. Note that the index number here is defined among the selected variables. For example, '`xvar = 1`' refers to the first variable among the selected predictors. In this example, '`xvar`' can be between 1 and 28. Second, one can also control the number of plots that appear in each window using the option '`nwin`'. For example, '`nwin = c(2,2)`' indicates that the plotting area will have 4 plots with two rows and two columns. An illustration of this plotting option is given below and the resulting plots are displayed in Figure 3.

```
R> coefplot.spls( f, nwin=c(2,2), xvar=c(1:4) )
```

5 eQTL Application

In this section, we study the application of SPLS regression to the expression quantitative trait loci (eQTL) mapping. We provide the mice dataset [3] as an example of the eQTL application. Mice were collected from a F_2-*ob/ob* cross and lacked a functional leptin protein hormone. The functional leptin protein hormone is known to be important for reproduction and regulation of body weight and metabolism. The predictors are the marker map consisting of 145 microsatellite markers from 19 non-sex mouse chromosomes. The responses are the gene expression measurements of 83 transcripts from liver tissues of 60 mice. This group of 83 transcripts was obtained as one of the clusters, when 45,265 transcripts were clustered using a hierarchical clustering approach. See [2] for more details.

The mice dataset with the '`y`' matrix as the gene expression (responses) and the '`x`' matrix as the marker map (predictors) can be loaded as follows:

