Lab 4

Viktor Sjöberg

2023-12-01

Part 1

The first part is just the code/ answers to the questions.

1.1 Set up

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-8
```

```
library(ggplot2)
library(bigstep)
library(MASS)
library(SLOPE)
library(gt)
setwd("/Users/viktorsjoberg/Desktop/high-dimensional/Assignment 4")
# Set the parameters
n_samples <- 1000
n_features <- 950
k <- 20
sigma <- 1 / sqrt(1000)

# Generate the design matrix X with elements from N(0, sigma)
X <- matrix(rnorm(n_samples * n_features, mean = 0, sd = sigma),
            nrow = n_samples, ncol = n_features)

# Generate the beta coefficients
# First k beta coefficients are 6, the rest are 0
beta <- c(rep(6, k), rep(0, n_features - k))

# Generate the error term epsilon ~ N(0, I)
epsilon <- rnorm(n_samples, mean = 0, sd = 1)

# Generate the response variable Y = X %*% beta + epsilon
Y <- X %*% beta + epsilon

# Display the first 5 elements of the response variable
Y[1:5]
```

```
## [1] -1.03766346 -2.39437674 -0.78541953  0.06105576  2.34747873
```

1.2 BIC

```
model <- prepare_data(Y,X)

# Apply mBIC2 criterion for variable selection
selected_model <- stepwise(model, crit = bic)
```

```
## Starting stepwise, 0 variables, crit = 521.57, MSE = 1.685.
## Variable 4 added with crit = 476.42, MSE = 1.599.
## Variable 11 added with crit = 446.36, MSE = 1.541.
## Variable 9 added with crit = 422.03, MSE = 1.494.
## Variable 14 added with crit = 396.07, MSE = 1.445.
## Variable 18 added with crit = 371.07, MSE = 1.4.
## Variable 8 added with crit = 350.69, MSE = 1.362.
## Variable 1 added with crit = 331.42, MSE = 1.327.
## Variable 5 added with crit = 313.12, MSE = 1.294.
## Variable 17 added with crit = 292.88, MSE = 1.259.
## Variable 7 added with crit = 273, MSE = 1.226.
## Variable 3 added with crit = 251.86, MSE = 1.192.
## Variable 2 added with crit = 232.63, MSE = 1.162.
## Variable 10 added with crit = 212.73, MSE = 1.131.
## Variable 6 added with crit = 191.34, MSE = 1.099.
## Variable 12 added with crit = 173.98, MSE = 1.073.
## Variable 13 added with crit = 157.24, MSE = 1.048.
## Variable 20 added with crit = 137.64, MSE = 1.02.
## Variable 15 added with crit = 120.38, MSE = 0.996.
## Variable 16 added with crit = 107.24, MSE = 0.976.
## Variable 19 added with crit = 94.16, MSE = 0.957.
## Variable 793 added with crit = 80.08, MSE = 0.937.
## Variable 126 added with crit = 77.42, MSE = 0.928.
## Variable 562 added with crit = 74.41, MSE = 0.919.
## Variable 746 added with crit = 72.08, MSE = 0.911.
## Variable 710 added with crit = 70.42, MSE = 0.903.
## Variable 87 added with crit = 69.36, MSE = 0.896.
## Variable 88 added with crit = 68.55, MSE = 0.889.
## Variable 552 added with crit = 67.98, MSE = 0.882.
## Variable 373 added with crit = 67.04, MSE = 0.875.
## Variable 382 added with crit = 66.96, MSE = 0.869.
```

```
## Done.
```

```
# Summarize the selected model & Calculate square estimation error
selected_model_s <- summary(selected_model)
name<-selected_model$model
BIC_name <-as.numeric(name)
BIC <-rep(0,n_features)
BIC[BIC_name] <- (
  selected_model_s$coefficients[2:(length(selected_model$model)+1),1])
BIC_se <- sum((BIC-beta)^2)
BIC_se_x <- sum((X%*%(BIC-beta))^2)

PowerBIC <- sum(abs(BIC[1:k])>0)/k
FDPBIC <- sum(abs(BIC[(k+1):n_features])>0)/sum(abs(BIC[1:n_features])>0)
```

1.3 Ridge

```
X_matrix <- as.matrix(X)

# Perform cross-validated ridge regression
set.seed(123) # For reproducibility
cv_ridge <- cv.glmnet(X_matrix, Y, alpha = 0, standardize = TRUE)

# The best lambda (tuning parameter) according to cross-validation
best_lambda_ridge <- cv_ridge$lambda.min

# Fit the ridge regression model with the selected best lambda
ridge_model <- glmnet(X_matrix, Y, alpha = 0, lambda = best_lambda_ridge,
                      standardize = TRUE)

# Coefficients of the ridge regression model
ridge_coefficients <- coef(ridge_model)

ridge_se <- sum((ridge_coefficients[-1] - beta)^2)
ridge_se_x <- sum((X_matrix%*%(ridge_coefficients[-1]-beta))^2)
```

1.4 LASSO and Cross Validation

```
cv_lasso <- cv.glmnet(X_matrix, Y, alpha = 1, standardize = TRUE)

