Project 1, topic 3: Comparing variable selection methods

Xue Jin, Ruoyuan Qian, Adeline Shin, Rachel Tsong, Alyssa Vanderbeek

February 14, 2020

Objectives

When dealing with high-dimensional data, we are often interested in developing a model that maximizes predictive capability while minimizing complexity. There are several variable selection methods that seek to choose the best set of predictors. Some, including as stepwise forward selection, use a pre-specified criterion like AIC or p-value to iteratively add predictors to a null model until an acceptable threshold is reached. Others such as LASSO or ridge regression, apply a regularization parameter to shrink some coefficients towards zero to aid in feature selection.

Identifying strong predictors of the outcome of interest, and distinguishing them from null predictors, are always central goals of variable selection. But a less understood issue in variable selection is the presence of weak predictors - variables with small but non-zero effects. It may be difficult for variable selection methods to distinguish these weak predictors from null predictors. This challenge can be further complicated by the weak predictors' correlation with other variables. The main objective of this simulation study is to compare the performance of stepwise forward selection to LASSO regression in identifying and estimating strong, weak-but-correlated (WBC), weak-and-independent (WAI), and null predictors.

Statistical Methods Studied

Stepwise Forward Selection

Stepwise forward selection starts with the null model and adds predictors until a given criterion is optimized. In our simulations, we used the Akaike Information Criterion (AIC), which rewards model goodness-of-fit as assessed by the likelihood function while penalizing model complexity (number of parameters). The AIC is defined as follows:

$$\text{AIC} = n \ln(\textstyle\sum_{i=1}^{n} y_i - \hat{y}_i)^2/n) + 2p$$

LASSO

LASSO regression is not a criterion based method. Instead, LASSO shrinks coefficient estimates to zero by applying a penalty term, so the LASSO technique tends to yield sparse models. The equation for estimating LASSO coefficients is:

$$\min_{\beta} \frac{1}{2n} \textstyle\sum_{i=1}^{n} (y_i - x_i\beta)^2 + \lambda || \sum_{k=1}^{p} |\beta_k|$$

Lambda is a tuning parameter that was chosen in our simulations using 10-fold cross-validation.

Scenarios Investigated

Several parameters are specified in our simulations, including the total number of predictors; the number of strong, WBC, WAI, and null predictors, the degree of correlation for the WBC predictors; and the condition that distinguishes strong from weak from null predictors. We vary the number of weak predictors and the degree of correlation to compare forward selection and LASSO regression under 8 scenarios (Table 1). Note that in each scenario, the number of weak predictors is evenly split between WBC and WAI.

Definition of strong signals:

$$S_1 = \{j : |\beta_j| > c\sqrt{log(p)/n},\ \text{some } c > 0, 1{\leq}j{\leq}p\}$$

Definition of WBC signals:

$$S_2 = \{j : 0 < |\beta_j| \leq \sqrt{log(p)/n},\ \text{some } \ c > 0, corr(X_j, X_j') \neq 0, for \ \text{ some } \ j' \in S_1, 1{\leq}j{\leq}p\}$$

Definition of WAI signals:

$$S_3 = \{j : 0 < |\beta_j| \leq \sqrt{log(p)/n},\ \text{some } \ c > 0, corr(X_j, X_j') = 0, for\ all\ j' \in S_1, 1{\leq}j{\leq}p\}$$

Methods for Generating Data

Data is generated using a 4-step method:

1. Fix number of strong, WAI, and WBC predictors
2. Randomly populate a variance-covariance matrix Σ with correlations between strong, weak, and null predictors
3. Generate data $X \sim MVN(0, \Sigma)$
4. Generate coefficients for strong and weak predictors from a $N(0, 1)$ distribution, based on established distinguishing condition
5. Generate outcome data from a linear model

This process insures that the number of different predictor types are fixed, but which predictors fall into which class is varied between simulations.

Performance Measures

Ability of Model to Identify Weak and Strong Predictors

One method of assessing how well a model does in variable selection is in the identification and inclusion of true predictors in a model, as well as the exclusion of null predictors in the final model. Two main metrics are used to test the model ability to identify true predictors: type I error and power. The following definitions were used for these metrics:

$$\text{Type I Error} = \frac{\text{number of null predictors in the model}}{\text{total number of null predictors in the data}}$$

$$\text{Power} = \frac{\text{number of true predictors in the model}}{\text{total number of true predictors in the data}}$$

Effect of Missing Weak Predictors

On the other hand, if a model fails to include a weak parameter by classifying it as a null predictor, this may also affect the outcome and overall fit of the model. To assess the effect of a model missing these weak predictors, the squared error (SE) and mean squared error (MSE) of coefficient estimates was calculated for all eight scenarios according to the following equations:

$$SE = (\hat{\beta} - \beta_0)^2$$

$$MSE = E(\hat{\beta} - \beta_0)^2$$

Results

According to Table 2, forward stepwise always includes more predictors than the number of true predictors in original data while LASSO is more selective.

Scenarios 1-4: Varying total number of weak predictors

Type I error and power

While both methods do are decent at selecting true predictors for the model, LASSO is much better at excluding null predictors than forward selection (Table 3).

Consequently, the type I error of forward stepwise can get quite high. Although it has a decreasing trend while the total number of weak predictors increases, it still chooses more null predictors in the model than LASSO. LASSO is also more stable regardless of the number of weak predictors.

The power of the two methods is similar, and when the number of weak predictors increases, the power of LASSO is slightly higher, meaning that LASSO includes more true predictors than forward stepwise when the number of weak predictors is large.