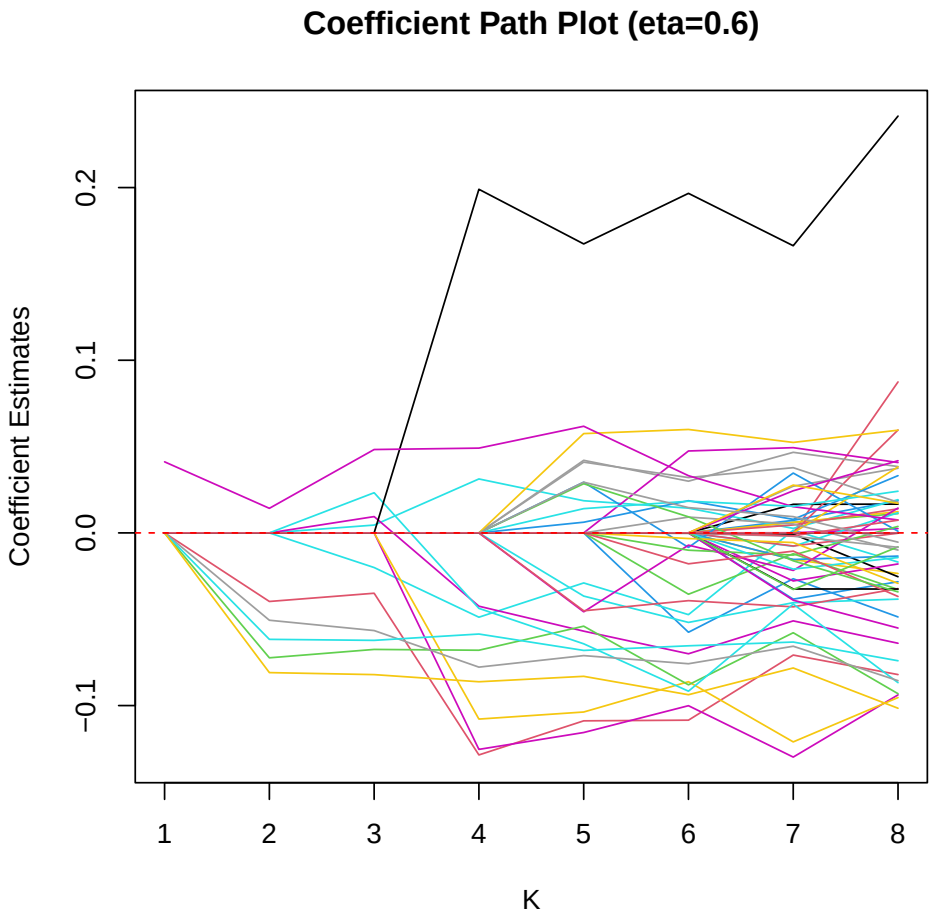

Figure 2: Coefficient path plot of the SPLS fit for the yeast data when eta=0.7.

```
R> data(mice)
R> mice$x[1:5,1:5]

  D1Mit64 D1Mit211 D1Mit303 D1Mit46 D1Mit87
1       2        2        2       2       1
2       2        2        2       2       2
3       2        2        2       2       2
4       2        2        2       3       3
5       2        1        3       3       3

R> mice$y[1:5,1:5]

  1415889_a_at 1415965_at 1416308_at 1417017_at 1417208_at
2         6.82      10.56       9.79       7.96       8.34
3         8.28      10.26      10.31       8.86       9.35
4         9.10      11.42       9.92       9.13       9.34
5         8.74      10.99      10.56       8.61       9.31
6         7.16       8.56      10.08       9.66       8.28
```

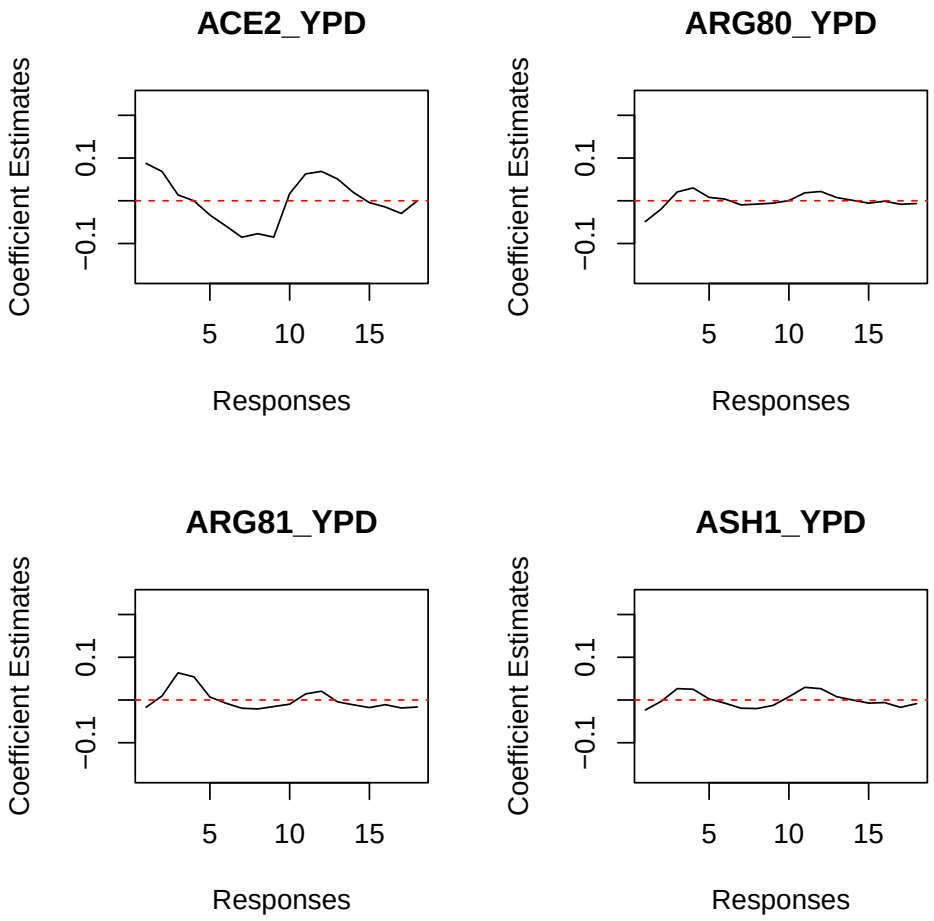

Figure 3: Plot of the estimated coefficients.

The optimal parameters were chosen as '`eta=0.6`' and '`K=1`' from '`cv.spls`' as follows.

```
R> set.seed(1)
R> cv <- cv.spls( mice$x, mice$y, eta = seq(0.1,0.9,0.1), K = c(1:5) )
```

SPLS fits are obtained as below.

```
R> f <- spls( mice$x, mice$y, eta = cv$eta.opt, K = cv$K.opt )
R> print(f)

Sparse Partial Least Squares for multivariate responses
----
Parameters: eta = 0.6, K = 1, kappa = 0.5
PLS algorithm:
pls2 for variable selection, simpls for model fitting

SPLS chose 30 variables among 145 variables
```

```
Selected variables:
D2Mit274        D2Mit17        D2Mit106        D2Mit194        D2Mit263
D2Mit51        D2Mit49        D2Mit229        D2Mit148        D5Mit348
D5Mit75        D5Mit267        D5Mit259        D5Mit9        D5Mit240
D5Mit136        D8Mit249        D8Mit211        D8Mit113        D9Mit206
D9Mit2        D9Mit21        D9Mit207        D9Mit8        D9Mit15
D9Mit18        D15Mit174        D15Mit136        D15Mit63        D15Mit107
```