# The best lambda according to cross-validation
best_lambda_lasso_cv <- cv_lasso$lambda.min
# Lambda that is within one standard error of the minimum

lambda_cv_1se <- cv_lasso$lambda.1se

# Fit the lasso model with the best lambda
lasso_cv_model_min <- glmnet(X_matrix, Y, alpha = 1,
                             lambda = best_lambda_lasso_cv, standardize = TRUE)

# Fit the lasso model with lambda.1se
lasso_cv_model_1se <- glmnet(X_matrix, Y, alpha = 1,
```

```
                              lambda = lambda_cv_1se, standardize = TRUE)

# Coefficients of the lasso model using lambda.min
coefficients_lasso_cv_min <- coef(lasso_cv_model_min, s = "lambda.min")

# Coefficients of the lasso model using lambda.1se
coefficients_lasso_cv_1se <- coef(lasso_cv_model_1se, s = "lambda.1se")

lasso_cv_min_se <- sum((coefficients_lasso_cv_min[-1]-beta)^2)
lasso_cv_min_se_x <- sum((X_matrix%*%(coefficients_lasso_cv_min[-1]-beta))^2)
lasso_cv_min_power <- sum(abs(coefficients_lasso_cv_min[1:k])>0)/k
lassocv_cv_min_FDP <-(
  sum(abs(coefficients_lasso_cv_min[(k+1):n_features])>0)/
  sum(abs(coefficients_lasso_cv_min[1:n_features])>0))

lasso_cv_1se_se <- sum((coefficients_lasso_cv_1se[-1]-beta)^2)
lasso_cv_1se_se_x <- sum((X_matrix%*%(coefficients_lasso_cv_1se[-1]-beta))^2)
lasso_cv_1se_power <- sum(abs(coefficients_lasso_cv_1se[1:k])>0)/k
lassocv_cv_1se_FDP <- (
  sum(abs(coefficients_lasso_cv_1se[(k+1):n_features])>0)/
  sum(abs(coefficients_lasso_cv_1se[1:n_features])>0))
```

1.5 LASSO with pre-determined λ

```
lambda_pre <- qnorm(1 - 0.1 / (2*n_features)) / n_samples

# Perform LASSO regression with this specific lambda
lasso_model_pre <- glmnet(as.matrix(X), Y, alpha = 1, lambda = lambda_pre,
                          standardize = FALSE)

# Coefficients of the LASSO model
lasso_pre_coefficients <- coef(lasso_model_pre)


lasso_pre_se <- sum((lasso_pre_coefficients[-1]-beta)^2)
lasso_pre_se_x <- sum((X_matrix%*%(lasso_pre_coefficients[-1]-beta))^2)
lasso_pre_power <- sum(abs(lasso_pre_coefficients[1:k])>0)/k
lassocv_pre_FDP <-(
  sum(abs(lasso_pre_coefficients[(k+1):n_features])>0)/
    sum(abs(lasso_pre_coefficients[1:n_features])>0))
```

1.6 SLOPE

```
lambda_slope <- qnorm(1 - (1:n_features) * 0.1 / (2 * n_features)) / n_samples

# Fit the SLOPE model
slope_model <- SLOPE(X, Y, lambda = lambda_slope, alpha=1,intercept=FALSE,
                     scale='none', solver='fista')

# Coefficients of the SLOPE model
slope_coefficients <- coef(slope_model)
```

```r
slope_se <- sum((slope_coefficients-beta)^2)
slope_se_x <- sum((X_matrix%*%(slope_coefficients-beta))^2)
slope_power <- sum(abs(slope_coefficients[1:k])>0)/k
slope_FDP <-(
  sum(abs(slope_coefficients[(k+1):n_features])>0)/
    sum(abs(slope_coefficients[1:n_features])>0))
```

1.7 Results

```r
metrics_df <- data.frame(
  Method = c("BIC", "Ridge", "LASSO_CV_Min",
             "LASSO_CV_1se", "LASSO_Pre", "SLOPE"),
  SE = c(BIC_se, ridge_se, lasso_cv_min_se,
         lasso_cv_1se_se, lasso_pre_se, slope_se),
  SE_X = c(BIC_se_x, ridge_se_x, lasso_cv_min_se_x,
           lasso_cv_1se_se_x, lasso_pre_se_x, slope_se_x),
  Power = c(PowerBIC, NA, lasso_cv_min_power,
            lasso_cv_1se_power, lasso_pre_power, slope_power),
  FDP = c(FDPBIC, NA, lassocv_cv_min_FDP,
          lassocv_cv_1se_FDP, lassocv_pre_FDP, slope_FDP)
)

# Ensure that NAs are displayed as blank cells
metrics_df[is.na(metrics_df)] <- ""

# Create the gt table
gt_table <- gt(metrics_df)


# Display the table
gt_output <- "img/gt_table.png"
gtsave(gt_table, gt_output)
```

Method	SE	SE_X	Power	FDP
BIC	108.8919	107.0241	1	0.333333333333333
Ridge	499.1261	333.7187		
LASSO_CV_Min	118.3835	108.3115	1	0.834710743801653
LASSO_CV_1se	148.5402	138.5208	1	0.565217391304348
LASSO_Pre	347.4396	331.6647	0.95	0.0952380952380952
SLOPE	254.9937	241.9819	1	0.333333333333333

Part 2 LOOP threw different beta