Coefficient estimation

Although the median SE of the coefficient estimates for forward stepwise is lower than LASSO, its MSE is higher when there are few true weak predictors in the data. Additionally, the variance for forward stepwise is much larger when the weak predictors is lower than 8, suggesting that the estimation of forward stepwise is not stable compared to LASSO when number of weak predictors is small. The MSE under both methods decreases as the number of true weak predictors in the data increases. (Figure 2).

Broken down by predictor type, both methods have inflated MSE of the weak predictors, whereas they have comparatively low MSE for strong and null coefficients. The estimates of all kinds of predictors in LASSO are more accurate than the forward stepwise when the number of weak predictors is smaller than 14. (Figure 3)

As the proportion of weak predictors that are missing from the model increases, the overall MSE across predictor types increases in both methods (Figure 4), due to the increasing proportion of strong predictors that are instead included (Figure 5).

Scenarios 4-8: Varying degree of correlation

Type I error and power

As the correlation of the WBC predictors increases, the type I error for forward stepwise and LASSO remain fairly constant around 0.35 and 0.05, respectively. Meanwhile, the power of forward selection increases, while the power of LASSO decreases slightly. (Figure 6)

Coefficient estimation

Although forward stepwise selection includes more non-null predictors on average (Table 3), it has increasing and higher MSE compared with LASSO, showing a growing trend when correlation becomes larger. The MSE of LASSO is controlled under 0.1, whereas forward selection has an inflated MSE (over 0.2) when the correlation gets very high.

Conclusion

General

Forward stepwise selection always includes more predictors than LASSO, and the model size given by each method increases as the number of total true predictors in the original data increases. The type I error of forward stepwise is consistently inflated compared to LASSO, suggesting that forward selection tends to include more null predictors into the model. The power of two models is similar and performs slightly differently in different scenarios. Finally, the MSE of LASSO is smaller and more stable than the forward stepwise, meaning that although the powers of two methods are close, the estimation of LASSO is more accurate than the forward stepwise.

Varying total number of weak predictors

The SE variance for forward stepwise is much larger than LASSO when the number of weak predictors is small, meaning that the estimation of forward stepwise is not stable compared to LASSO. Besides, when the number of weak predictors increases, both methods have greater chance to include true predictors, the MSE decreases. The total MSE is raised in both methods by the estimation of weak coefficients. Both methods see a decreasing trend in MSE as the ratio of missing weak predictors increases, because the more true predictors in original data, the larger number of strong predictors in the model. Moreover, the estimates of all kinds of predictors in LASSO are more accurate than the forward stepwise when the number of weak predictors is small.

Varying degree of correlation

With increased correlation, the forward stepwise tends to obtain higher power but also higher type I error by including more null predictors into the model, compared with LASSO.

The MSE of coefficient estimates given in forward selection is high compared to LASSO, and increases with increasing degree of correlation.

Discussion

As the degree of correlation increases, the power of LASSO is less than forward stepwise, which may be because that LASSO tends to select only one predictor when the pairwise correlation is very high. An potentially interesting facet of this selection process is LASSO's rate of selecting strong versus WBC predictors, but we did not examine this in our analysis. The fact that forward selection has higher power but higher MSE with increased correlation may be a result of a higher number of correlated variables being selected in the model. Meanwhile, LASSO controls better for this by generally selecting only one of two correlated variables.

Overall, our study shows the relative merits of each variable selection process under different data composition. Although LASSO generally had favorable type I error and MSE, robust to scenarios explored, the choice of method depends ultimately on the specific goal of the model. In this study, we did not look at the predictive ability of either method. LASSO generates more parsimonious models, but it may be that by including

more predictors, forward selection results in more accurate predictions. Depending on the preference for interpretability versus prediction, either method may be suitable.

Additionally, this study explored the performance of these two methods under variation of two variables. Many other parameters may be varied to further assess performance, including the total number of predictors and the condition to distinguish strong from weak from null predictors.

Figures and Tables

Parameter	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6	Scenario 7	Scenario 8
# predictors	50	50	50	50	50	50	50	50
# strong	8	8	8	8	8	8	8	8
# weak	2	8	14	20	8	8	8	8
Total # non-null predictors	6	16	22	28	16	16	16	16
correlation	0.4	0.4	0.4	0.4	0.2	0.4	0.6	0.8
c	4	4	4	4	4	4	4	4

Median total number of predictors selected

Different number of weak predictors | Different correlation

Method	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6	Scenario 7	Scenario 8
Forward Selection	14	19	21	25	18	18	19	18
LASSO	8	13	19	24	14	13	13	13
# true predictors	10	16	22	28	16	16	16	16

Red numbers: > total number of true predictors.
Blue number: < total number of true predictors.

Percent of True Predictors selected per method

Different number of weak predictors | Different correlation

Method	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6	Scenario 7	Scenario 8
Forward Selection	77.4%	81.3%	77.2%	75.8%	80.5%	80.9%	81.5%	81.7%
Lasso	75.8%	80.1%	84.3%	80.7%	80.4%	80.0%	79.9%	79.3%