In this eQTL analysis, we can improve the initial SPLS fits further as described in [2]. First, we obtain the bootstrapped confidence intervals for the coefficients of the selected predictors using the function '`ci.spls`'. '`ci.spls`' also provides the confidence interval plots and lets users to control the plots with two options, '`plot.fix`' and '`plot.var`'. If '`plot.fix="x"`', then the function '`ci.spls`' plots the confidence intervals of a given predictor specified by the option '`plot.var`' across all the responses. Similarly, '`plot.fix="y"`' draws the plot against the predictors for the given responses. Note that if '`plot.fix="x"`', then the index number is defined among the selected variables. Figure 4 shows the confidence interval plot of the coefficients.

```
R> set.seed(1)
R> ci.f <- ci.spls( f, plot.it=TRUE, plot.fix='x', plot.var=20 )
```

The function '`ci.spls`' returns the list whose element is the matrix of the confidence intervals of the coefficients. The names of elements of the list are the same as the column names of the responses.

```
R> cis <- ci.f$cibeta
R> cis[[20]][1:5,]
```

```
                2.5%      97.5%
D2Mit274 -0.004702266 0.02254638
D2Mit17  -0.005072756 0.02271000
D2Mit106 -0.004535333 0.02315933
D2Mit194 -0.005701689 0.02102956
D2Mit263 -0.005618393 0.02290074
```

After we obtain the confidence intervals of the coefficients, the function '`correct.spls`' updates the coefficient estimates of the selected variables by setting the coefficients with zero-containing confidence intervals to zero. In addition, '`correct.spls`' provides the heatmap-type plots of the original SPLS coefficient estimates (Figure 5) and the corrected coefficient estimates (Figure 6).

```
R> cf <- correct.spls( ci.f )
```

```
R> cf[15:20,1:5]
```

```
         1415889_a_at 1415965_at 1416308_at 1417017_at 1417208_at
D2Mit327            0 0.00000000          0          0          0
D2Mit35             0 0.00000000          0          0          0
D2Mit249            0 0.00000000          0          0          0
D2Mit274            0 0.00000000          0          0          0
D2Mit17             0 0.04682557          0          0          0
D2Mit106            0 0.05103286          0          0          0
```