```
# Initialize lists to store results
results_list <- list()

# Loop over beta values
for(beta_value in 2:6) {
  n_samples <- 1000
  n_features <- 950
  k <- 20
  sigma <- 1 / sqrt(1000)

  # Generate the design matrix X with elements from N(0, sigma)
  X <- matrix(rnorm(n_samples * n_features, mean = 0, sd = sigma),
              nrow = n_samples, ncol = n_features)

  beta <- c(rep(beta_value, k), rep(0, n_features - k))
  epsilon <- rnorm(n_samples, mean = 0, sd = 1)

  Y <- X %*% beta + epsilon

  model <- prepare_data(Y,X)

  # Apply mBIC2 criterion for variable selection
  selected_model <- stepwise(model, crit = bic)

  # Summarize the selected model & Calculate square estimation error
  selected_model_s <- summary(selected_model)
  name<-selected_model$model
  BIC_name <-as.numeric(name)
  BIC <-rep(0,n_features)
  BIC[BIC_name] <- (
    selected_model_s$coefficients[2:(length(selected_model$model)+1),1])
  BIC_se <- sum((BIC-beta)^2)
  BIC_se_x <- sum((X%*%(BIC-beta))^2)

  PowerBIC <- sum(abs(BIC[1:k])>0)/k
  FDPBIC <- sum(abs(BIC[(k+1):n_features])>0)/sum(abs(BIC[1:n_features])>0)


  ############ ridge #############
  X_matrix <- as.matrix(X)

  # Perform cross-validated ridge regression
  set.seed(123) # For reproducibility
  cv_ridge <- cv.glmnet(X_matrix, Y, alpha = 0, standardize = TRUE)

  # The best lambda (tuning parameter) according to cross-validation
  best_lambda_ridge <- cv_ridge$lambda.min

  # Fit the ridge regression model with the selected best lambda
  ridge_model <- glmnet(X_matrix, Y, alpha = 0,
                        lambda = best_lambda_ridge, standardize = TRUE)
```

```r
# Coefficients of the ridge regression model
ridge_coefficients <- coef(ridge_model)

ridge_se <- sum((ridge_coefficients[-1] - beta)^2)
ridge_se_x <- sum((X_matrix%*%(ridge_coefficients[-1]-beta))^2)


###### LASSO Cross Validation #######

cv_lasso <- cv.glmnet(X_matrix, Y, alpha = 1, standardize = TRUE)

# The best lambda according to cross-validation
best_lambda_lasso_cv <- cv_lasso$lambda.min
# Lambda that is within one standard error of the minimum

lambda_cv_1se <- cv_lasso$lambda.1se

# Fit the lasso model with the best lambda
lasso_cv_model_min <- glmnet(X_matrix, Y, alpha = 1,
                             lambda = best_lambda_lasso_cv,
                             standardize = TRUE)

# Fit the lasso model with lambda.1se
lasso_cv_model_1se <- glmnet(X_matrix, Y, alpha = 1,
                             lambda = lambda_cv_1se,
                             standardize = TRUE)

# Coefficients of the lasso model using lambda.min
coefficients_lasso_cv_min <- coef(lasso_cv_model_min, s = "lambda.min")

# Coefficients of the lasso model using lambda.1se
coefficients_lasso_cv_1se <- coef(lasso_cv_model_1se, s = "lambda.1se")

lasso_cv_min_se <- sum((coefficients_lasso_cv_min[-1]-beta)^2)
lasso_cv_min_se_x <- sum((X_matrix%*%(coefficients_lasso_cv_min[-1]-beta))^2)
lasso_cv_min_power <- sum(abs(coefficients_lasso_cv_min[1:k])>0)/k
lassocv_cv_min_FDP <-(
  sum(abs(coefficients_lasso_cv_min[(k+1):n_features])>0)/
    sum(abs(coefficients_lasso_cv_min[1:n_features])>0))

lasso_cv_1se_se <- sum((coefficients_lasso_cv_1se[-1]-beta)^2)
lasso_cv_1se_se_x <- sum((X_matrix%*%(coefficients_lasso_cv_1se[-1]-beta))^2)
lasso_cv_1se_power <- sum(abs(coefficients_lasso_cv_1se[1:k])>0)/k
lassocv_cv_1se_FDP <- (
  sum(abs(coefficients_lasso_cv_1se[(k+1):n_features])>0)/
    sum(abs(coefficients_lasso_cv_1se[1:n_features])>0))

####### LASSO with pre-determined \( \lambda \) ########

lambda_pre <- qnorm(1 - 0.1 / (2*n_features)) / n_samples

# Perform LASSO regression with this specific lambda
lasso_model_pre <- glmnet(as.matrix(X), Y, alpha = 1,
```

```
                            lambda = lambda_pre, standardize = FALSE)

  # Coefficients of the LASSO model
  lasso_pre_coefficients <- coef(lasso_model_pre)


  lasso_pre_se <- sum((lasso_pre_coefficients[-1]-beta)^2)
  lasso_pre_se_x <- sum((X_matrix%*%(lasso_pre_coefficients[-1]-beta))^2)
  lasso_pre_power <- sum(abs(lasso_pre_coefficients[1:k])>0)/k
  lassocv_pre_FDP <-(
    sum(abs(lasso_pre_coefficients[(k+1):n_features])>0)/
       sum(abs(lasso_pre_coefficients[1:n_features])>0))