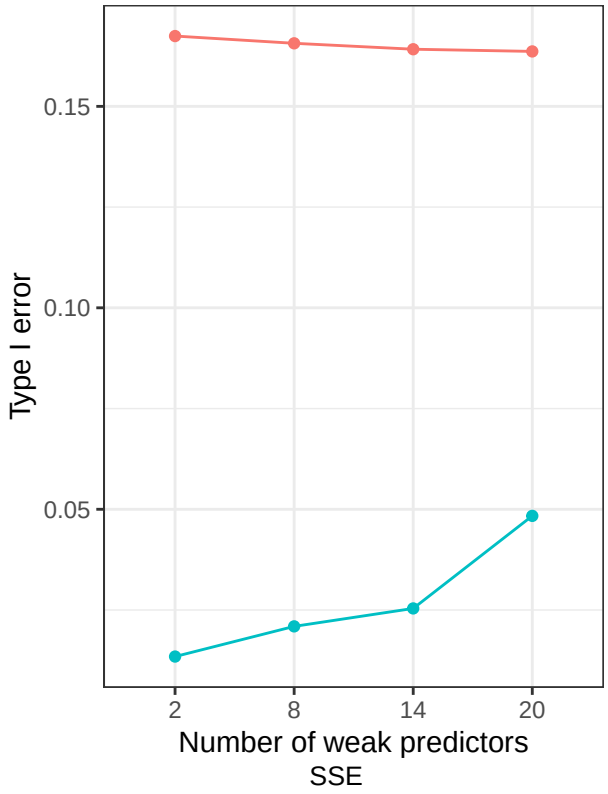
Percent of Null Predictors selected per method

Different number of weak predictors | Different correlation

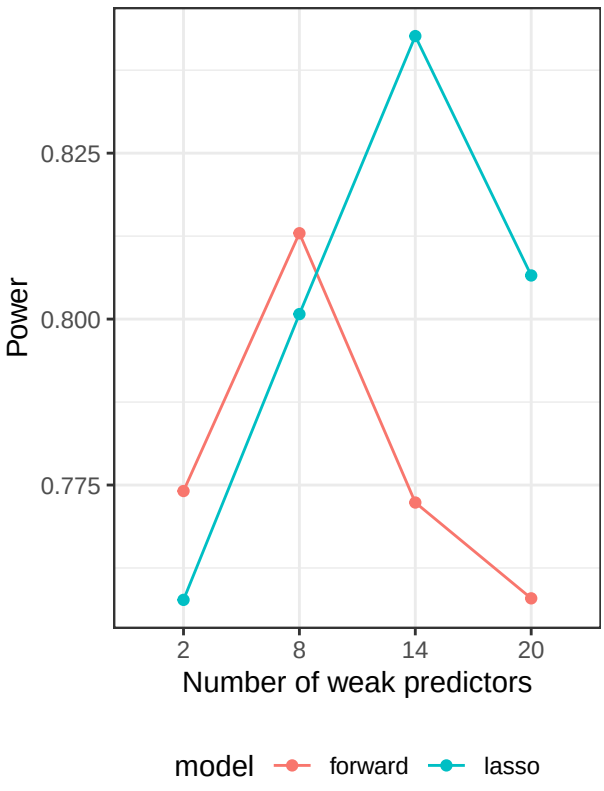
Method	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6	Scenario 7	Scenario 8
Forward Selection	16.7%	16.5%	16.4%	16.3%	16.3%	16.1%	16.3%	16.3%
Lasso	1.3%	2.1%	2.5%	4.8%	2.7%	2.1%	1.5%	1.7%

