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

Contents

1 Family index

A **family selection index** is used to predict the breeding value of a **target trait** (y_i) of individuals collecting information from their relatives. The borrowing of information relies in the use of a linear mixed model that decomposes the target trait's 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

$$y_i = \mu + u_i + e_i \qquad (1)$$

Both $\boldsymbol{u}$ and $\boldsymbol{e}$ are assumed to be normally distributed with null means and $\text{var}(\boldsymbol{u}) = \sigma_u^2\mathbf{G}$, $\text{var}(\boldsymbol{e}) = \sigma_e^2\mathbf{I}$, and $\text{cov}(\boldsymbol{u}, \boldsymbol{e}') = \mathbf{0}$, 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 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 (1936) and 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 (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 is a penalty parameter and $J(\boldsymbol{\beta}_i)$ is a penalty function. Commonly used penalties include the L2 ($||\boldsymbol{\beta}_i||_2^2 = \sum_{j=1}^n \beta_{ij}^2$) and L1 ($||\boldsymbol{\beta}_i||_1 = \sum_{j=1}^n |\beta_{ij}|$) norms (Fu, 1998) either alone or in a combination of both. 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 and Hastie, 2005) that combines the shrinkage-inducing feature of a **ridge-regression** (Hoerl and Kennard, 1970), that uses the L2-norm alone, and the variable selection feature of a **LASSO** regression (Tibshirani, 1996), that uses the L1-norm alone. 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., an L2-penalized 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 (Dekkers, 2007) as

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

where $h = \text{cor}(y_i, u_i)$ is the correlation between phenotypic and breeding values, and it is equivalent to the square root of the heritability of the trait.

4 Genomic relationship matrix and heritability

A genomic relationship matrix can be calculated using marker information as in VanRaden (2008) using marker information, $\mathbf{M} = \{m_{ij}\}$, as $\mathbf{G} = \mathbf{XX}'/p$, where p is the number of markers and $\mathbf{X} = \{(m_{ij} - \bar{m}_j)/\text{sd}_{m_j}\}$ is the matrix of centered and standardized markers obtained by subtracting from each marker entry, m_{ij}, the mean of each column, $\bar{m}_j$, and scaling it by the standard deviation of the column, sd_{m_j}.

The estimated $\mathbf{G}$ matrix can be used to fit to the trait phenotypes the linear model in (1). Then the heritability is estimated from the estimated genetic and residual variances, σ^2_u and σ^2_e, as

$$h^2 = \frac{\sigma^2_u}{\sigma^2_u + \sigma^2_e} \tag{6}$$

Parameter λ_0 in (3) and (5) can be expressed in terms of the heritability as

$$\lambda_0 = \frac{1 - h^2}{h^2} \tag{7}$$

5 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.

5.1 Training-testing partitions

The accuracy of the index is evaluated as follows: (a) data is split into training and testing sets where matrix $\mathbf{G}$ is separated into sub-matrices $\mathbf{G}_{TRN,TRN}$ and $\mathbf{G}_{TRN,TST}$, (b) a grid of different values of λ (in equation (5)) within the range of possible values is obtained, (c) regression coefficients are obtained for each individual i in the testing set using $\mathbf{G}_{TRN,TST(i)}$ and information from training data ($\mathbf{G}_{TRN,TRN}$), for each value of λ, (d) 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 (e) the prediction accuracy is calculated for all the indices fitted in the testing set. This TRN-TST procedure is repeated to calculate standard deviations.

5.2 k-folds CV

A value of λ is estimated using only data from training set as follows: (a) training data is further partitioned into k subsets (called *folds*), (b) 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, (c) the optimal λ is chosen such that the averaged (across all k folds) SFI has the biggest correlation between observed and predicted values, and (d) 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 (c) 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.

6 Experimental data

The data set consists of 58,798 wheat lines from the Global Wheat Program of CIMMYT (International Maize and Wheat Improvement Center). Lines were evaluated at the 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 or drip irrigation), and sowing date (optimum, late or early planting); during five cycles between 2009 and 2013. Several trials were established in an α-lattice design with three replicates into incomplete blocks.

