Screen & Clean Tutorial

University of Pittsburgh Medical Center
Computational Genetics Lab

August, 2010

1 Introduction

Screen & Clean is an R procedure that identifies associations between SNP allele count data and a continuous or binary phenotype. The core function is a *screen* that identifies the first K SNPs to enter an L_1-penalized regression of the phenotype on the allele counts, where K is chosen by a stability criterion. The lasso regression is fit with the *glmnet* package, which must be downloaded from a CRAN server and installed before Screen & Clean will work. The program includes several optional procedures, including a *pre-screen* using marginal regression p-values, a second *screen* for pairwise interaction effects, and a multivariate regression *clean* of the screened SNPs. By default these extra methods are turned off.

To run the Screen & Clean procedure, open an R terminal and run the R script *screen_clean.R* with the command *source("screen_clean.R")*. Call the Screen & Clean procedures with the function *screenClean*:

```
screenClean(chr, pheno, L = NULL, K = NULL, n_samp = floor(10 * sqrt(dim(chr)[1])),
            K_pairs = NULL, response = "gaussian", standardize = TRUE, alpha = NULL,
            snp_fix = NULL, cov_struct = NULL)
```

The data for this tutorial are saved in the file *sc_tutorial.RData*. This file contains simulated allele counts for 1,000 people at 5,000 tag SNPs, a simulated binary phenotype, and several useful objects for evaluating the accuracy of the method. The last section of this tutorial contains a more detailed description of the data.

2 Core Function

The core method in *screenClean* is the screen phase. The user specifies the type of phenotype as *gaussian* or *binomial* and any structural covariates or "fixed SNPs" which the user wishes to include in all models. *screenClean* returns a list of the first K SNPs to enter a lasso-penalized regression of phenotype on allele counts. By default, K is chosen according to internally computed stability statistics, where the pre-set subsample size is $n\_samp = \lfloor 10\sqrt{n} \rfloor$ (here $n = 1,000$, so $n\_samp = 316$), but the user may change *n_samp*. The stability method is described in *Stability Approach to Regularization Selection (StARS) for High Dimensional Graphical Models* (Liu, et al. arXiv:1006.3316v1).

- Function call

  ```
  > sc1 = screenClean(chr = chr, pheno = pheno, response = "binomial",
      standardize = FALSE, snp_fix = id_fix, cov_struct = cov_struct)
  ```

  Note that *standardize* is set to false because our example data is already standardized. We include both structural covariates (*cov_struct*) and fixed SNPs (*id_fix*) but these may be omitted, or left as NULL. The fixed SNPs may be indentified by column index or column name.

- Program steps and screen output

  - `centering and scaling the additional structural covariates....`

    centers the structural covariates to mean 0 and scales to standard deviation 1. Imputes any missing values to be the mean.

  - `computing stability statistics to choose K....`

    ```
    fitting lasso for subsample 5 of 50....
    fitting lasso for subsample 10 of 50....
    fitting lasso for subsample 15 of 50....
    fitting lasso for subsample 20 of 50....
    fitting lasso for subsample 25 of 50....
    fitting lasso for subsample 30 of 50....
    fitting lasso for subsample 35 of 50....
    fitting lasso for subsample 40 of 50....
    fitting lasso for subsample 45 of 50....
    fitting lasso for subsample 50 of 50....
    k_star chosen by stability: 144
    ```

    computes stability statistics to choose K, the number of SNPs to return from the lasso screen. Stability is computed by fitting a lasso model for each of 50 subsamples of the data, so this process is somewhat slow.

  - `fitting lasso with glmnet....`

    screen for K SNPs with lasso, fit with the *glmnet* package.

- Function values

  - *snp_screen* - a vector of the column names of the SNPs picked by the screen. Typically these column names will be RS IDs for SNPs.