Type I error under different number of weak

Figure 1

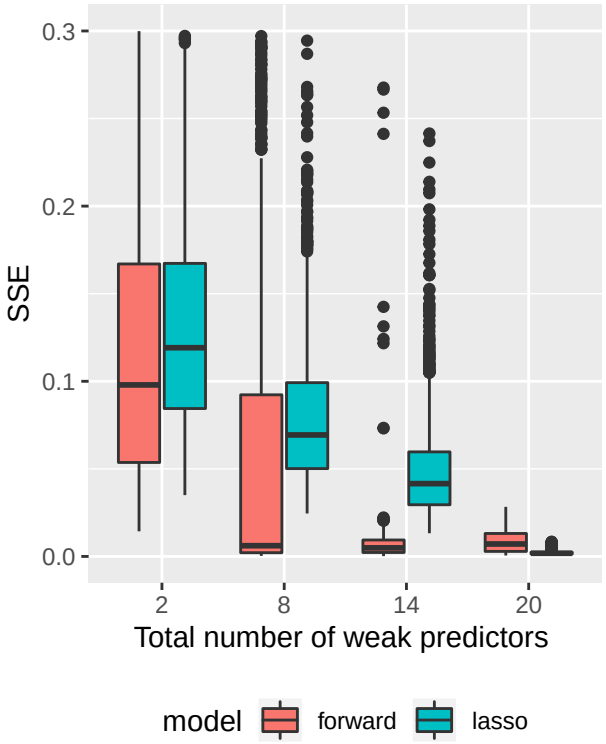


Power



SSE

Figure 2



MSE

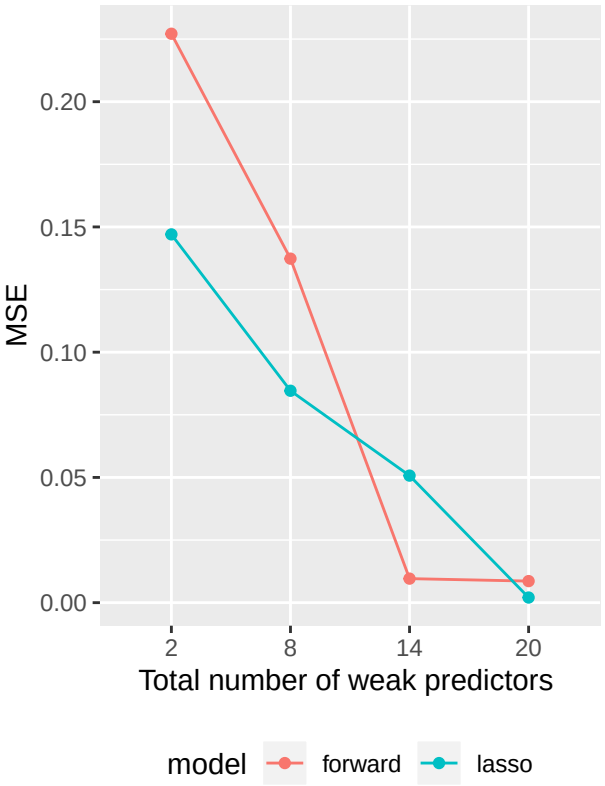
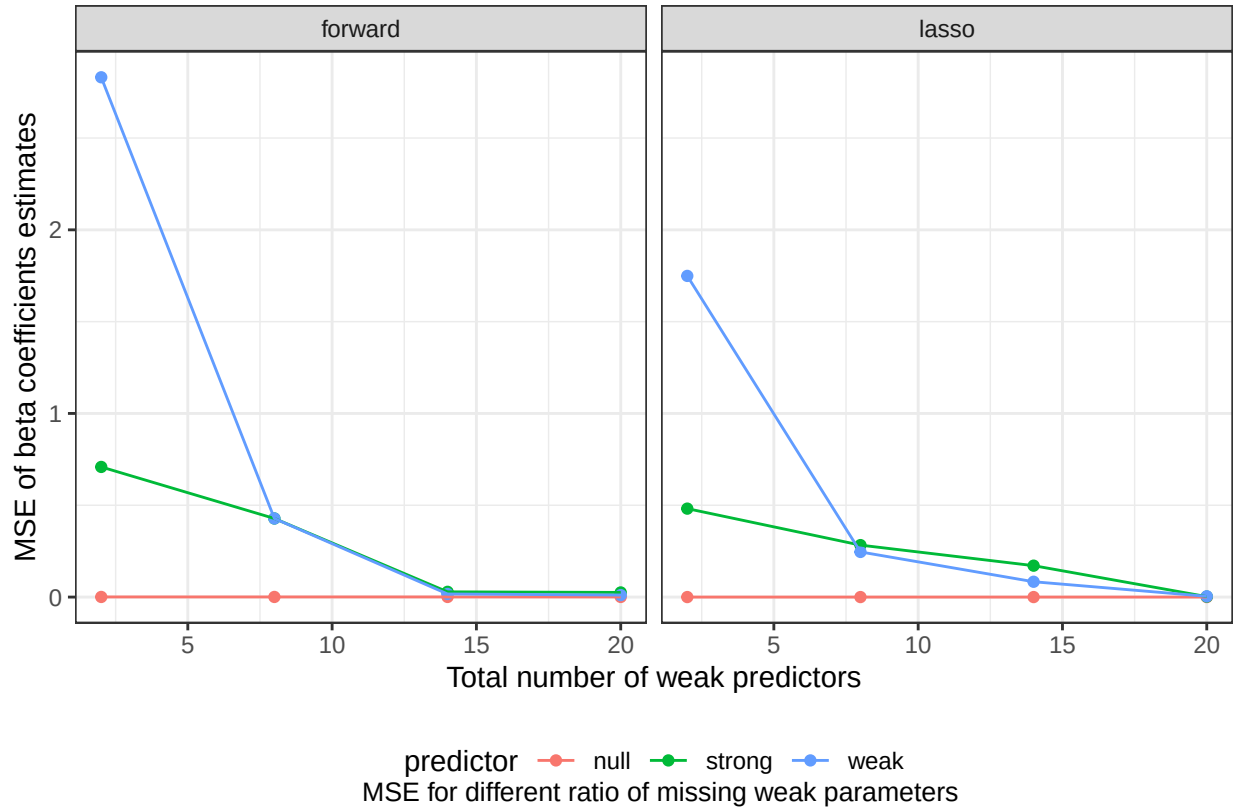