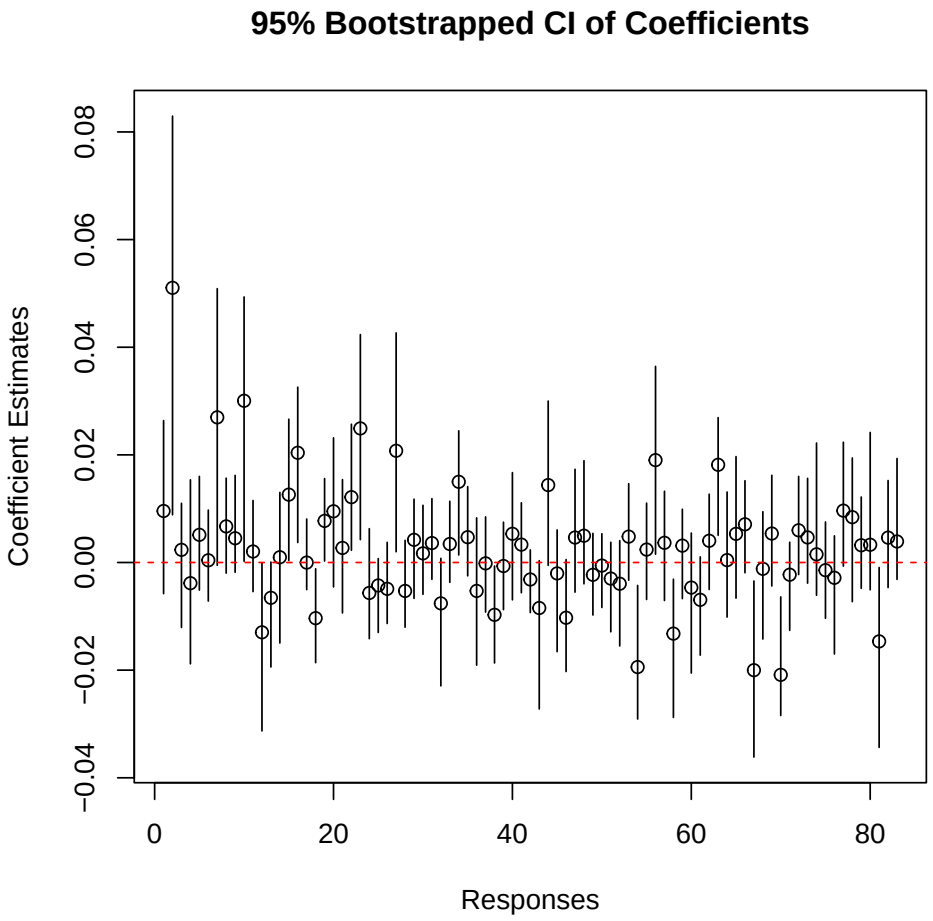

Figure 4: Example plot of the confidence intervals of the coefficients.

References

[1] Chun, H. and Keleş, S. (2007) "Sparse partial least squares for simultaneous dimension reduction and variable selection", (`http://www.stat.wisc.edu/~keles/Papers/SPLS_Nov07.pdf`).

[2] Chun, H. and Keleş, S. (2008). "Expression quantitative trait loci mapping with multivariate sparse partial least squares regression", (`http://www.stat.wisc.edu/~keles/Papers/chun_keles_eQTL_SPLS_submit.pdf`).

[3] Lan, H., M. Chen, J. B. Flowers, B. S. Yandell, D. S. Stapleton, C. M. Mata, E. T-K Mui, M. T. Flowers, K. L. Schueler, K. F. Manly, R. W. Williams, C. Kendziorski, and A. D. Attie (2006). "Combined expression trait correlations and expression quantitative trait locus mapping", *PLoS Genetics*, 2, e6.

[4] Lee, T. I., N. J. Rinaldi, F. Robert, D. T. Odom, Z. Bar-Joseph, G. K. Gerber, N. M. Hannett, C. T. Harbison, C. M. Thomson, I. Simon, J. Zeitlinger, E. G. Jennings, H. L. Murray, D. B. Gordon, B. Ren, J. J. Wyrick, J.-B. Tagne, T. L. Volkert, E. Fraenkel, D. K. Gifford, and R. A.