  ##### SLOPE ########

  lambda_slope <- qnorm(1 - (1:n_features) * 0.1 / (2 * n_features)) / n_samples

  # Fit the SLOPE model
  slope_model <- SLOPE(X, Y, lambda = lambda_slope,
                       alpha=1,intercept=FALSE, scale='none', solver='fista')

  # Coefficients of the SLOPE model
  slope_coefficients <- coef(slope_model)

  slope_se <- sum((slope_coefficients-beta)^2)
  slope_se_x <- sum((X_matrix%*%(slope_coefficients-beta))^2)
  slope_power <- sum(abs(slope_coefficients[1:k])>0)/k
  slope_FDP <-(
    sum(abs(slope_coefficients[(k+1):n_features])>0)/
      sum(abs(slope_coefficients[1:n_features])>0))

  results_list[[as.character(beta_value)]] <- data.frame(
    Method = c("BIC", "Ridge", "LASSO_CV_Min",
               "LASSO_CV_1se", "LASSO_Pre", "SLOPE"),
    Beta_Value = beta_value,
    SE = c(BIC_se, ridge_se, lasso_cv_min_se,
           lasso_cv_1se_se, lasso_pre_se, slope_se),
    SE_X = c(BIC_se_x, ridge_se_x, lasso_cv_min_se_x,
             lasso_cv_1se_se_x, lasso_pre_se_x, slope_se_x),
    Power = c(PowerBIC, NA, lasso_cv_min_power,
              lasso_cv_1se_power, lasso_pre_power, slope_power),
    FDP = c(FDPBIC, NA, lassocv_cv_min_FDP,
            lassocv_cv_1se_FDP, lassocv_pre_FDP, slope_FDP)
  )
}
```

```
## Starting stepwise, 0 variables, crit = -39.86, MSE = 0.961.

## Variable 19 added with crit = -57.88, MSE = 0.937.

## Variable 5 added with crit = -63.64, MSE = 0.925.

## Variable 9 added with crit = -67.77, MSE = 0.915.

## Variable 10 added with crit = -71.3, MSE = 0.906.
```

```
## Variable 369 added with crit = -74.31, MSE = 0.897.
## Variable 856 added with crit = -76.32, MSE = 0.889.
## Variable 45 added with crit = -78.2, MSE = 0.881.
## Variable 54 added with crit = -79.57, MSE = 0.874.
## Variable 2 added with crit = -80.71, MSE = 0.867.
## Variable 1 added with crit = -82.3, MSE = 0.86.
## Variable 153 added with crit = -84, MSE = 0.852.
## Variable 315 added with crit = -85.18, MSE = 0.845.
## Variable 13 added with crit = -86.45, MSE = 0.838.
## Variable 394 added with crit = -87.87, MSE = 0.831.
## Variable 109 added with crit = -89.24, MSE = 0.825.
## Variable 3 added with crit = -89.87, MSE = 0.818.
## Variable 11 added with crit = -90.93, MSE = 0.812.
## Variable 266 added with crit = -92.08, MSE = 0.805.
## Variable 907 added with crit = -93.41, MSE = 0.799.
## Variable 615 added with crit = -93.9, MSE = 0.793.
## Variable 45 removed with crit = -94.59, MSE = 0.798.
## Variable 20 added with crit = -95.47, MSE = 0.792.
## Variable 46 added with crit = -96.48, MSE = 0.785.
## Variable 94 added with crit = -97.14, MSE = 0.779.
## Variable 4 added with crit = -98.58, MSE = 0.773.
## Variable 344 added with crit = -98.92, MSE = 0.767.
## Variable 832 added with crit = -99.95, MSE = 0.761.
## Variable 626 added with crit = -100.34, MSE = 0.756.
## Done.
## Starting stepwise, 0 variables, crit = 187.45, MSE = 1.206.
## Variable 2 added with crit = 179.32, MSE = 1.188.
## Variable 12 added with crit = 171.56, MSE = 1.171.
## Variable 11 added with crit = 163.55, MSE = 1.154.
## Variable 1 added with crit = 156.98, MSE = 1.138.
## Variable 15 added with crit = 150.75, MSE = 1.123.
## Variable 10 added with crit = 143.98, MSE = 1.108.
## Variable 470 added with crit = 138.51, MSE = 1.094.
## Variable 17 added with crit = 133.12, MSE = 1.081.
## Variable 13 added with crit = 130.58, MSE = 1.071.
## Variable 375 added with crit = 127.7, MSE = 1.06.
```

