Sparse Family Indices for breeding value prediction using the SFSI R-package

Marco Lopez-Cruz
lopezcru@msu.edu
Crop, Soil, and Microbial Sciences

Gustavo de los Campos
gustavoc@msu.edu
Epidemiology and Biostatistics

Michigan State University

1 Family index

A **family selection index** is used to predict the breeding value of individuals collecting information from relatives. The borrowing of information relies in the use of a linear mixed model that decomposes phenotypic observations, $\boldsymbol{y} = (y_1, \dots, y_n)'$, as the sum of the population mean (μ), breeding values, $\boldsymbol{u} = (u_1, \dots, u_n)'$, and environmental deviations, $\boldsymbol{e} = (e_1, \dots, e_n)'$, as

$$y_i = \mu + u_i + e_i \tag{1}$$

Both $\boldsymbol{u}$ and $\boldsymbol{e}$ are assumed to be normally distributed with null means and $var(\boldsymbol{u}) = \sigma_u^2\mathbf{G}$, $var(\boldsymbol{e}) = \sigma_e^2\mathbf{I}$, and $cov(\boldsymbol{u}, \boldsymbol{e}') = \mathbf{0}$, where σ_u^2 and σ_e^2 are the common genetic and residual variances, respectively; $\mathbf{G}$ is the **genetic relationship matrix** and $\mathbf{I}$ is an identity matrix. In the family index all the available phenotypic observations (as deviations from the population mean) contribute (to a different extent) to the predicted breeding value of each selection candidate as

$$\mathcal{I}_i = \boldsymbol{\beta}_i'(\boldsymbol{y} - \mu\mathbf{1}) \tag{2}$$

The weights $\boldsymbol{\beta}_i = (\beta_{i1}, \dots, \beta_{in})'$ are chosen such that the mean square error (MSE) of prediction, $\mathbb{E}(u_i - \mathcal{I}_i)^2$, is minimum. This is, solutions $\hat{\boldsymbol{\beta}}_i$ are found by solving:

$$\hat{\boldsymbol{\beta}}_i = \arg\min_{\beta} \frac{1}{2}\mathbb{E}\left[u_i - \boldsymbol{\beta}_i'(\boldsymbol{y} - \mu\mathbf{1})\right]^2$$

Given the assumptions given in model (1), the above problem is equivalent to

$$\hat{\boldsymbol{\beta}}_i = \arg\min_{\beta} \left[-\boldsymbol{\beta}_i'\mathbf{G}_i + \frac{1}{2}\boldsymbol{\beta}_i'(\mathbf{G} + \lambda_0\mathbf{I})\boldsymbol{\beta}_i\right] \tag{3}$$

where $\lambda_0 = \sigma_e^2/\sigma_u^2$ and $\boldsymbol{G}_i$ corresponds to the i^{th} column of the genetic relationship matrix. The solution to this optimization problem is

$$\hat{\boldsymbol{\beta}}_i = (\mathbf{G} + \lambda_0\mathbf{I})^{-1}\mathbf{G}_i \tag{4}$$

Variance components σ_e^2, σ_u^2, and $\mathbf{G}$ are assumed to be accurately estimated. With μ known, this family index corresponds to the selection index developed by Smith (Smith, 1936) and Hazel (Hazel, 1943). When the population mean is replaced by its least square estimate $\hat{\mu}$, the resulting index is the **kinship-based BLUP** found by Henderson (Henderson, 1963). Therefore, there is an equivalence between the family index and the kinship-based BLUP when the mean is null.