    ```
    > print(sc1$snp_screen)
      [1] "rs1"    "rs2"    "rs3"    "rs13"   "rs19"   "rs120"  "rs130"  "rs190"
      [9] "rs191"  "rs252"  "rs272"  "rs385"  "rs397"  "rs415"  "rs429"  "rs472"
     [17] "rs524"  "rs559"  "rs574"  "rs713"  "rs743"  "rs841"  "rs930"  "rs1008"
     [25] "rs1035" "rs1086" "rs1092" "rs1093" "rs1108" "rs1113" "rs1128" "rs1154"
     [33] "rs1172" "rs1217" "rs1232" "rs1263" "rs1311" "rs1340" "rs1398" "rs1427"
     [41] "rs1472" "rs1480" "rs1499" "rs1506" "rs1520" "rs1564" "rs1625" "rs1628"
     [49] "rs1658" "rs1682" "rs1699" "rs1797" "rs1821" "rs1850" "rs1913" "rs2016"
     [57] "rs2067" "rs2102" "rs2160" "rs2161" "rs2176" "rs2221" "rs2283" "rs2296"
     [65] "rs2339" "rs2374" "rs2385" "rs2440" "rs2470" "rs2509" "rs2548" "rs2620"
     [73] "rs2631" "rs2638" "rs2648" "rs2652" "rs2659" "rs2679" "rs2745" "rs2749"
     [81] "rs2762" "rs2820" "rs2844" "rs2901" "rs2905" "rs2909" "rs3101" "rs3115"
     [89] "rs3126" "rs3129" "rs3258" "rs3281" "rs3308" "rs3331" "rs3422" "rs3427"
     [97] "rs3434" "rs3529" "rs3616" "rs3624" "rs3631" "rs3633" "rs3649" "rs3677"
    [105] "rs3700" "rs3755" "rs3831" "rs3870" "rs3876" "rs3917" "rs3927" "rs3985"
    [113] "rs3992" "rs4005" "rs4072" "rs4206" "rs4214" "rs4220" "rs4242" "rs4255"
    [121] "rs4262" "rs4269" "rs4353" "rs4371" "rs4539" "rs4597" "rs4652" "rs4759"
    [129] "rs4782" "rs4818" "rs4833" "rs4866" "rs4893" "rs4907" "rs5039" "rs5058"
    [137] "rs5068" "rs5080" "rs5091"
    ```

- Evaluation

  As a check, we can see that all of the SNPs that are supposed to be forced into the models are retained by the lasso screen:

```
> table(id_fix %in% sc1$snp_screen)

TRUE
   9
```

We can also check the accuracy of the screen by tabulating the number of true effects in the screen results. Note that *snp_main* is a column index, so the ID numbers must first be pulled from the column labels of *chr*. The screen successfully finds 42 of the main effects.

```
> ids = colnames(chr)
> table(ids[snp_main] %in% sc1$snp_screen)

FALSE  TRUE
   12    48
```

The user may set K manually to circumvent the stability method:

- Function call

  ```
  > sc2 = screenClean(chr = chr, pheno = pheno, K = 100, response = "binomial",
      standardize = FALSE, snp_fix = id_fix, cov_struct = cov_struct)
  ```

- Function values

  - *snp_screen* - a vector of the column names of the screened SNPs.

3 General User Notes

- The lasso screen successfully chooses no more than one SNP in each region of high linkage disequilibrium, but often retains no SNPs at all even in regions with very strong marginal significance. Because of this, we recommend choosing tag SNPs prior to using Screen & Clean.
- To save time, fitting of the lasso models stops when the desired number of SNPs is reached. For binary response variables *glmnet* lasso produces a warning message when this happens before the full range of regularization parameter values has been fit. For example,

  ```
  2: from glmnet Fortran code - Number of nonzero coefficients along the path
  exceeds pmax=110 at 69th lambda value; solutions for larger lambdas returned
  ```

  These warnings should be ignored.