```
## Variable 3 added with crit = 125.57, MSE = 1.051.
## Variable 6 added with crit = 122.93, MSE = 1.041.
## Variable 19 added with crit = 120.63, MSE = 1.031.
## Variable 702 added with crit = 118.74, MSE = 1.022.
## Variable 16 added with crit = 116.06, MSE = 1.013.
## Variable 431 added with crit = 115.4, MSE = 1.005.
## Variable 557 added with crit = 114.97, MSE = 0.998.
## Variable 284 added with crit = 114.57, MSE = 0.99.
## Done.
## Starting stepwise, 0 variables, crit = 302.86, MSE = 1.354.
## Variable 2 added with crit = 288.53, MSE = 1.325.
## Variable 11 added with crit = 274.5, MSE = 1.298.
## Variable 17 added with crit = 260.85, MSE = 1.271.
## Variable 10 added with crit = 246.96, MSE = 1.245.
## Variable 12 added with crit = 233.34, MSE = 1.22.
## Variable 1 added with crit = 220.47, MSE = 1.196.
## Variable 15 added with crit = 208.63, MSE = 1.174.
## Variable 13 added with crit = 199.21, MSE = 1.155.
## Variable 3 added with crit = 190.45, MSE = 1.137.
## Variable 6 added with crit = 182.08, MSE = 1.12.
## Variable 19 added with crit = 174.6, MSE = 1.104.
## Variable 5 added with crit = 168.49, MSE = 1.089.
## Variable 16 added with crit = 162.41, MSE = 1.075.
## Variable 14 added with crit = 155.04, MSE = 1.06.
## Variable 470 added with crit = 150.63, MSE = 1.048.
## Variable 18 added with crit = 146.73, MSE = 1.037.
## Variable 20 added with crit = 143.44, MSE = 1.026.
## Variable 4 added with crit = 138.44, MSE = 1.014.
## Variable 9 added with crit = 135.7, MSE = 1.004.
## Variable 7 added with crit = 133.69, MSE = 0.996.
## Variable 8 added with crit = 131.62, MSE = 0.987.
## Variable 702 added with crit = 130.24, MSE = 0.979.
## Variable 375 added with crit = 129.11, MSE = 0.971.
## Variable 431 added with crit = 128.58, MSE = 0.963.
## Variable 827 added with crit = 127.67, MSE = 0.956.
## Done.
```