2 Sparse family index

In the **sparse family index** (SFI), some of the regression coefficients become zero and the most predictive ones are given a non-zero value. This sparsity feature is achieved by adding a penalization factor on the coefficients $\boldsymbol{\beta}_i$ in the optimization problem in (3), as follows:

$$\hat{\boldsymbol{\beta}}_i = \arg\min_{\beta} \left[-\boldsymbol{\beta}_i'\mathbf{G}_i + \frac{1}{2}\boldsymbol{\beta}_i'(\mathbf{G} + \lambda_0\mathbf{I})\boldsymbol{\beta}_i + \lambda \cdot J(\boldsymbol{\beta}_i) \right] \tag{5}$$

where λ and $J(\boldsymbol{\beta}_i)$ are, respectively, a penalty parameter and a penalty function. We considered a penalty function as

$$J(\boldsymbol{\beta}_i) = \frac{1}{2}(1-\alpha)\sum_{j=1}^{n}\beta_{ij}^2 + \alpha\sum_{j=1}^{n}|\beta_{ij}|$$

where α is a weighting factor. This penalty is used in an **elastic-net-type** regression (Zou & Hastie, 2005) that combines the shrinkage-inducing feature of a **ridge-regression** (Hoerl & Kennard, 1970), that uses the L2-norm: $J(\boldsymbol{\beta}_i) = \frac{1}{2}\sum_{j=1}^{n}\beta_{ij}^2$, and the variable selection feature of a **LASSO** regression (Tibshirani, 1996), that uses the L1-norm: $J(\boldsymbol{\beta}_i) = \sum_{j=1}^{n}|\beta_{ij}|$. The ridge-regression and LASSO types are special cases when $\alpha = 0$ and $\alpha = 1$, respectively. With no penalization ($\lambda = 0$), the solution for (5) is equivalent to that of the (non-sparse) kinship-based BLUP in (4).

A closed-form solution for the regression coefficients can be found only when $\alpha = 0$ (Hastie et al., 2009) (i.e., a ridge-regression-type SFI). In this case, the solution is $\hat{\boldsymbol{\beta}}_i = [\mathbf{G} + (\lambda_0 + \lambda)\mathbf{I}]^{-1}\,\mathbf{G}_i$; however, for $0 < \alpha \leq 1$ solutions are obtained using iterative algorithms such as **least angle regression** (Efron et al., 2004) or **coordinate descent** (Friedman et al., 2007) for different values of the parameters α and λ. These combinations of the values of α and λ will result in different SFIs from which an optimal index can be obtained such as the prediction accuracy is maximum.

3 Accuracy of the index

The accuracy of the index is defined as the correlation between the index ($\mathcal{I}_i$) and the breeding values (u_i) and can be derived from path coefficients as

$$cor(\mathcal{I}_i, u_i) = cor(\mathcal{I}_i, y_i)/h$$

where $h = cor(y_i, u_i)$ is the correlation between phenotypic and breeding values, and it is equivalent to the square root of the heritability of the trait, defined as

$$h^2 = \frac{\sigma_u^2}{\sigma_u^2 + \sigma_e^2}$$

4 Cross validation

Two types of cross-validation (CV) will be used: (i) training-testing (TRN-TST) partitions to avoid bias in the estimation in the accuracy of the index and (ii) k-folds cross-validation to obtain a point estimate of the penalization parameter λ.

- **Training-testing partitions**. The accuracy of the index is evaluated as follows: (i) data is split into training and testing sets, (ii) a grid of different values of λ (in equation (5)) within the range of possible values is obtained, (iii) regression coefficients are obtained for each individual in the testing set using information from training data, for each value of λ, (iv) an index is calculated for each value of λ for all individuals in the testing set as a linear combination of the observed values in training set (see equation (2)), and (v) the prediction accuracy is calculated for all the indices fitted in the testing set. This TRN-TST procedure is repeated to calculate standard deviations.

- ***k*-folds CV**. A value of λ is estimated using only data from training set as follows: (i) training data is further partitioned into k subsets (called *folds*), (ii) an SFI is calculated within each fold using the remaining $k-1$ folds to train the model for a grid of different values of λ within the range of possible values, (iii) the optimal λ is chosen such that the averaged (across all k folds) SFI has the biggest correlation between observed and predicted values, and (iv) this optimal penalization is used along with the whole training data to calculate the optimal SFI for the testing data. An optimal value for λ can be also chosen in step (iii) such that the cross-validated MSE is minimum. The k-folds partition can be repeated many times to obtain an optimal λ averaging across folds and partitions.