4 Additional Methods: Pairwise Interactions

Screen & Clean can look for main effects and pairwise interaction effects at the same time. The program first does a lasso screen to find K main effects, where K can be specified by the user or chosen internally by the stability criterion. The dictionary of candidate SNPs is then expanded to include all pairwise interactions between these K SNPs. A second lasso screen is then performed on this expanded dictionary, and the first *K_pairs* terms to enter the lasso model are retained. To look for pairwise interactions, the user must specify a number for *K_pairs*; it cannot be chosen by the stability criterion.

- Function call

  ```
  sc4 = screenClean(chr = chr, pheno = pheno, K = 100, K_pairs = 100, response = "binomial",
    standardize = FALSE, snp_fix = id_fix, cov_struct = cov_struct)
  ```

  Here, we specify K to be 100, circumventing the stability procedure. The search for pairwise interactions is triggered by setting *K_pairs* to 100.

- Program steps and screen output

  -
    ```
    centering and scaling the additional structural covariates....
    fitting lasso with glmnet....
    ```
    The program scales the structural covariates and does the lasso screen for main effects.

  -
    ```
    generating pairwise interactions....
    fitting lasso with glmnet....
    ```
    Interaction terms are generated for each pair of SNPs in the K selected in the main effects screen. These interactions, plus the orignal K main effects are then screened again, and *K_pairs* terms are retained.

- Function values
  - *snp_screen* - a vector of K SNPs retained by the first lasso screen.
  - *snp_screen2* - a vector of *K_pairs* SNPs and SNP pairs retained by the second lasso screen. *snp_screen2* is a data frame with *K_pairs* rows and three columns. The second and third columns indicate the members of the pairs; for main effects column three is "NA". The first column is an internal ID number for the SNPs and SNP pairs and should be ignored.

    ```
    > dim(sc4$snp_screen2)
    [1] 97 3

    > sc4$snp_screen2[70:85,]
       column   snp1   snp2
    70   4818 rs4818   <NA>
    71   4893 rs4893   <NA>
    72   4907 rs4907   <NA>
    73   5058 rs5058   <NA>
    74   5068 rs5068   <NA>
    75   5080 rs5080   <NA>
    76   5091 rs5091   <NA>
    77   5095 rs5095   <NA>
    78   5155    rs1 rs2820
    79   5178    rs1 rs4214
    80   5896  rs574 rs3422
    81   5897  rs574 rs3427
    82   5992  rs713 rs3917
    83   6215 rs1035 rs2283
    84   6222 rs1035 rs2638
    85   6399 rs1092 rs3126
    ```

5 Additional Methods: Marginal Regressions

If the number of SNPs in the data is very large, users may *pre-screen* them by specifying a value for L. Screen & Clean calculates the p-value from a marginal regression of the phenotype on each SNP (controlling for the structural covariates) and retains the SNPs with the L smallest p-values.

The pre-screen is not a necessary step, but is useful if the number of SNPs is very large. We believe most users will have done this step prior to using Screen & Clean, so it is omitted by default. It would typically not be useful to pre-screen a data set that has only 5,000 SNPs, but it is included in the tutorial for illustration purposes.

- Function call

```
> sc3 = screenClean(chr = chr, pheno = pheno, L = 2500, response = "binomial",
    standardize = FALSE, snp_fix = id_fix, cov_struct = cov_struct)
```

- Program steps and screen output

  ```
  centering and scaling the additional structural covariates....
  marginal regression pre-screen....
  snp: 500
  snp: 1000
  snp: 1500
  snp: 2000
  snp: 2500
  snp: 3000
  snp: 3500
  snp: 4000
  snp: 4500
  snp: 5000
  fitting lasso with glmnet....
  ```

- Function values

  - *snp_prescreen* - a vector of the SNPs retained by the marginal regression pre-screen.