Total grain yield (GY, 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 29,489 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. Table below shows the number of available observations and average grain yield, by environmental condition.

Planting conditions		**Number of**			**Average (SD)**
Date	**System**	**irrigations**	**Name**	**n**	**GY (ton ha^{-1})**
Optimum	Bed	2	B2I_OBR	3,732	4.53 (0.261)
Optimum	Bed	5	B5I_OBR	29,473	7.12 (0.372)
Optimum	Flat	5	MEL_OBR	4,403	5.76 (0.304)
Late	Bed	5	LTH_OBR	4,404	3.83 (0.375)
Optimum	Bed	Drip	DRB_OBR	3,763	2.74 (0.275)
Early	Bed	5	EHT_OBR	2,040	6.16 (0.525)

7 Implementation

The above described data will be used to implement sparse family indices methodology for grain yield. Analyses will be implemented in `R` software (R Core Team, 2019) using the `SFSI` R-package. The package can be installed from the `GitHub` (development version) repository at `https://github.com/MarcooLopez/SFSI`. Regression coefficients (in (5)) are separately calculated for the i^{th} individual ($i = 1, ..., n_{TST}$) using $\mathbf{G}_{TRN,TST(i)}$ and $\mathbf{G}_{TRN,TRN} + \lambda_0\mathbf{I}$ as inputs as in Lopez-Cruz et al. (2019) for a penalized selection index.

7.1 Data preparation

Both phenotypic and marker data can be downloaded from CIMMYT's repository at the site `http://genomics.cimmyt.org/wheat_50k/PG`. Marker information for $42,706$ SNPs is contained in the file `G80_42706_29489_correctedgid.RData`. Adjusted phenotypes for all environmental conditions are found in the csv file `Blups_condition_group_random.csv`. **Box 1** below shows how to prepare data only for the environment *EHT_OBR* containing $n = 2,040$ genotypes, and hereinafter analyses will be with reference to this environment.

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 <- as.vector(scale(Y[common, ]))

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

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

7.2 Heritability and variance components

Heritability can be calculated from the whole data by fitting the model (1). Code in **Box 2** below illustrates how to fit this model using the `solveMixed` function from `SFSI` package.

Box 2. Variance components

```
library(SFSI)

load("data/geno_pheno.RData")   # Load data

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

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

7.3 Training-testing partitions

Code in **Box 3** below illustrates how to create partitions splitting data into TRN and TST sets. In this example, 70% of the data ($n_{TRN} = 1,428$ observations) will be randomly assigned to the training set and the remaining 30% ($n_{TST} = 612$) to the testing set.

The output will be a matrix with `nPart` columns and in rows containing indices indicating the 612 observations that are assigned to testing sets. The object will be saved in the file `partitions.RData` and will be used for later analyses.

Box 3. Create testing set partitions

```
nPart <- 5        # Number of partitions
seeds <- round(seq(1E3, .Machine$integer.max, length = nPart))
load("data/geno_pheno.RData")    # Load data
nTST <- ceiling(0.3*length(y))   # Number of elements in TST set

partitions <- matrix(NA,nrow=nTST,ncol=nPart)     # Object to store partitions

for(k in 1:nPart)
{   set.seed(seeds[k])
    partitions[,k] <- sample(1:length(y),nTST,replace=FALSE)
}
save(partitions,file="output/partitions.RData")
```

7.4 Fitting the sparse family index

Code in **Box 4a** below illustrates how to fit the SFI using the partitions above created. The SFI is calculated usng the `SFI` function for `nLambda=100` values of λ. The G-BLUP model is fitted for comparison using the `solveMixed` function.

Estimates $\hat{\mu}$ and $\hat{h}^2$ obtained from the G-BLUP model are passed to the `SFI` function to avoid being computed again thus saving time. The accuracy of both G-BLUP and SFI models is reported and will be stored in object `accSFI` and saved in the file `results_accuracy.RData`.

Box 4a. Accuracy of prediction