5 Experimental data

The data set consists of 58,798 wheat lines from CIMMYT's Global Wheat Program. Lines were evaluated at the main experimental station in Ciudad Obregon, Mexico, under several environmental conditions representing a combination of planting system (bed vs flat, the later referred to as melgas), number of irrigations (2, 5 irrigations or drip irrigation), and sowing date (optimum, late or early planting); during five cycles between 2009 and 2013. The first cycle (09-10) was sown in late 2009 and harvested in 2010, the second cycle (10-11) was planted in late 2010 and harvested early 2011; and so on. Several trials were established in an α-lattice design with three replicates into (incomplete) blocks. Total grain yield ($ton\ ha^{-1}$) after maturity was collected at each plot. Mixed models including effects of the mean (fixed), trial (random), replicate within trial (random), incomplete block within trial and replicate (random), and genotype (random) were fitted within environment. Grain yield records are reported as adjusted means obtained from removing effects from trial, replicate and block. Only a subset of 29,484 genotypes were genotyped using GBS (Genotyping-by-sequencing) technology followed by SNP calling for 42,706 markers. Quality control was applied by removing SNP markers with minor allele frequency larger lower than 5% and with more than 80% of missing values. The leftover 9,045 SNP markers that passed quality control were imputed using observed data. Finally, only phenotypic information genotypes with marker information within environment were kept for analyses.

6 Implementation

6.1 Data preparation

Both phenotypic and marker data for the large dataset can be downloaded from CIMMYT's repository at `http://genomics.cimmyt.org/wheat_50k/PG` (accessed January, 28, 2020). File `G80_42706_29489_correctedgid.RData` contains the marker information and `Blups_condition_group_random.csv` contains the adjusted phenotypes for all environmental conditions. Box 1 below shows how to prepare data for a single environment which will be used in latter analyses.

Box 1. Data preparation

```
rm(list = ls())
setwd("/mnt/research/quantgen/projects/PFI/pipeline")
site <-  "http://genomics.cimmyt.org/wheat_50k/PG"
filename1 <- "Blups_condition_group_random.csv"
filename2 <- "G80_42706_29489_correctedgid.RData"

# Read files
Y <- read.table(paste0(site,"/",filename1),row.names=1,sep=",",header=T)
load(url(paste0(site,"/",filename2)))

# Select an environment to work with
trait <- c("B2I_OBR","B5I_OBR","DRB_OBR","EHT_OBR","LHT_OBR","MEL_OBR")[4]

# Match genotypes in both files
Y <- Y[!is.na(Y[,trait]),trait,drop=FALSE]
common <- intersect(rownames(X),rownames(Y))
X <- X[common,]
Y <- Y[common, , drop=FALSE]

# Calculate G matrix
X <- scale(X)
G <- tcrossprod(X)/ncol(X)

dir.create("data", recursive=TRUE)
save(G,file="data/G.RData")
save(Y,file="data/PHENO.RData")
```

In the example shown above, the preparation was made for the environment *EHT_OBR* containing $n = 2,040$ genotypes.

6.2 Heritability and variance components

Heritability can be calculated from the whole data by fitting the model (1). Code below illustrates how to fit this model using the function `solveMixed` from package `SFSI`.

Box 2. Variance components calculation

```
library(SFSI)
setwd("/mnt/research/quantgen/projects/PFI/pipeline")

# Load data
load("data/G.RData")
load("data/PHENO.RData")
y <- as.vector(scale(Y[,1]))

# Fit model
fm0 <- solveMixed(y,K=G)
fm0$varU
fm0$varE
fm0$h2

dir.create("output", recursive=TRUE)
save(fm0,file="output/varComps.RData")
```

6.3 Accuracy of the SFI along the penalization parameter

Box 3a below illustrates how to fit the SFI using a TRN-TST procedure where data is split to contain 70% of the data in the training set and the remaining 30% in the testing set. Likewise, the kinship-based BLUP using genomic data model (G-BLUP) is fitted for comparison. The number of TRN-TST partitions can be set by the variable `nPart`.

In this example containing $n = 2,040$ individuals, $n_{TRN} = 1,428$ are assigned to the training set to predict the remaining $n_{TST} = 612$ individuals in the testing set.