    ```
    > length(sc3$snp_prescreen)
    [1] 2502

    > sc3$snp_prescreen[1:30]
    [1] "rs2374" "rs3427" "rs5068" "rs5067" "rs4446" "rs5064" "rs5062" "rs3529"
     [9] "rs5069" "rs5070" "rs5063" "rs5061" "rs5066" "rs1520" "rs5065" "rs1564"
    [17] "rs5029" "rs1821" "rs3129" "rs5028" "rs5024" "rs2762" "rs5027" "rs5025"
    [25] "rs5026" "rs1154" "rs5030" "rs5021" "rs5022" "rs1913"
    ```

  - *snp_screen* - a vector of SNPs also retained by the lasso screen.

If Screen & Clean is set to look for pairwise interaction effects and the number of screened SNPs (K) plus the number of computed pairwise SNP interactions $\binom{K}{2}$ is larger than L, the program will do a marginal regression pre-screening on the expanded SNP dictionary, whether or not an initial pre-screen was performed.

- Function call

  ```
  > sc5 = screenClean(chr = chr, pheno = pheno, L = 2500, K = 100, K_pairs = 100,
      response = "binomial", standardize = FALSE, snp_fix = id_fix, cov_struct = cov_struct)
  ```

- Program steps and screen output

  ```
  centering and scaling the additional structural covariates....
  marginal regression pre-screen....
  snp: 500
  snp: 1000
  snp: 1500
  snp: 2000
  snp: 2500
  snp: 3000
  snp: 3500
  snp: 4000
  snp: 4500
  snp: 5000
  fitting lasso with glmnet....
  ```

```
generating pairwise interactions....
marginal regression pre-screen....
snp: 500
snp: 1000
snp: 1500
snp: 2000
snp: 2500
snp: 3000
snp: 3500
snp: 4000
snp: 4500
fitting lasso with glmnet....
```

- Function values
  - *snp_prescreen* - a vector of SNPs retained by the marginal regression pre-screen.
  - *snp_screen* - a vector of SNPs retained by the lasso screen.
  - *snp_prescreen2* - because this involves pairwise interactions, this is a 3-column data frame. The first column is an ID number intended for internal use only.

    ```
    > sc5$snp_prescreen2[40:60,]
       column   snp1   snp2
    40   9289 rs4214    rs1
    41   5094 rs5094   <NA>
    42   6175 rs1850 rs3101
    43   1499 rs1499   <NA>
    44   5095 rs5095   <NA>
    45   5257 rs3427  rs574
    46   8645 rs3633  rs930
    47   3422 rs3422   <NA>
    48   2648 rs2648   <NA>
    49   1217 rs1217   <NA>
    50   2161 rs2161   <NA>
    51   5699 rs1821 rs5095
    52   5039 rs5039   <NA>
    53   2470 rs2470   <NA>
    54   6450 rs1093 rs2749
    55   5636 rs1564  rs841
    56   7878 rs4759 rs1628
    57   3126 rs3126   <NA>
    58   3308 rs3308   <NA>
    59   6400 rs1093 rs1108
    60   1108 rs1108   <NA>
    ```

  - *snp_screen2* - this is also a 3-column data frame because of the pairwise interactions.

    ```
    > sc5$snp_screen2[40:60,]
      column   snp1   snp2
    40   9289 rs4214    rs1
    41   5094 rs5094   <NA>
    42   6175 rs1850 rs3101
    43   1499 rs1499   <NA>
    44   5095 rs5095   <NA>
    45   5257 rs3427  rs574
    46   3422 rs3422   <NA>
    ```

```
47   2648 rs2648   <NA>
48   1217 rs1217   <NA>
49   5699 rs1821 rs5095
50   2470 rs2470   <NA>
51   6450 rs1093 rs2749
52   3126 rs3126   <NA>
53   3308 rs3308   <NA>
54   6400 rs1093 rs1108
55   2221 rs2221   <NA>
56   3631 rs3631   <NA>
57   3700 rs3700   <NA>
58   1035 rs1035   <NA>
59   7000 rs2820    rs1
60   4072 rs4072   <NA>
```

6 Additional Methods: Multivariate Regression

The SNPs retained by the screening phase (or second screening phase, if pairwise interactions are included) can be fed into a multivariate regression *clean* by specifying a value for *alpha*. This value is automatically divided by the number of terms in the regression; terms with p-values less than this Bonferroni-corrected *alpha* are retained. A final regression with only the clean SNPs is fit to get final p-values.