```
load("data/geno_pheno.RData")   # Load data
load("output/varComps.RData");  load("output/partitions.RData")

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

for(k in 1:ncol(partitions))
{   cat("  partition = ",k,"\n")
    tst <- partitions[,k]
    trn <- (1:length(y))[-tst]
    yNA <- y;   yNA[tst] <- NA

    # G-BLUP model
    fm1 <- solveMixed(yNA,K=G)
    mu[k] <- fm1$b
    h2[k] <- fm1$h2

    # Sparse FI
    fm2 <- SFI(y,K=G,b=mu[k],h2=h2[k],trn=trn,tst=tst,mc.cores=10,nLambda=100)
    fm3 <- summary(fm2)

    accuracy <- c(cor(fm1$u[tst],y[tst]),fm3$accuracy)/sqrt(fm0$h2)
    lambda <- c(min(fm3$lambda),fm3$lambda)
    df <- c(max(fm3$df),fm3$df)
    namesSFI <- c("GBLUP",paste0("SFI_",1:length(fm3$lambda)))
    accSFI <- rbind(accSFI,data.frame(rep=k,SFI=namesSFI,accuracy,lambda,df))
}
save(mu,h2,accSFI,file="output/results_accuracy.RData")
```

7.4.1 Accuracy of the SFI along the penalization parameter

After performing the above analysis, code in **Box 4b** 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, 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 4b. Effect of penalization on the accuracy

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

dat <- data.frame(do.call(rbind,lapply(split(accSFI,accSFI$SFI),function(x)
 c(apply(x[,-c(1:2)],2,mean),se=qnorm(0.975)*sd(x$accuracy)/sqrt(nrow(x)))  )))
dat$Model <- unlist(lapply(strsplit(rownames(dat),"_"),function(x)x[1]))

dat2 <- rbind(dat["GBLUP",],dat[which.max(dat$accuracy),] )
 ggplot(dat[dat$df>3,],aes(-log(lambda),accuracy)) +
   geom_hline(yintercept=dat["GBLUP",]$accuracy, linetype="dashed") +
   geom_line(aes(color=Model),size=1.1) + theme_bw() +
   geom_errorbar(data=dat2,aes(ymin=accuracy-se,ymax=accuracy+se),width=0.35) +
   geom_point(data=dat2,aes(color=Model),size=2.5)
```

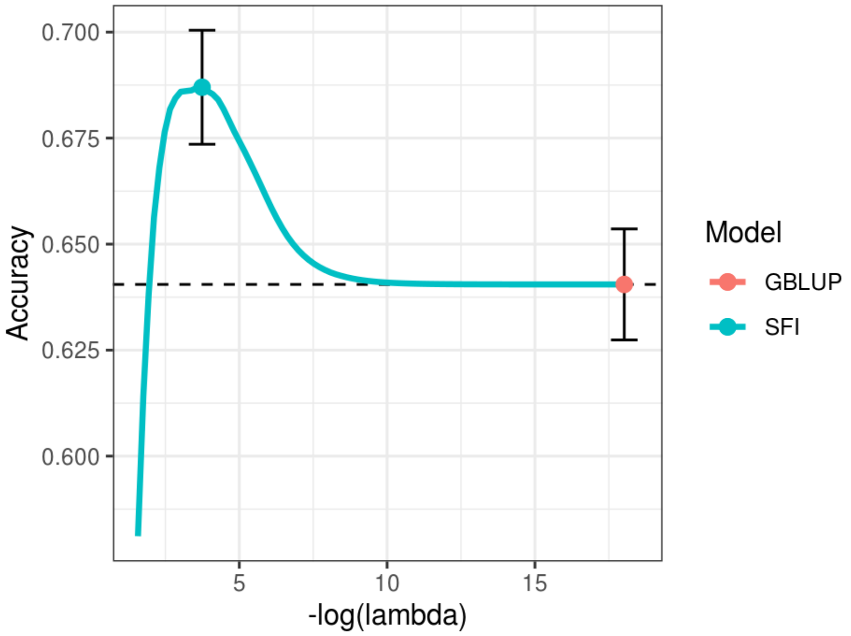

Figure 1: Prediction accuracy (average across 50 TRN-TST partitions) of the SFI versus the penalization parameter λ (logarithm scale). Vertical bars represent the 95% confidence interval for the SFI with highest accuracy and for the G-BLUP model