Box 3a. Accuracy of prediction

```
load("output/varComps.RData")

# Number of elements in TST set
nTST <- ceiling(0.3*length(y))

# Create a vector of seeds for randomization to create folds
nPart <- 5
seeds <- round(seq(1E3, .Machine$integer.max, length = nPart))
partitions <- matrix(NA,nrow=nTST,ncol=nPart)

# Objects to store results
accGBLUP <- accSFI <- lambda <- df <- mu <- h2 <- c()

# Fit model
for(k in seq_along(seeds))
{
    cat("  partition = ",k,"\n")
    set.seed(seeds[k])
    tst <- sample(1:length(y),nTST,replace=FALSE)
    partitions[,k] <- tst
    trn <- (1:length(y))[-tst]
    yNA <- y
    yNA[tst] <- NA

    fm1 <- solveMixed(yNA,K=G)
    accGBLUP[k] <- cor(fm1$u[tst],y[tst])/sqrt(fm0$h2)
    mu[k] <- fm1$b
    h2[k] <- fm1$h2

    fm2 <- SFI(y,K=G,b=mu[k],h2=h2[k],trn=trn,tst=tst,mc.cores=10)
    accSFI <- rbind(accSFI,summary(fm2)$accuracy/sqrt(fm0$h2))
    lambda <- rbind(lambda,summary(fm2)$lambda)
    df <- rbind(df,summary(fm2)$df)
}
save(mu,h2,accGBLUP,accSFI,lambda,df,partitions,file="output/results_accuracy.RData")
```

After performing the above analysis, code in Box 3b below can be used to create a plot depicting the evolution of accuracy over values of penalization. Large values of λ (corresponding to small values of $-log(\lambda)$) allow small number of individuals in training data to contribute (i.e., their corresponding weights are nonzero, $\beta_{ij} \neq 0$) to each individual family index in testing data, $\mathcal{I}_i = \boldsymbol{\beta}'_i(\boldsymbol{y}_{TRN} - \mu\mathbf{1})$, $i = 1, ..., n_{TST}$, resulting in a more sparse index. As the parameter λ is relaxed toward zero (i.e., $-log(\lambda)$ is increasing) individuals from training set pass from having a null regression coefficient to have a non-zero coefficient that contributes to the index. The accuracy of the G-BLUP is also shown at the rightmost side in the same plot as it is equivalent to the SFI when $\lambda = 0$.

Box 3b. Accuracy of prediction (cont ...)

```
library(ggplot2)
load("output/results_accuracy.RData")

dat1 <- data.frame("SFI",colMeans(df),colMeans(lambda),colMeans(accSFI),apply(accSFI,2,sd))
dat2 <- data.frame("GBLUP",max(dat1[,2]),min(dat1[,3]),mean(accGBLUP),sd(accGBLUP))
names(dat1) <- names(dat2) <- c("Model","df","lambda","Accuracy","sd")
dat1$se <-  qnorm(0.975)*dat1$sd/sqrt(length(accGBLUP))
dat2$se <-  qnorm(0.975)*dat2$sd/sqrt(length(accGBLUP))

datOpt <- dat1[which.max(dat1$Accuracy),]
ggplot(dat1[dat1$df>3,],aes(-log(lambda),Accuracy,color=Model)) +
   geom_hline(yintercept=mean(accGBLUP), linetype="dashed") +
   geom_line(size=1.1) + theme_bw() +
   geom_errorbar(data=dat2,aes(ymin=Accuracy-se, ymax=Accuracy+se),color=1) +
   geom_errorbar(data=datOpt,aes(ymin=Accuracy-se, ymax=Accuracy+se),color=1) +
   geom_point(data=datOpt,aes(-log(lambda),Accuracy),size=2.5) +
   geom_point(data=dat2,aes(-log(lambda),Accuracy),size=2.5)
```