- Function call:

```
> sc6 = screenClean(chr = chr, pheno = pheno, K = 100, response = "binomial",
    standardize = FALSE, alpha = 0.05, snp_fix = id_fix, cov_struct = cov_struct)
```

- Program steps and screen output

```
centering and scaling the additional structural covariates....
fitting lasso with glmnet....
fitting multivariate regression clean....
fitting final regression....
```

- Function values:
  - *snp_screen* - a vector of SNPs retained by the lasso screen.
  - *snp_clean* - a vector of screened SNPs also retained by the multivariate regression clean.
  - *clean* - a data frame with regression output for all of the screened SNPs. The *snp2* column accomodates pairwise interaction effects, if applicable. For main effects, this is left as “NA”. The *type* column indicates if the covariate has a special status. The codes are:
    - ∗ 0 - no special status. A regular main effect or pairwise interaction effect.
    - ∗ 1 - a fixed SNP. Forced into the lasso and multivariate regression models.
    - ∗ 2 - a structural covariate. Forced into all models.
    - ∗ 3 - other. The intercept is the only term that should carry this label.

```
> dim(sc6$clean)
[1] 100  8

> sc5$clean[1:20,]
   column      snp1 snp2    Estimate Std..Error    z.value     Pr...z.. type
1    <NA> intercept <NA>  0.03533726  0.1244216  0.2840122 7.764010e-01    3
2       1       rs1 <NA> -0.10304250  0.1290410 -0.7985255 4.245656e-01    1
```

```
3       2       rs2 <NA>  0.21695985  0.1362569  1.5922855 1.113206e-01    1
4       3       rs3 <NA>  0.06497387  0.1322525  0.4912865 6.232238e-01    1
5      19      rs19 <NA>  0.26617513  0.1282268  2.0758145 3.791111e-02    0
6     120     rs120 <NA> -0.47162395  0.1319360 -3.5746432 3.507059e-04    0
7     272     rs272 <NA> -0.43708733  0.1358787 -3.2167473 1.296527e-03    0
8     415     rs415 <NA>  0.41857063  0.1339079  3.1258098 1.773162e-03    0
9     472     rs472 <NA>  0.60067499  0.1399917  4.2907900 1.780386e-05    1
10    574     rs574 <NA>  0.45851500  0.1347650  3.4023293 6.681408e-04    0
11    713     rs713 <NA>  0.45680711  0.1302949  3.5059468 4.549861e-04    0
12    841     rs841 <NA>  0.51562066  0.1379208  3.7385281 1.851008e-04    0
13    930     rs930 <NA>  0.37101002  0.1289070  2.8781225 4.000498e-03    0
14   1035    rs1035 <NA>  0.37609664  0.1322252  2.8443642 4.450015e-03    0
15   1086    rs1086 <NA>  0.51309453  0.1356456  3.7826122 1.551911e-04    0
16   1092    rs1092 <NA> -0.31719016  0.1351517 -2.3469189 1.892937e-02    0
17   1093    rs1093 <NA>  0.64309622  0.1361612  4.7230512 2.323323e-06    0
18   1108    rs1108 <NA>  0.04911318  0.1276722  0.3846820 7.004730e-01    0
19   1113    rs1113 <NA>  0.43935179  0.1394885  3.1497349 1.634187e-03    0
20   1128    rs1128 <NA>  0.49119834  0.1376080  3.5695482 3.575974e-04    0
```