7.5 Estimation of the penalization parameter

Code in **Box 5** below can be used to implement k-folds CV to get an 'optimal' value of λ and then used it to fit the SFI for testing data. The CV is performed using the `SFI_CV` function within each TRN-TST partition. A single CV repetition of 5-folds will be performed but this can be set by changing `nCV` and `nFolds` parameters. The choosing of λ is done by the two criteria: (a) maximizing the correlation between observed and predicted values, and (b) minimizing the MSE, both in the training set.

Box 5. Optimal value of lambda via CV

```
load("data/geno_pheno.RData");      load("output/varComps.RData")
load("output/partitions.RData");    load("output/results_accuracy.RData")

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

for(k in 1:ncol(partitions))
{   cat("  partition = ",k,"\n")
    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 both criteria
    fm2 <- SFI(y,K=G,b=mu[k],h2=h2[k],trn=trn,tst=tst,lambda=lambdaCV[k,1])
    fm3 <- SFI(y,K=G,b=mu[k],h2=h2[k],trn=trn,tst=tst,lambda=lambdaCV[k,2])

    acc <- c(SFI_COR=summary(fm2)$accuracy,SFI_MSE=summary(fm2)$accuracy)
    accSFI_CV <- rbind(accSFI_CV,acc/sqrt(fm0$h2))
    dfCOR <- cbind(dfCOR,fm2$df);     dfMSE <- cbind(dfMSE,fm3$df)
}
save(accSFI_CV,lambdaCV,dfCOR,dfMSE,file="output/results_accuracyCV.RData")
```

7.5.1 Optimal sparse family index vs G-BLUP

After running the above analysis, code in **Box 6** below can be used to create a bar plot comparing the accuracy of the optimal SFI obtained by both criteria with that of the G-BLUP.

Box 6. Accuracy comparison

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

dat <- data.frame(GBLUP=accSFI[accSFI$SFI=="GBLUP",]$acc,accSFI_CV)
dat <- data.frame(Model=names(dat),Accuracy=apply(dat,2,mean),sd=apply(dat,2,sd))
dat$se <-  qnorm(0.975)*dat$sd/sqrt(nrow(accSFI_CV))

ggplot(dat,aes(Model,Accuracy)) + 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=min(dat$Accuracy)*0.8) + theme_bw()
```

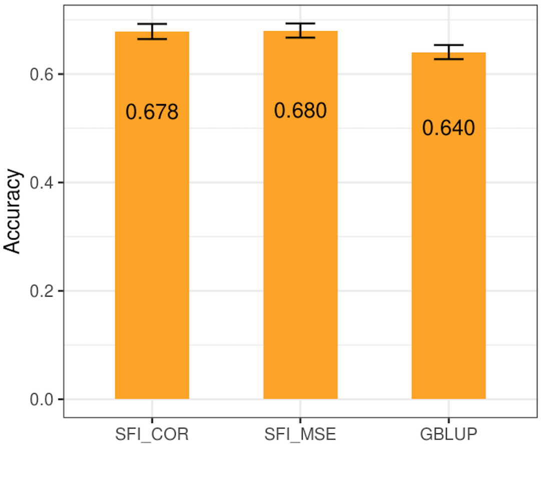

Figure 2: Prediction accuracy (average across 50 TRN-TST partitions) for the optimal SFI (using both correlation and MSE criteria) and for the G-BLUP model. Vertical bars represent the 95% confidence interval for the mean

7.6 Sparsity of the index

Each cross-validation criteria yielded different values of λ, thus one criteria result in a more sparse index than the other. Code in **Box 7** below creates a plot showing the distribution of the number of predictors supporting the index $\mathcal{I}_i$, $i = 1, ..., n_{TST}$, across all partitions.

Box 7. Sparsity of the optimal index

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

dat <- cbind(SFI_COR=as.vector(dfCOR),SFI_MSE=as.vector(dfMSE))
dat <- data.frame(reshape::melt(dat))

bw <- round(diff(range(dat$value))/40)
ggplot(data=dat, aes(value,stat(count)/length(dfCOR),fill=X2)) +  theme_bw() +
    geom_histogram(color="gray45",alpha=0.5,binwidth=bw,position="identity") +
    labs(x = "Number of active predictors",y="Frequency",fill="")
```