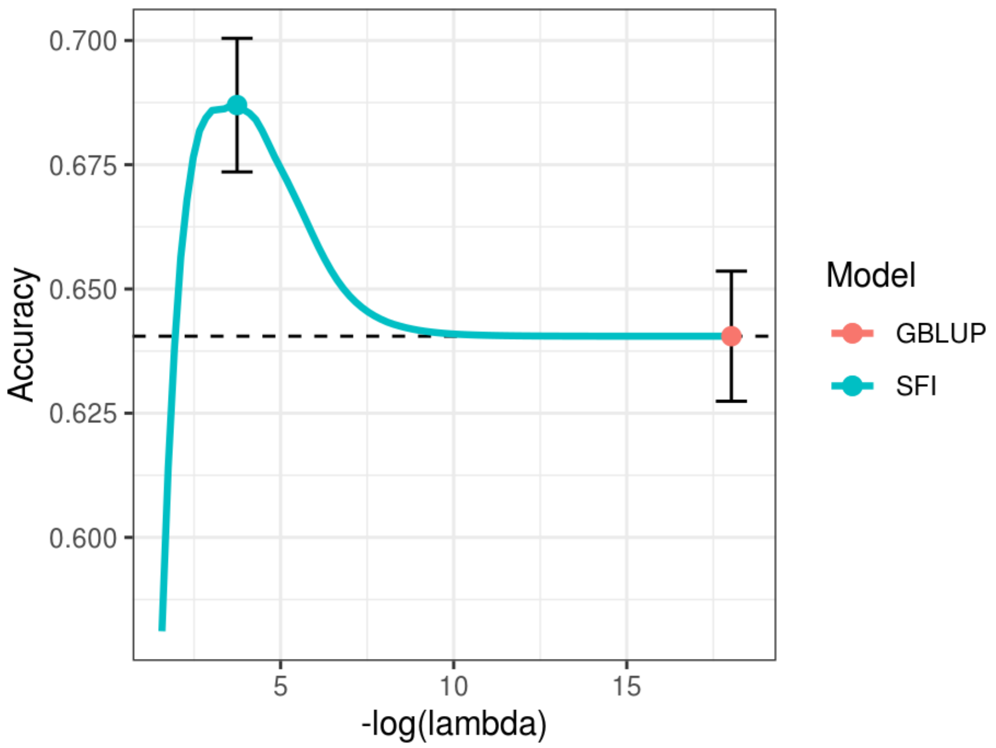

Figure 1: Prediction accuracy (average across 50 TRN-TST partitions) of the SFI versus the penalization parameter λ (logarithm scale). Vertical bars represent the 95% confidence interval for the SFI with highest accuracy and for the G-BLUP model

6.4 Estimation of the penalization parameter

Code in Box 4a below can be used to implement k-folds CV to get an estimate of the parameter λ. The CV is performed using `SFI_CV` within each TRN-TST partition previously created and saved in object `partitions`. The number of folds can be set by the variable `nFolds`.

The choosing of λ is done by the two criteria: maximizing the correlation between observed and predicted values, and minimizing the MSE, both in the training set.

Box 4a. Accuracy of prediction

```
load("output/results_accuracy.RData")

# Objects to store results
accSFI_CV  <- lambdaCV <- matrix(NA,ncol=2,nrow=ncol(partitions))
dfCV1 <- dfCV2 <- c()

# Fit model
for(k in 1:ncol(partitions))
{
    tst <- partitions[,k]
    trn <- (1:length(y))[-tst]

    # Cross-validation in training set
    fm1 <- SFI_CV(y,K=G,trn.CV=trn,mc.cores=10,nFolds=5,nCV=1)
    lambdaCV[k,1] <- summary(fm1)$optCOR["mean","lambda"]
    lambdaCV[k,2] <- summary(fm1)$optMSE["mean","lambda"]

    # Fit SFI with lambda estimated using 'correlation' criteria
    fm2 <- SFI(y,K=G,b=mu[k],h2=h2[k],trn=trn,tst=tst,lambda=lambdaCV[k,1])
    accSFI_CV[k,1] <- summary(fm2)$accuracy/sqrt(fm0$h2)
    dfCV1 <- cbind(dfCV1,drop(fm2$df))

    # Fit SFI with lambda estimated using 'MSE' criteria
    fm2 <- SFI(y,K=G,b=mu[k],h2=h2[k],trn=trn,tst=tst,lambda=lambdaCV[k,2])
    accSFI_CV[k,2] <- summary(fm2)$accuracy/sqrt(fm0$h2)
    dfCV2 <- cbind(dfCV2,drop(fm2$df))
}
save(accSFI_CV,lambdaCV,dfCV1,dfCV2,file="output/results_accuracyCV.RData")
```

After running the above analysis, code in Box 4b below can be used to create a plot where the accuracy of the SFI obtained by both criteria are compared with that of the non-sparse SFI (G-BLUP).