Figure 3



MSE for different ratio of missing weak parameters

Figure 4

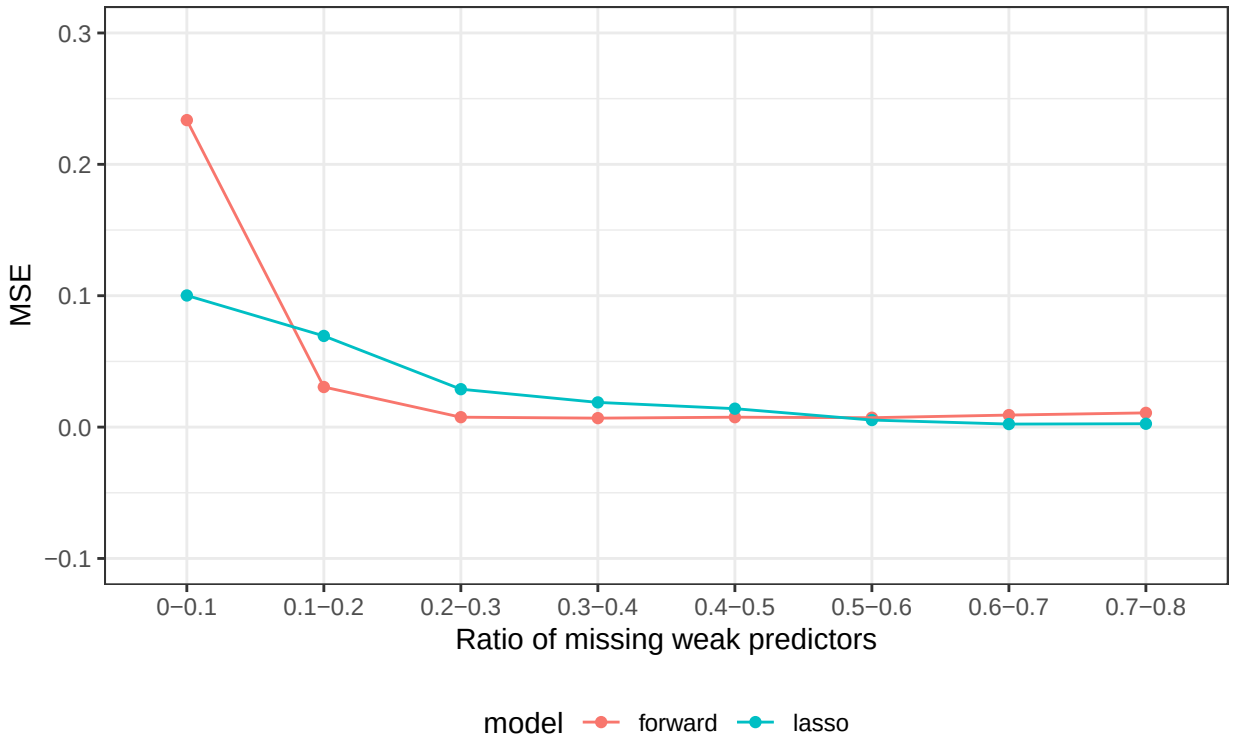
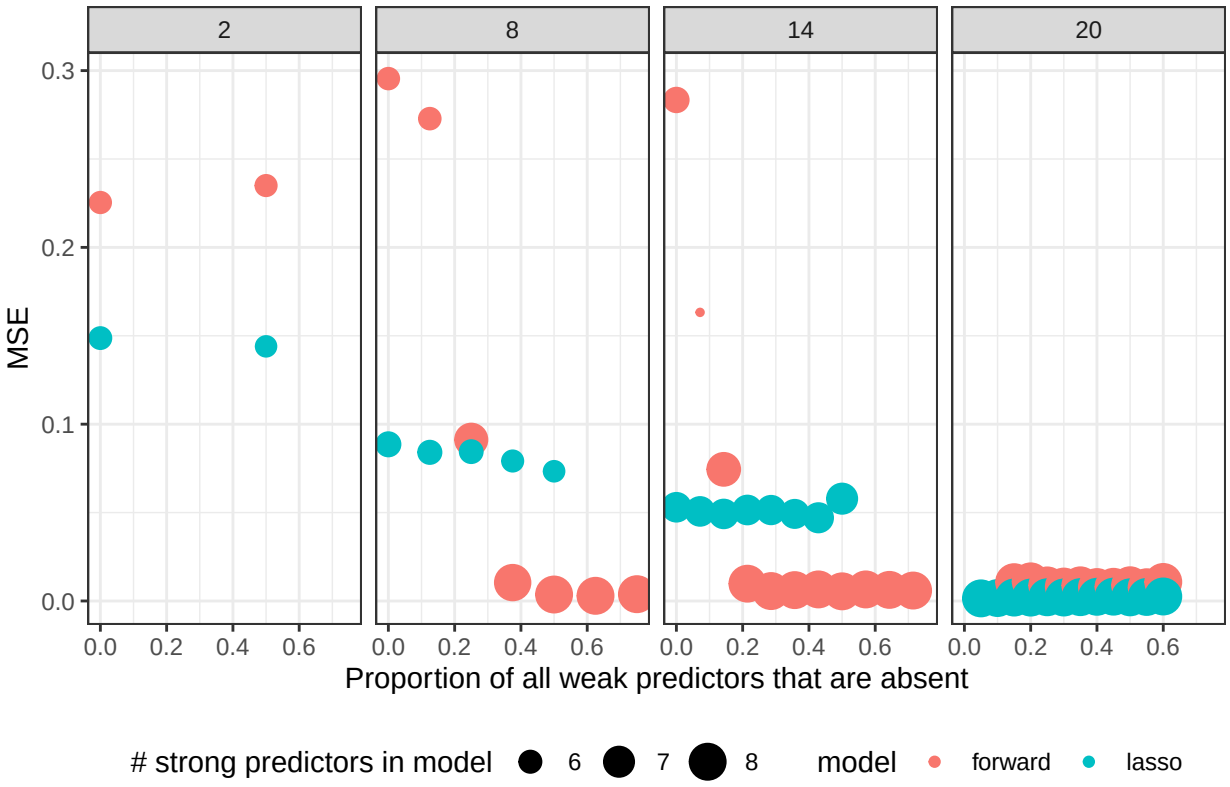
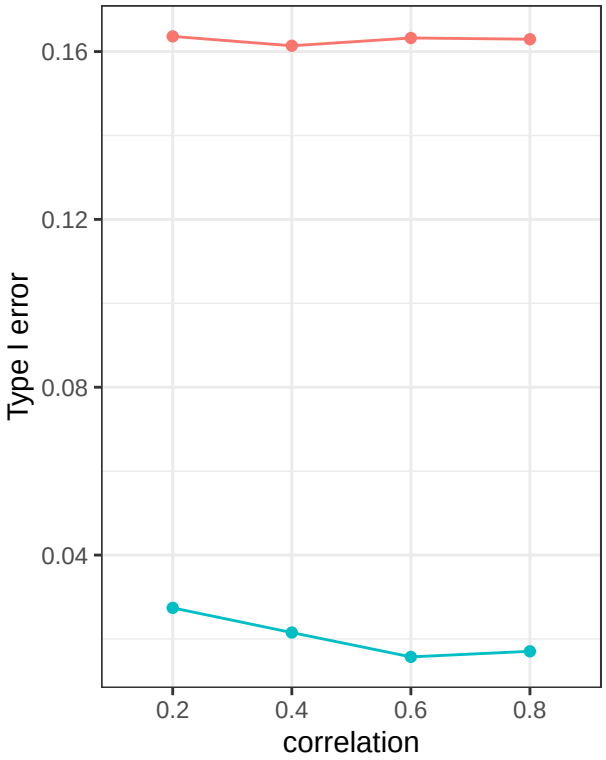