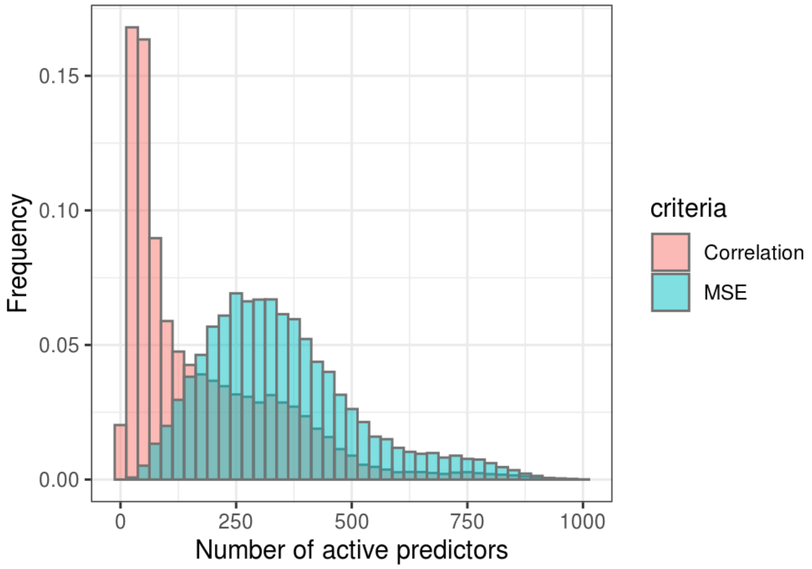

Figure 3: Distribution of the number of active predictor in the SFI (across 50 partitions) for each criteria: (1) maximizing correlation between predicted and observed values and (2) minimizing the MSE

7.7 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 8** can be run to create a network plot showing (for a single TRN-TST partition) 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 8. Individualized training sets