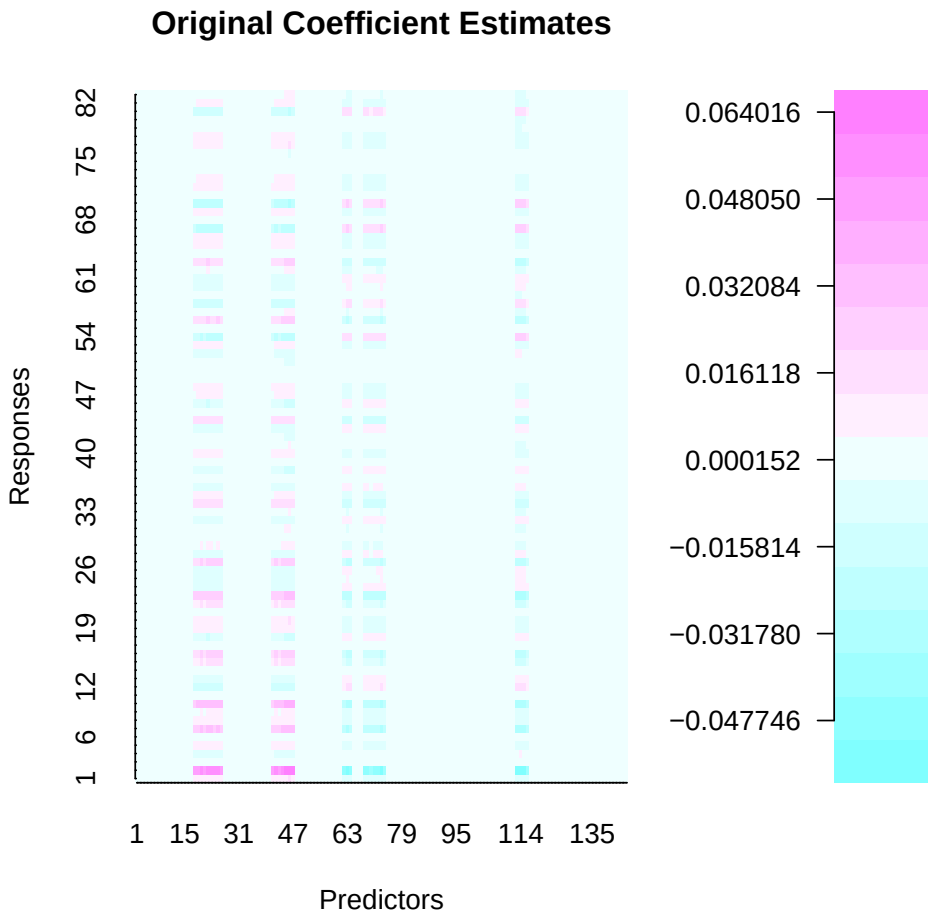

Figure 5: Plot of the original coefficient estimates.

Young (2002). "Transcriptional regulatory networks in *Saccharmomyces cerevisiae*". *Science*, **298**, pp. 799–804.

[5] Spellman, P. T., G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher (1998). "Comprehensive identification of cell cycle- regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hydrization". *Molecular Biology of the Cell*, **9**, pp. 3273–3279.

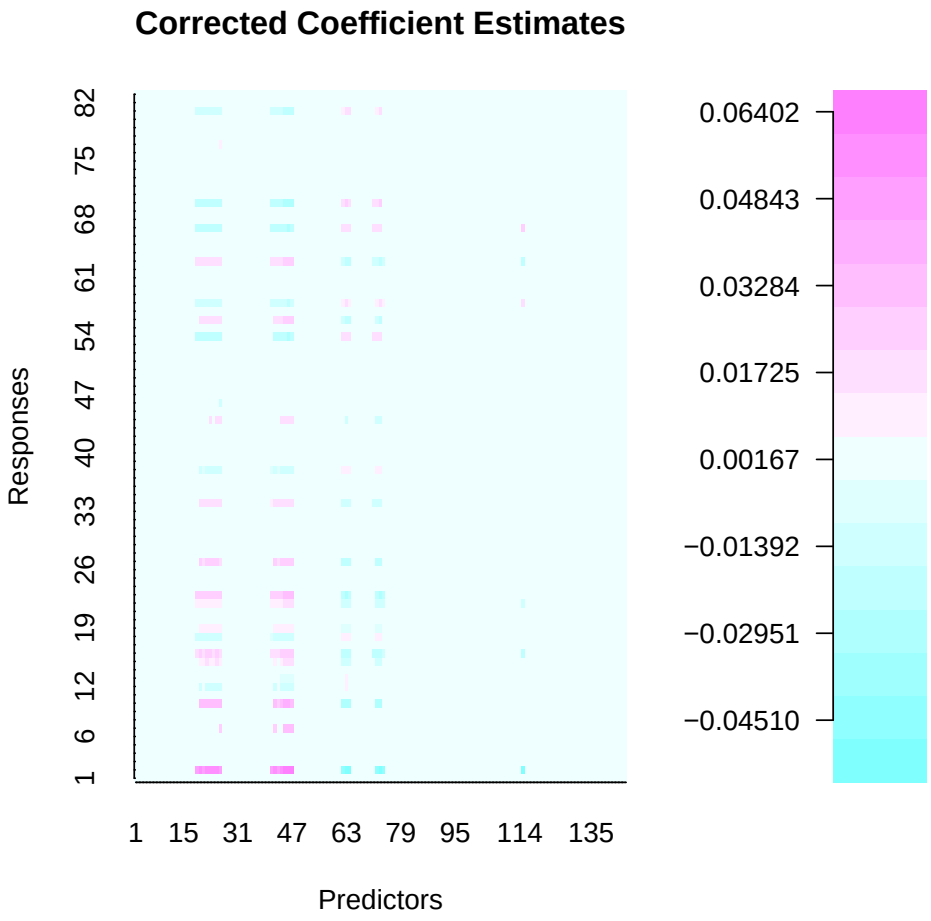

Figure 6: Plot of the corrected coefficient estimates.