Figure 5

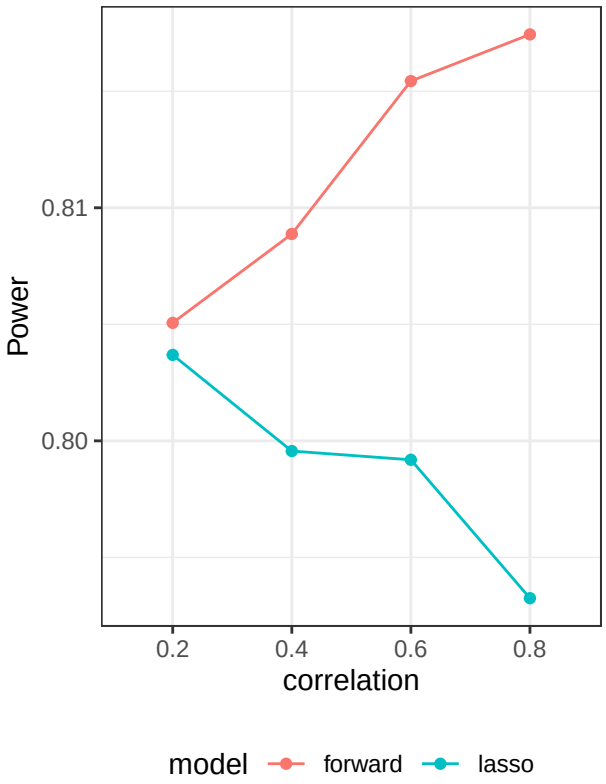

2
8
14
20
MSE
0.3
0.2
0.1
0.0
0.0
0.2
0.4
0.6
Proportion of all weak predictors that are absent
strong predictors in model
6
7
8
model
forward
lasso