Box 4b. Estimation of an optimal penalization (cont ...)

```
load("output/results_accuracy.RData")
load("output/results_accuracyCV.RData")

dat1 <- data.frame(mod="SFI_COR",Accuracy=mean(accSFI_CV[,1]),sd=sd(accSFI_CV[,1]))
dat2 <- data.frame(mod="SFI_MSE",Accuracy=mean(accSFI_CV[,2]),sd=sd(accSFI_CV[,2]))
dat3 <- data.frame(mod="GBLUP",Accuracy=mean(accGBLUP),sd=sd(accGBLUP))
dat <- rbind(dat1,dat2,dat3)
dat$se <-  qnorm(0.975)*dat$sd/sqrt(length(accGBLUP))

ggplot(dat,aes(mod,Accuracy)) +  theme_bw() + labs(x="") +
   geom_bar(stat="identity",width=0.5,fill="orange") +
   geom_errorbar(aes(ymin=Accuracy-se, ymax=Accuracy+se), width=0.2) +
   geom_text(aes(label=sprintf("%.3f", Accuracy),y=(Accuracy-se)*0.8))
```

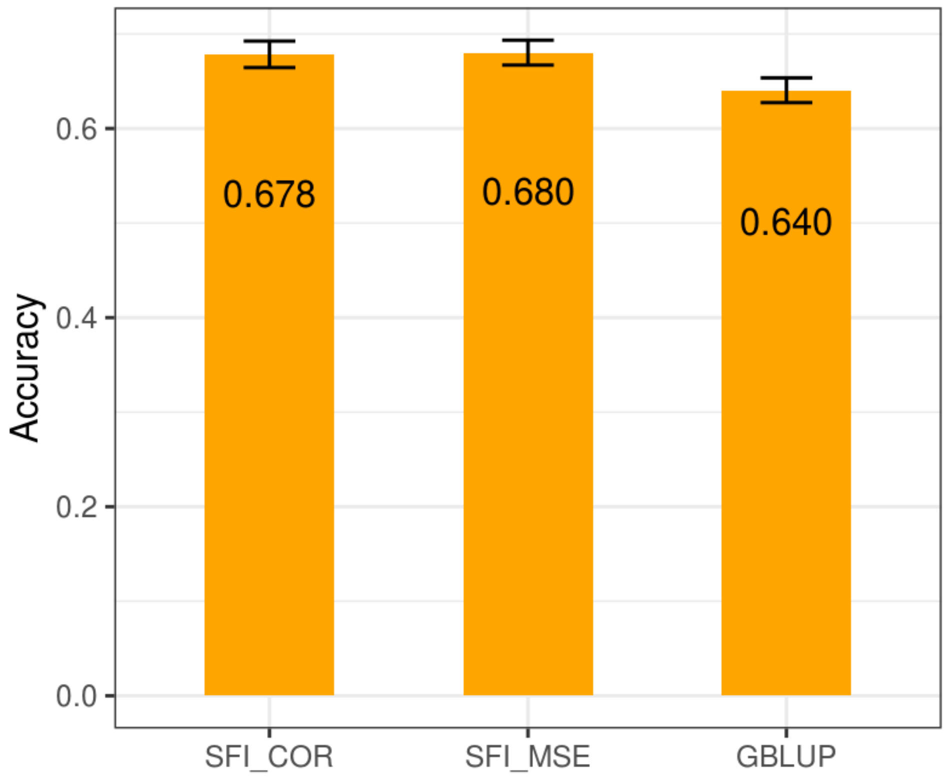

Figure 2: Prediction accuracy (average across 50 TRN-TST partitions) for the cross-validated SFI (using both correlation and MSE criteria) and for the G-BLUP model

6.5 Sparsity of the index

The value of λ define the sparsity level of the SFI, i.e., the number of predictors with non-zero coefficient in the index . Although performing similar in accuracy, each cross-validation criteria yielded different values of λ, thus one criteria result in a more sparse index than the other. Code in Box 5 below can be used to create a plot showing the distribution of the number of predictors supporting the index $\mathcal{I}_i$ of each of the n_{TST} individuals in the testing set across all partitions.