```
## Starting stepwise, 0 variables, crit = 432.96, MSE = 1.542.
## Variable 2 added with crit = 412.75, MSE = 1.501.
## Variable 11 added with crit = 392, MSE = 1.46.
## Variable 17 added with crit = 370.94, MSE = 1.419.
## Variable 10 added with crit = 350.18, MSE = 1.381.
## Variable 12 added with crit = 331.79, MSE = 1.346.
## Variable 1 added with crit = 311.3, MSE = 1.31.
## Variable 15 added with crit = 293.17, MSE = 1.277.
## Variable 13 added with crit = 277.32, MSE = 1.249.
## Variable 3 added with crit = 261.57, MSE = 1.221.
## Variable 6 added with crit = 247.18, MSE = 1.195.
## Variable 5 added with crit = 234.64, MSE = 1.172.
## Variable 19 added with crit = 221.43, MSE = 1.149.
## Variable 16 added with crit = 210.41, MSE = 1.128.
## Variable 14 added with crit = 195.55, MSE = 1.104.
## Variable 18 added with crit = 184.38, MSE = 1.084.
## Variable 20 added with crit = 173.68, MSE = 1.065.
## Variable 4 added with crit = 161.85, MSE = 1.045.
## Variable 9 added with crit = 151.31, MSE = 1.027.
## Variable 7 added with crit = 143.36, MSE = 1.012.
## Variable 8 added with crit = 134.42, MSE = 0.996.
## Variable 470 added with crit = 131.62, MSE = 0.987.
## Variable 702 added with crit = 130.24, MSE = 0.979.
## Variable 375 added with crit = 129.11, MSE = 0.971.
## Variable 431 added with crit = 128.58, MSE = 0.963.
## Variable 827 added with crit = 127.67, MSE = 0.956.
## Done.
## Starting stepwise, 0 variables, crit = 571.22, MSE = 1.77.
## Variable 2 added with crit = 545.74, MSE = 1.714.
## Variable 11 added with crit = 518.79, MSE = 1.657.
## Variable 17 added with crit = 490.67, MSE = 1.6.
## Variable 10 added with crit = 463.42, MSE = 1.546.
## Variable 15 added with crit = 440.59, MSE = 1.501.
## Variable 12 added with crit = 416.34, MSE = 1.455.
## Variable 1 added with crit = 388.47, MSE = 1.405.
## Variable 13 added with crit = 366.16, MSE = 1.365.
```

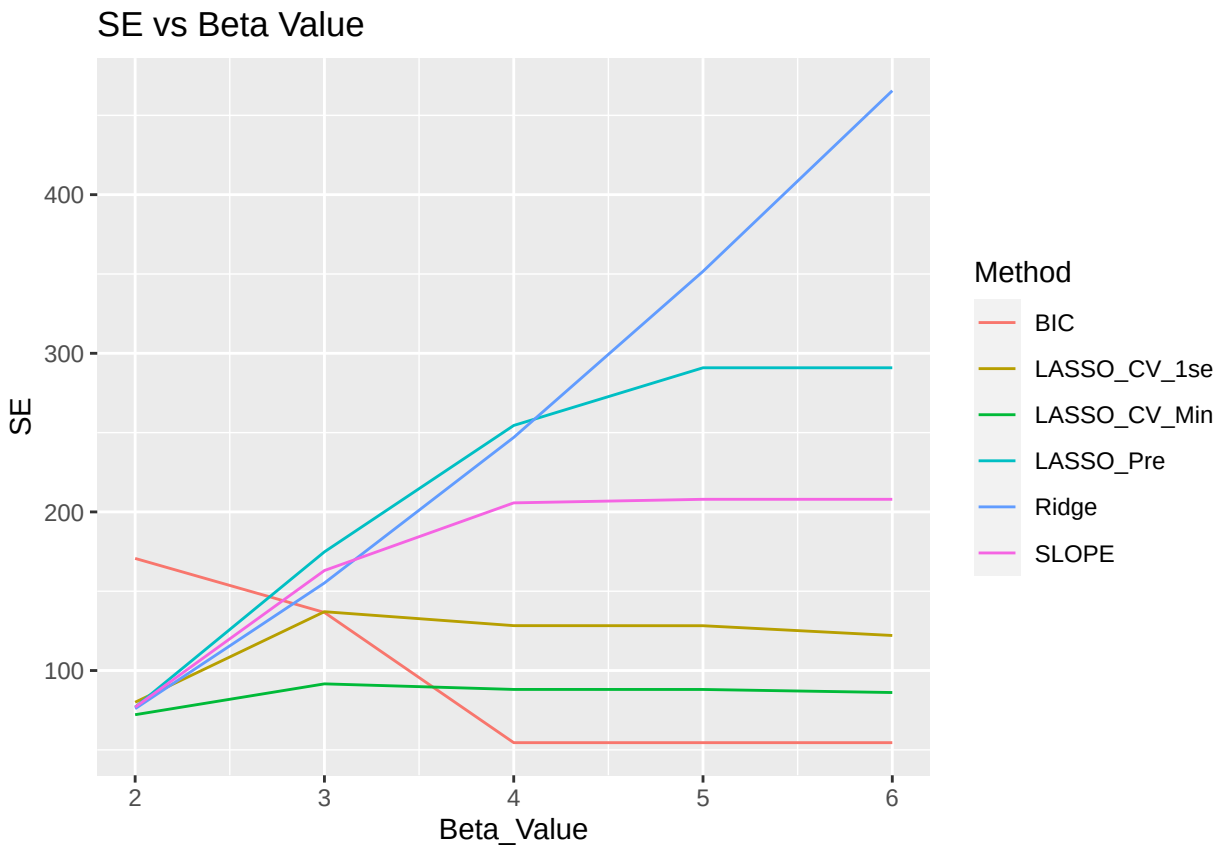
```
## Variable 3 added with crit = 343.15, MSE = 1.324.
## Variable 6 added with crit = 322.58, MSE = 1.289.
## Variable 5 added with crit = 302.59, MSE = 1.254.
## Variable 19 added with crit = 283.87, MSE = 1.223.
## Variable 18 added with crit = 266.68, MSE = 1.193.
## Variable 16 added with crit = 248.73, MSE = 1.164.
## Variable 14 added with crit = 224.77, MSE = 1.129.
## Variable 9 added with crit = 206.36, MSE = 1.101.
## Variable 20 added with crit = 187.91, MSE = 1.073.
## Variable 4 added with crit = 168.33, MSE = 1.045.
## Variable 7 added with crit = 152.34, MSE = 1.021.
## Variable 8 added with crit = 134.42, MSE = 0.996.
## Variable 470 added with crit = 131.62, MSE = 0.987.
## Variable 702 added with crit = 130.24, MSE = 0.979.
## Variable 375 added with crit = 129.11, MSE = 0.971.
## Variable 431 added with crit = 128.58, MSE = 0.963.
## Variable 827 added with crit = 127.67, MSE = 0.956.
## Done.
```

2.1 Get the results and make plots

```
all_results_df <- do.call(rbind, results_list)

# Create plots
plot_SE <- ggplot(all_results_df, aes(x = Beta_Value, y = SE, color = Method)) +
  geom_line() +
  ggtitle("SE vs Beta Value")

plot_SE_X <- ggplot(all_results_df, aes(x = Beta_Value,
                                        y = SE_X, color = Method)) +
  geom_line() +
  ggtitle("SE_X vs Beta Value")

plot_Power <- ggplot(all_results_df, aes(x = Beta_Value,
                                         y = Power, color = Method)) +
  geom_line() +
  ggtitle("Power vs Beta Value")