- *final* - a data frame with output from the regression of phenotype on the final cleaned SNPs. The columns and codes are identical to the *clean* regression output.

```
> dim(sc5$final)
[1] 27  8

> sc5$final
   column      snp1 snp2     Estimate Std..Error    z.value     Pr...z.. type
1    <NA> intercept <NA>  0.008202847 0.08051433  0.1018806 9.188515e-01    3
2     120     rs120 <NA> -0.263394637 0.08285415 -3.1790156 1.477761e-03    0
3     472     rs472 <NA>  0.234971561 0.08213643  2.8607473 4.226437e-03    1
4     713     rs713 <NA>  0.312885127 0.08274707  3.7812232 1.560597e-04    0
5     841     rs841 <NA>  0.233547799 0.08264819  2.8258067 4.716171e-03    0
6    1086    rs1086 <NA>  0.320933432 0.08355891  3.8408045 1.226318e-04    0
7    1093    rs1093 <NA>  0.351289229 0.08310134  4.2272393 2.365761e-05    0
8    1128    rs1128 <NA>  0.305191010 0.08295495  3.6789970 2.341531e-04    0
9    1154    rs1154 <NA>  0.401052062 0.08376985  4.7875468 1.688324e-06    1
10   1427    rs1427 <NA>  0.279804648 0.08375709  3.3406682 8.357705e-04    0
11   1480    rs1480 <NA>  0.242626503 0.08289154  2.9270357 3.422096e-03    0
12   1520    rs1520 <NA>  0.334732594 0.08230344  4.0670547 4.761107e-05    0
13   1564    rs1564 <NA>  0.329942824 0.08314567  3.9682501 7.240234e-05    0
14   1625    rs1625 <NA> -0.252559898 0.08288246 -3.0472055 2.309798e-03    0
15   1821    rs1821 <NA>  0.434123295 0.08339721  5.2054897 1.934857e-07    0
16   2374    rs2374 <NA>  0.427348375 0.08264563  5.1708526 2.330283e-07    0
17   2620    rs2620 <NA> -0.246374856 0.08349255 -2.9508604 3.168901e-03    0
18   2679    rs2679 <NA>  0.379357617 0.08376784  4.5286786 5.935370e-06    0
19   2762    rs2762 <NA>  0.276072256 0.08300153  3.3261103 8.806706e-04    0
20   2820    rs2820 <NA>  0.353708052 0.08272462  4.2757288 1.905131e-05    0
21   3129    rs3129 <NA> -0.325669343 0.08202429 -3.9704014 7.175165e-05    0
22   3331    rs3331 <NA>  0.332863189 0.08216810  4.0510026 5.099863e-05    0
23   3427    rs3427 <NA>  0.239361499 0.08265877  2.8957788 3.782189e-03    0
24   3529    rs3529 <NA>  0.376025786 0.08391587  4.4809853 7.429922e-06    0
25   3649    rs3649 <NA>  0.325804803 0.08565356  3.8037510 1.425214e-04    1
26   3917    rs3917 <NA>  0.351840800 0.08308508  4.2347049 2.288520e-05    0
27   4214    rs4214 <NA>  0.240202805 0.08184458  2.9348651 3.336927e-03    0
28   4220    rs4220 <NA> -0.329353358 0.08261007 -3.9868428 6.695836e-05    0
```

```
29   4353    rs4353 <NA>  0.287706688 0.08314509  3.4602968 5.395802e-04    1
30   4782    rs4782 <NA>  0.278561052 0.08268733  3.3688481 7.548301e-04    0
31   4893    rs4893 <NA>  0.332286683 0.08353340  3.9778902 6.952946e-05    0
32   5068    rs5068 <NA>  0.531212466 0.08544827  6.2167724 5.074849e-10    0
33      1       rs1 <NA> -0.073799634 0.08135373 -0.9071450 3.643301e-01    1
34      2       rs2 <NA>  0.096430000 0.08147243  1.1835906 2.365751e-01    1
35      3       rs3 <NA> -0.044018905 0.08123699 -0.5418579 5.879164e-01    1
36   2749    rs2749 <NA>  0.038667356 0.08237923  0.4693823 6.387964e-01    1
37   4269    rs4269 <NA>  0.232585977 0.08358856  2.7825098 5.394024e-03    1
38   <NA>    gender <NA> -0.087571626 0.08217908 -1.0656195 2.865957e-01    2
39   <NA>     group <NA> -0.071535130 0.08158925 -0.8767715 3.806108e-01    2
```