Box 5. Sparsity of the optimal index

```
load("output/results_accuracyCV.RData")

df <- c(as.vector(dfCV1),as.vector(dfCV2))
criteria <- c(rep("Correlation",length(dfCV1)),rep("MSE",length(dfCV2)))
dat <- data.frame(criteria,df)

bw <- round(diff(range(df))/40)

ggplot(data=dat, aes(df,fill=criteria)) +
    geom_histogram(aes(y=stat(count)/length(dfCV1)),color="gray45",alpha=0.5,
         binwidth=bw,position="identity") +
    theme_bw() + labs(x = "Number of active predictors",y="Frequency")
```

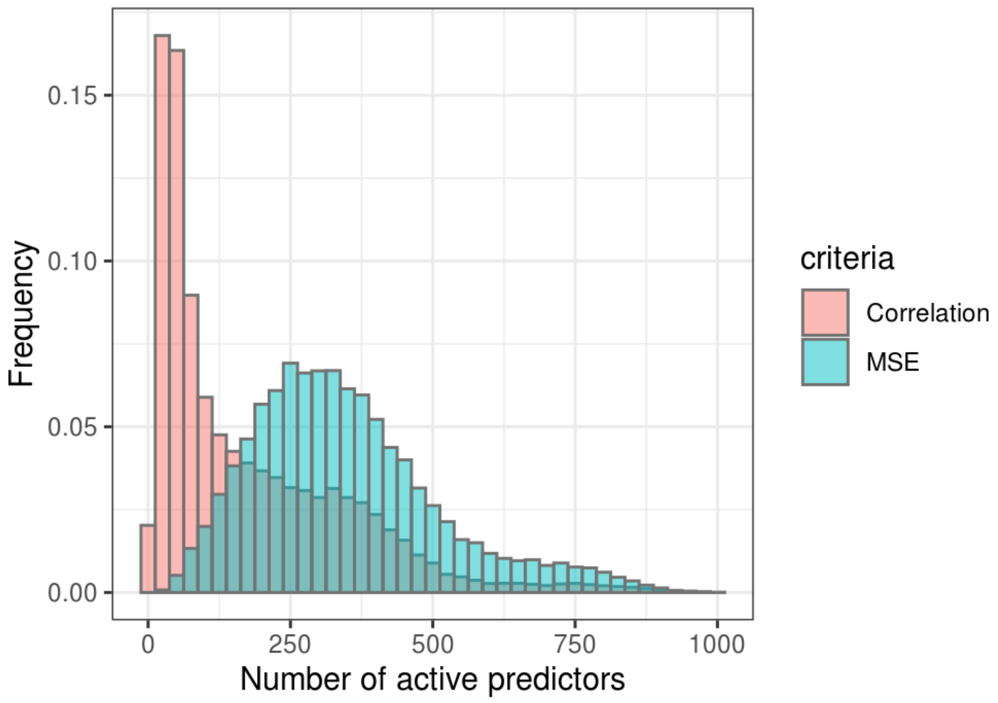

Figure 3: Distribution of the number of active predictor in the SFI (across 50 TRN-TST partitions) for each criteria: (1) maximizing correlation between predicted and observed values and (2) minimizing the MSE

6.6 Individualized training sets

Estimating the regression coefficients independently for each individual in the testing set (equation (5)) yields subset of predictor that are specific each individual from testing set. Code below in Box 6 can be run to create a network plot showing for each individual being predicted, the subset of predictors supporting the SFI. This plot can be made through the function `plotNet` included in the `SFSI` package.