plot_FDP <- ggplot(all_results_df, aes(x = Beta_Value,
                                       y = FDP, color = Method)) +
  geom_line() +
  ggtitle("FDP vs Beta Value")
```

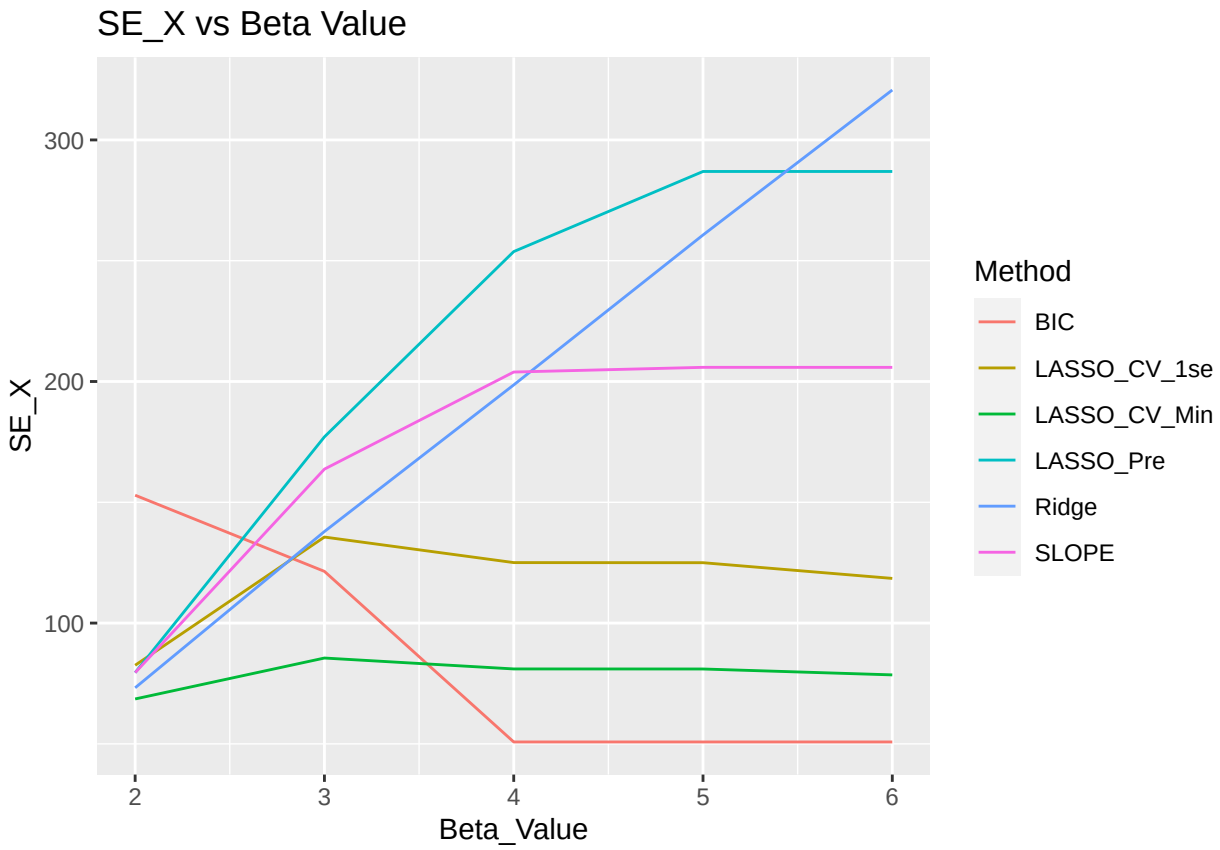
2.2 SE vs Beta Value

```
print(plot_SE)
```



2.3 SE_X vs Beta Value

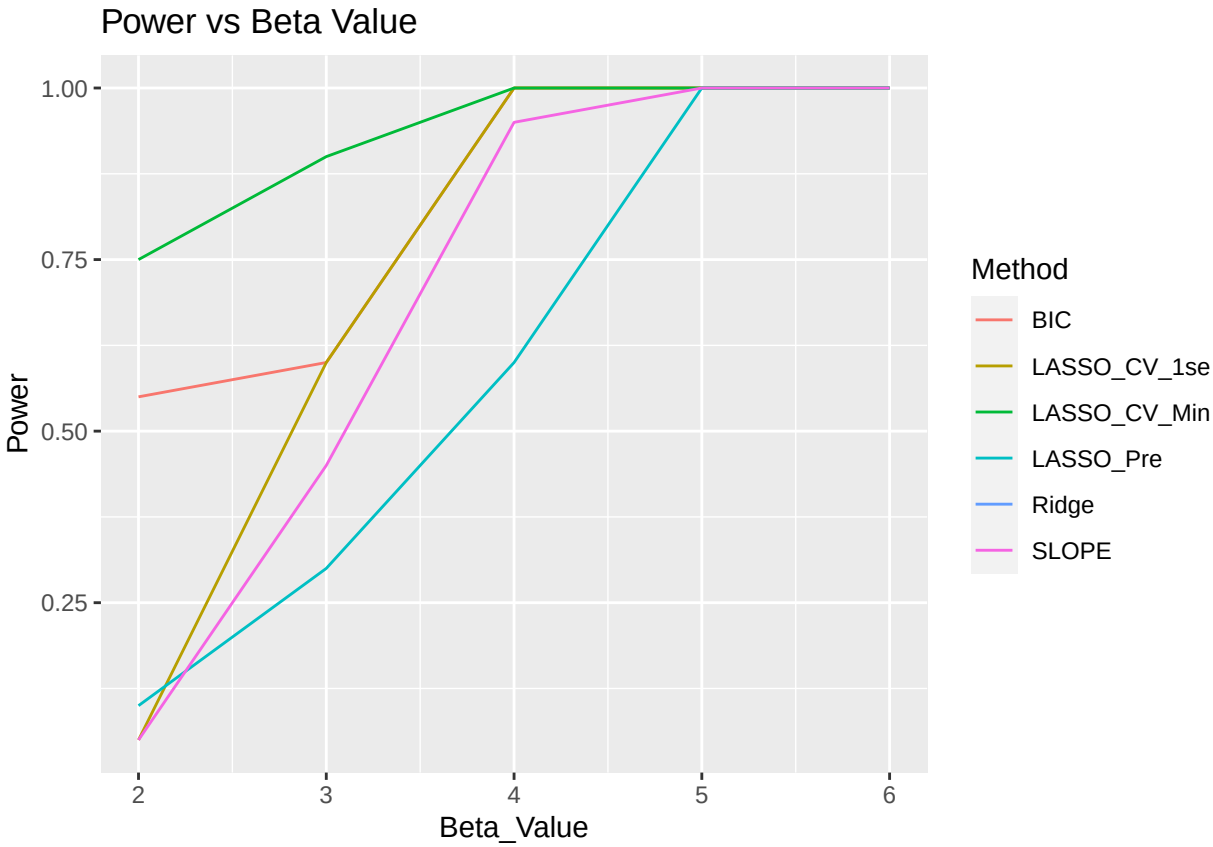
```
print(plot_SE_X)
```



2.4 Power vs Beta Value