Type I error under different correlation

Figure 6

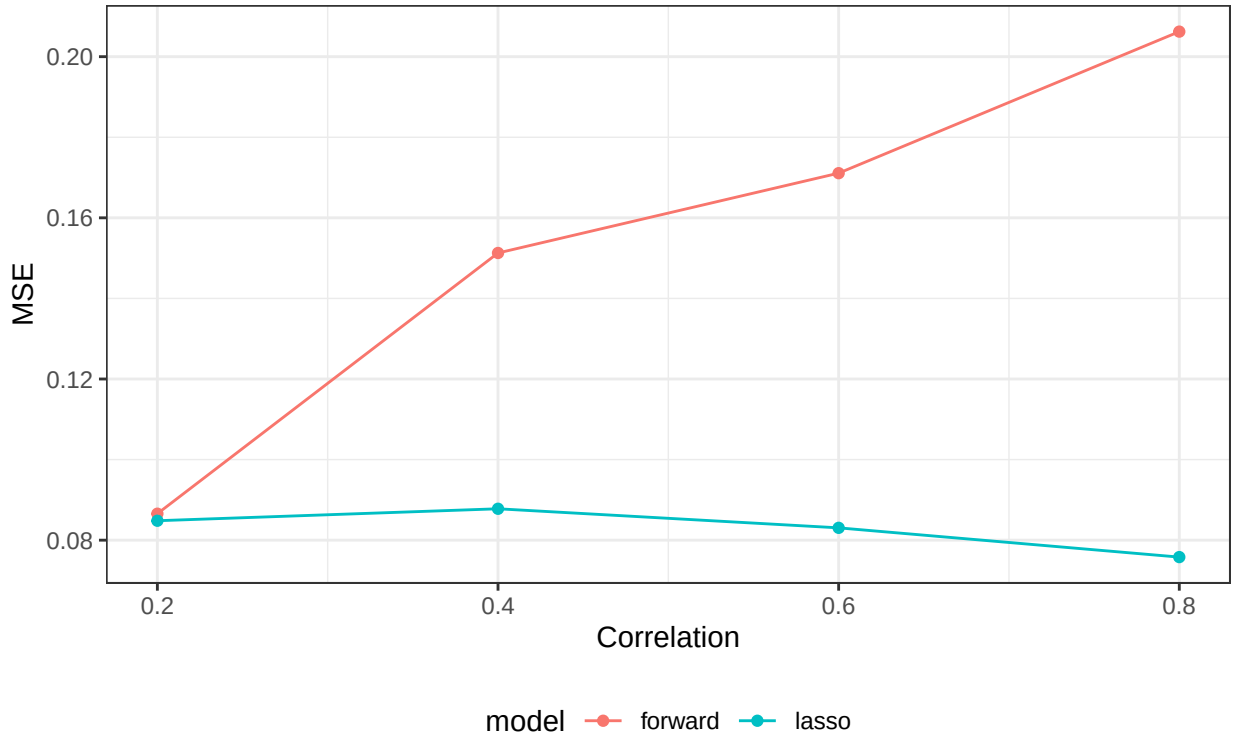


Power



MSE under different correlation

Figure 7



Appendix - R code

```
#### Function to simulate data and fit models

model_fit = function(n.rows, # number of observations
                     p = 50, # number of predictors
                     n.strong, # number of strong predictors
                     n.weak.ind, # number of weak-and-independent predictors
                     n.weak.cor, # number of weak-but-correlated predictors
                     c = 4, # threshold for strong vs weak vs null criteria
                     corr = 0.4) { # degree of correlation

  n.null = p - sum(n.strong, n.weak.ind, n.weak.cor) # number of null predictors based on inputs

  ##### SPECIFY CORRELATION MATRIX
  cor.matrix = diag(p)
  sigma = 1
  b.cor.index = sample(1:p, n.weak.cor)

  # randomly designate index of strong, WAI, WBC, and null predictors
  b.strong.index = sample(seq(1, p, 1)[-b.cor.index], size = n.strong) # randomly designate strong pred
  b.ind.index = sample(seq(1, p, 1)[-c(b.strong.index,b.cor.index)], size = n.weak.ind)
  b.null.index = seq(1, p, 1)[-c(b.strong.index, b.cor.index, b.ind.index)]
  b.weak.index = c(b.cor.index,b.ind.index)
```

```r
cor.matrix[b.strong.index, b.cor.index] = cor.matrix[b.cor.index, b.strong.index] = corr

##### GENERATE DATA from multivariate normal with specified covariance structure
X = MASS::mvrnorm(n = n.rows,
                  mu = rep(0, p),
                  Sigma = sigma*cor.matrix,
                  tol = 0.9)

# set conditions for strong vs. weak predictors
condition = c*sqrt(log(p)/n.rows)

# generate strong coefficients
strong <- NULL
repeat {
  ran = abs(rnorm(1))
  if (ran > condition) {
    strong = c(strong, ran)
  }
  if (length(strong) == n.strong){
    break
  }
}

# generate weak (non-zero) coefficients
weak <- NULL
repeat {
  ran = abs(rnorm(1))
  if (ran <= condition && ran > 0){
    weak = c(weak, ran)
  }
  if (length(weak) == n.weak.cor + n.weak.ind){
    break
  }
}

# list of all coefficients
b.true = rep(0, p)
b.true[b.strong.index] = strong
b.true[b.weak.index] = weak


## generate outcome Y
Y <- 1 + X %*% b.true + rnorm(n.rows)
df <- data.frame(cbind(X, Y))
names(df)[p + 1] <- "y"


##### MODEL FITTING
# Forward Selection
fit.forward <- step(object = lm(y ~ 1, data = df),
                    scope = formula(lm(y ~ ., data = df)),
                    direction = "forward",
                    k = 2,
```

```
                      trace = 0) # AIC
  params.forward = fit.forward$coefficients[-1]
  forward.varnumber = gsub(pattern = "X", replacement = "", x = names(params.forward))
  forward.varnumber = sort(as.numeric(forward.varnumber))

  # LASSO
  fit.lasso <- cv.glmnet(X, Y, nfolds = 10, type.measure = "mse") # 10-fold CV using mean squared error
  param.best <- fit.lasso$glmnet.fit$beta[, fit.lasso$lambda == fit.lasso$lambda.1se] # one standard-er
  params.lasso = param.best[param.best != 0]
  lasso.varnumber = gsub(pattern = "V", replacement = "", x = names(params.lasso))
  lasso.varnumber = as.numeric(lasso.varnumber)


  ##### INFORMATION EXTRACTION
  # truth (null or nonnull predictor)
  predictor.designation = NULL
  for (i in 1:p) {
    if (i %in% c(b.strong.index, b.weak.index)) {
      predictor.designation[i] = 1
    } else {
      predictor.designation[i] = 0
    }
  }

  # inclusion of predictor in model
  included.forward = NULL
  included.lasso = NULL
  for (i in 1:p) {
    included.forward[i] = (i %in% forward.varnumber)
    included.lasso[i] = (i %in% lasso.varnumber)
  }


  return(list(
    predictor.designation = predictor.designation, # designation of null verses non.null
    included.forward = included.forward, # T/F inclusion in forward model
    included.lasso = included.lasso, # T/F inclusion in LASSO model
    b.true = b.true, # true coefficients
    b.forward = params.forward, # forward coefficient estimates
    b.lasso = params.lasso, # LASSO coefficient estimates
    weak = b.weak.index # index of weak predictors
  ))
}
```

```
### Function to calculate chosen metrics for each model, taken from model_fit() function

metrics = function(models, # object returned from model_fit()
                   p = 50, # number of predictors
                   n.strong, # number of strong predcitors
                   n.weak.ind, # number of WAI predictors
                   n.weak.cor, # number of WBC predictors
                   n.rows = 1000, # number of observations in the data
                   c = 4){ # threshold for strong vs weak vs null criteria
```

```
##### EXTRACT FROM models
truth.df = lapply(models, `[[`, 1)
forward = lapply(models, `[[`, 2)
lasso = lapply(models, `[[`, 3)
beta.true = lapply(models, `[[`, 4)
beta.forward = lapply(models, `[[`, 5)
beta.lasso = lapply(models, `[[`, 6)
weak.preds = lapply(models, `[[`, 7)

n.true = sum(n.strong, n.weak.ind, n.weak.cor)

condition = c*sqrt(log(p)/n.rows)



### Size of models
forward.size = do.call(rbind,
                       lapply(1:N, function(i){
                         sum(forward[i][[1]])
                       })
)
lasso.size = do.call(rbind,
                     lapply(1:N, function(i){
                       sum(lasso[i][[1]])
                     })
)

### Number of strong predictors in model
forward.strong.n = do.call(rbind,
                           lapply(1:N, function(i){
                             strong.index = which(beta.true[i][[1]] > condition)
                             model.index = which(forward[i][[1]] == TRUE)

                             sum(model.index %in% strong.index)
                           })
)
lasso.strong.n = do.call(rbind,
                         lapply(1:N, function(i){
                           strong.index = which(beta.true[i][[1]] > condition)
                           model.index = which(lasso[i][[1]] == TRUE)

                           sum(model.index %in% strong.index)