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

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

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

plotNet(fm,K=G,tst=fm$tst[1:8],curve=TRUE,title=NULL)    # Only for 8 testing individuals
```

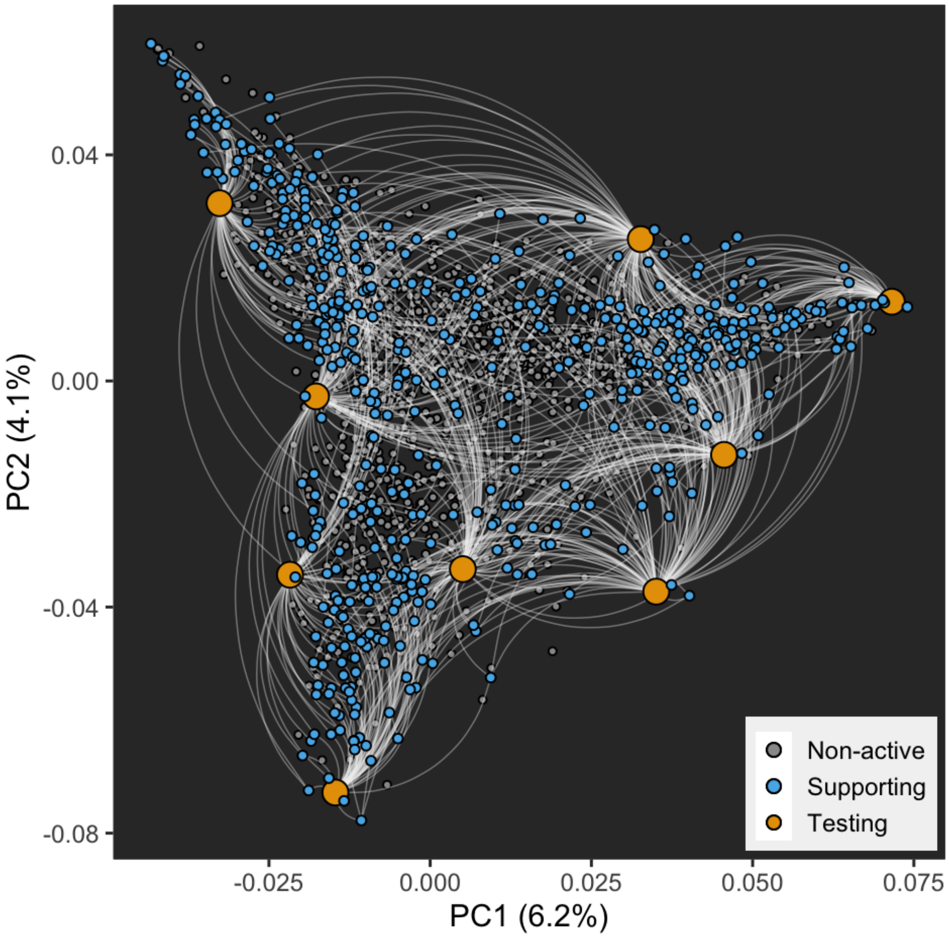

Figure 4: Top two PC of the genomic matrix, **G**. Each point represents and individual either from training or testing set. Orange points represent a sample of individuals from testing set that are connected by lines to individuals from training set with non-zero regression coefficients in the optimum SFI (active). Individuals in training set with a null coefficient do not contribute to the index (non-active)

References

Dekkers, J. C. M. (2007). Prediction of response to marker-assisted and genomic selection using selection index theory. *Journal of Animal Breeding and Genetics 124*, 331–341.

Efron, B., T. Hastie, I. Johnstone, and R. Tibshirani (2004). Least angle regression. *The Annals of Statistics 32*(2), 407–499.

Friedman, J., T. Hastie, H. Höfling, and R. Tibshirani (2007). Pathwise coordinate optimization. *The Annals of Applied Statistics 1*(2), 302–332.

Fu, W. J. (1998). Penalized regressions: the Bridge versus the LASSO. *Journal of Computational and Graphical Statistics 7*(3), 397–416.

Hastie, T., R. Tibshirani, and J. H. Friedman (2009). *The elements of statistical learning: data mining, inference, and prediction* (2nd ed.). New York, USA: Springer.

Hazel, L. N. (1943). The genetic basis for constructing selection indexes. *Genetics 28*(6), 476–490.

Henderson, C. R. (1963). Selection index and expected genetic advance. In *Statistical Genetics and Plant Breeding: A Symposium and Workshop*, Washington, D.C., pp. 141–163. National Academy of Sciences-National Research Council.

Hoerl, A. E. and R. W. Kennard (1970). Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics 12*(1), 55–67.

Lopez-Cruz, M., E. Olson, G. Rovere, J. Crossa, S. Dreisigacker, M. Suchismita, R. Singh, and G. de los Campos (2019). Regularized selection indices for breeding value prediction using hyper-spectral image data. *preprint bioRxiv*.

R Core Team (2019). *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing.

Smith, H. F. (1936). A discrimant function for plant selection. *Annals of Eugenics 7*, 240–250.

Tibshirani, R. (1996). Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society B 58*(1), 267–288.

VanRaden, P. M. (2008). Efficient Methods to Compute Genomic Predictions. *Journal of Dairy Science 91*(11), 4414–4423.

Zou, H. and T. Hastie (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society B 67*(2), 301–320.