```
print(plot_Power)
```

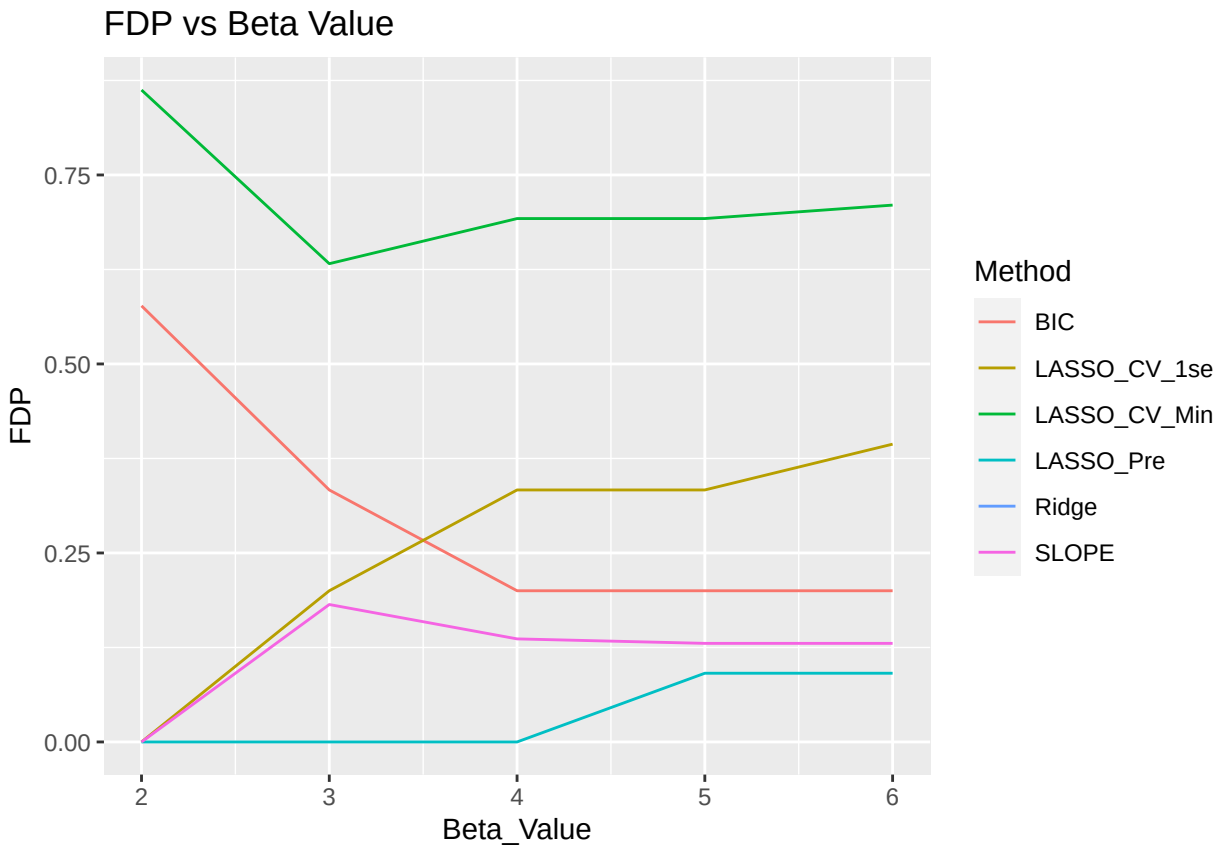
```
## Warning: Removed 5 rows containing missing values (`geom_line()`).
```



2.5 FDP vs Beta Value

```
print(plot_FDP)
```

```
## Warning: Removed 5 rows containing missing values (`geom_line()`).
```


FDP vs Beta Value
FDP
0.75
0.50
0.25
0.00
2
3
4
5
6
Beta_Value
Method
BIC
LASSO_CV_1se
LASSO_CV_Min
LASSO_Pre
Ridge
SLOPE