                         })
)

### Average percentage of true predictors chosen by each model
forward.true = do.call(rbind,
                       lapply(1:N, function(i){
                         ind = which(truth.df[[i]] == 1) # index of which parameters are true
                         forward.nonnull = forward[[i]][ind] # did the forward method select them?

                         return(forward.nonnull)
                       })
```

```
)
forward.pcttrue = mean(rowSums(forward.true)) / n.true

lasso.true = do.call(rbind,
                     lapply(1:N, function(i){
                       ind = which(truth.df[[i]] == 1) # index of which parameters are true
                       lasso.nonnull = lasso[[i]][ind] # did the lasso method select them?

                       return(lasso.nonnull)
                     })
)
lasso.pcttrue = mean(rowSums(lasso.true)) / n.true

### Average percentage of true predictors chosen by each model
forward.false = do.call(rbind,
                        lapply(1:N, function(i){
                          ind = which(truth.df[[i]] == 0) # index of which parameters are null
                          forward.nonnull = forward[[i]][ind] # did the forward method select them?

                          return(forward.nonnull)
                        })
)
forward.pctnull = mean(rowSums(forward.false)) / (p - n.true)

lasso.false = do.call(rbind,
                      lapply(1:N, function(i){
                        ind = which(truth.df[[i]] == 0) # index of which parameters are null
                        lasso.nonnull = lasso[[i]][ind] # did the lasso method select them?

                        return(lasso.nonnull)
                      })
)
lasso.pctnull = mean(rowSums(lasso.false)) / (p - n.true)


### percentage of weak predictors
forward.weak = do.call(rbind,
                       lapply(1:N, function(i){
                         ind = weak.preds[[i]] # index of which parameters are null
                         forward.nonnull = forward[[i]][ind] # did the forward method select them?

                         return(forward.nonnull)
                       })
)
forward.pctweak = rowSums(forward.weak) / (n.weak.cor + n.weak.ind)

lasso.weak = do.call(rbind,
                     lapply(1:N, function(i){
                       ind = weak.preds[[i]]  # index of which parameters are null
                       lasso.nonnull = lasso[[i]][ind] # did the lasso method select them?

                       return(lasso.nonnull)
                     })
```

```
)
lasso.pctweak = rowSums(lasso.weak) / (n.weak.cor + n.weak.ind)

### MSE for strong vs weak vs null predictors
forward. = do.call(rbind,
                           lapply(1:N, function(i){
                             strong.index = which(beta.true[i][[1]] > condition)
                             model.index = which(forward[i][[1]] == TRUE)

                             sum(model.index %in% strong.index)
                           })
)
lasso.strong.n = do.call(rbind,
                         lapply(1:N, function(i){
                           strong.index = which(beta.true[i][[1]] > condition)
                           model.index = which(lasso[i][[1]] == TRUE)

                           sum(model.index %in% strong.index)
                         })
)

MSE.df = do.call(rbind, lapply(1:N, function(i){
  params.forward = beta.forward[[i]]
  params.lasso = beta.lasso[[i]]
  b.true = beta.true[[i]]
  strong.index = which(beta.true[i][[1]] > condition)
  weak.index = which(beta.true[i][[1]] <= condition & beta.true[i][[1]] > 0)
  null.index = which(beta.true[i][[1]] == 0)

  # get beta estimates from each model
  names(params.forward) = as.numeric(gsub("X", "", names(params.forward)))
  forward.betas = data.frame(X = as.numeric(names(params.forward)),
                             b.forward = params.forward)
  names(params.lasso) = as.numeric(gsub("V", "", names(params.lasso)))
  lasso.betas = data.frame(X = as.numeric(names(params.lasso)),
                           b.lasso = params.lasso)

  # calculated MSE across betas for each model
  beta.estimates = data.frame(X = 1:p,
                              b.true = b.true) %>% # true betas
    left_join(forward.betas, by = "X") %>% # forward selection
    left_join(lasso.betas, by = "X") %>% # lasso
    mutate_all(funs(replace_na(., 0))) %>% # let missing params be 0
    mutate(MSE.forward = (b.forward - b.true)^2, # calculate MSE of each beta
           MSE.lasso = (b.lasso - b.true)^2) %>%
    summarise(avg.MSE.forward = mean(MSE.forward, na.rm = TRUE), # average MSE across betas
              avg.MSE.lasso = mean(MSE.lasso, na.rm = TRUE),
              # calculate MSE for strong and weak predictors
              avg.forward.strong = mean((b.forward[strong.index] - b.true[strong.index])^2, na.rm = TR
              avg.forward.weak = mean((b.forward[weak.index] - b.true[weak.index])^2, na.rm = TRUE),
              avg.lasso.strong = mean((b.lasso[strong.index] - b.true[strong.index])^2, na.rm = TRUE)
              avg.lasso.weak = mean((b.lasso[weak.index] - b.true[weak.index])^2, na.rm = TRUE),
              avg.forward.null = mean((b.forward[null.index] - b.true[null.index])^2, na.rm = TRUE),
```

```
                avg.lasso.null = mean((b.lasso[null.index] - b.true[null.index])^2, na.rm = TRUE)) %>%
      unlist

    return(beta.estimates)
  })
  ) %>%
    cbind(., forward.pctweak) %>%
    cbind(., lasso.pctweak)


  return(list(
    model.size = c(forward = forward.size, lasso = lasso.size), # model sizes
    n.weak = c(forward = rowSums(forward.weak), lasso = rowSums(lasso.weak)), # number of weak predicto
    n.strong = c(forward = forward.strong.n, lasso = lasso.strong.n), # number of strong predictors cho
    pcttrue = c(forward = forward.pcttrue, lasso = lasso.pcttrue), # power
    pctnull = c(forward = forward.pctnull, lasso = lasso.pctnull), # type I error
    MSE.df = MSE.df # MSE
  ))
}
```