Box 6. Individualized training sets

```
load("output/results_accuracy.RData")
load("output/results_accuracyCV.RData")

part <- which.min(apply(dfCV1,2,mean))
tst <- partitions[,part]
trn <- (1:length(y))[-tst]

# Fit SFI with lambda estimated using 'correlation' criteria
fm1 <- SFI(y,K=G,trn=trn,tst=tst,lambda=lambdaCV[part,1])

plotNet(fm1,K=G,tst=fm1$tst[1:8],curve=TRUE)
```

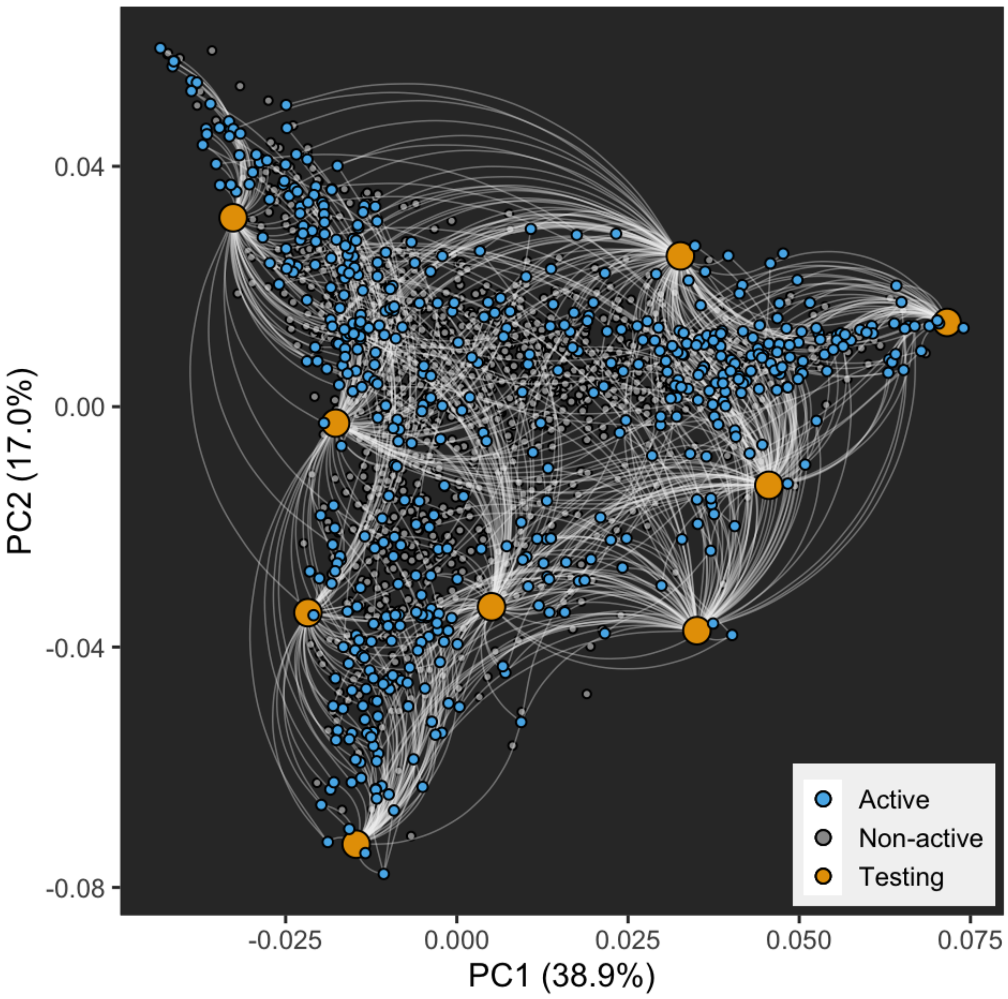

Figure 4: Top two principal components of the genomic relationship matrix, G. Each point represent individuals separated into training ($n_{TRN} = 1,428$) and testing ($n_{TST} = 612$) sets. Orange points represent a sample of individuals from testing set which are connected by lines to individuals from training set whose regression coefficients in the optimum SFI are non-zero (active). Individuals in training set with a null coefficient do not contribute to the index (non-active)