7 Combining Additional Methods: The Kitchen Sink

All of the "additional components" may be mixed and matched. The following example does an initial pre-screen by marginal regression p-values, finds K with the stability criterion, does the core main effects lasso screen, creates pairwise interactions, does a second marginal regression pre-screen, fits the SNP pairs and screened main effects into a second lasso screen, and feeds the resulting list of SNPs into a multivariate regression.

- Function call

```
sc7 = screenClean(chr = chr, pheno = pheno, L = 2000, K_pairs = 100, response = "binomial",
  standardize = FALSE, alpha = 0.05, snp_fix = id_fix, cov_struct = cov_struct)
```

- Program steps and screen output

```
centering and scaling the additional structural covariates....
marginal regression pre-screen....
snp: 500
snp: 1000
snp: 1500
snp: 2000
snp: 2500
snp: 3000
snp: 3500
snp: 4000
snp: 4500
snp: 5000
computing stability statistics to choose K....
fitting lasso for subsample 5 of 50....
fitting lasso for subsample 10 of 50....
fitting lasso for subsample 15 of 50....
fitting lasso for subsample 20 of 50....
fitting lasso for subsample 25 of 50....
fitting lasso for subsample 30 of 50....
fitting lasso for subsample 35 of 50....
fitting lasso for subsample 40 of 50....
fitting lasso for subsample 45 of 50....
fitting lasso for subsample 50 of 50....
K chosen by stability: 73
fitting lasso with glmnet....
generating pairwise interactions....
marginal regression pre-screen....
```

```
snp: 500
snp: 1000
snp: 1500
snp: 2000
fitting lasso with glmnet....
fitting multivariate regression clean....
fitting final regression....
```

- Function values:
  - *snp_prescreen* - a vector of SNPs retained by the initial marginal regression pre-screen.
  - *snp_screen* - a vector of SNPs retained by the main effects lasso screen.
  - *snp_prescreen2* - a data frame listing the SNPs and SNP pairs retained by the second marginal regression pre-screen.
  - *snp_screen2* - a data frame listing the SNPs and SNP pairs retained by the second lasso screen. A subset of *snp_prescreen2*.
  - *snp_clean* - a data frame listing the SNPs and SNP pairs retained by the multivariate regression clean. A subset of *snp_screen2*.
  - *clean* - a data frame with output from the regression of phenotype on all SNPs in *snp_screen2*.
  - *final* - a data frame with output from the regression of phenotype on cleaned SNPs only.

8 Data Description

- *chr* - Simulated allele count data for 1,000 individuals and 5,000 tag SNPs. These data—initially coded as 0, 1, or 2—are centered to mean 0 and standard deviation 1. Also includes 10 blocks of highly correlated simulated SNPs (see *corr_table*).

- *snp_main* - 60 randomly chosen SNPs from the 5,000 in *chr*.

- *snp_pair* - 10 randomly chosen pairs of SNPs from the $\binom{60}{2}$ pairs in *snp_main*.

- *pheno* - a binary phenotype simulated from the 60 SNPs in *snp_main* and the 10 SNP pairs in *snp_pair*.

- *corr_table* - 10 SNPs in *snp_main* were chosen randomly and replicated 10 times with minor perterbations to simulate regions of high linkage disequilibrium on a chromosome. *Corr_table* is a dictionary of these correlated blocks, which are appended to *chr*.

- *cov_struct* - a matrix of two structural covariates, in this case gender and ethnic group. In this data set the structural covariates were generated randomly, so they have no association with the phenotype. These covariates are forced to be in every model fit by Screen & Clean.

- *id_fix* - an index of SNPs that are forced into the lasso and multivariate regression models (but not the marginal regressions).