TITLE: Comparison of Model Selection Using PCR, PLSR, Best subsets, Ridge Regression and Lasso Regression on cystfibr dataset

by Gap Kim

INTRODUCTION

In this report, the performances of five model selection methods, Principal Component regression (PCR), Partial Least Squares regression (PLSR), Best subsets, Ridge regression and Lasso regression, have been compared using the 'cystfibr' dataset from the 'ISwR' library. A Monte Carlo cross validation with sampling size of 100 is used to determine the optimal model that regressed maximum expiratory pressure (pemax) on 9 predictors. First, a detailed analysis was performed for PCR and PLSR. Then, a comparison of the five model selection methods was performed in terms of Test MSE and predictor selection. The PCR, PLSR and Best subsets selection method had the lowest Test MSE. A spectral analysis was used to determine the predictors that had the postive and negative contributions on 'pemax'

EXPLORTORY DATA ANALYSIS

The 'cystfibr' dataset from ISwR library contains a study of 25 patients (14 males and 11 females) aged 7-23 years with cystic fibrosis. The following 10 variables are included in the dataset:

```
age: in years
sex: male(coded 0), female(coded 1)
height: in cm
weight: in kg
bmp: body mass (% of normal)
fev1: forced expiratory volume
rv: residual volume
frc: functional residual capacity
tlc: total lung capacity
pemax: maximum expiratory pressure
```

Upon analysis of the dataset, strong correlations (absolute value of r above 0.7) are observed for age~height, age~weight, rv~frc, and tlc~frc. Figure 1 shows the relationship among age, height and weight by sex where a strong correlation was found between height and age for both male and female. The rate of increse in height and weight are similar for both males and females up to the age of about 12 after which the growth rate of male is higher than female.

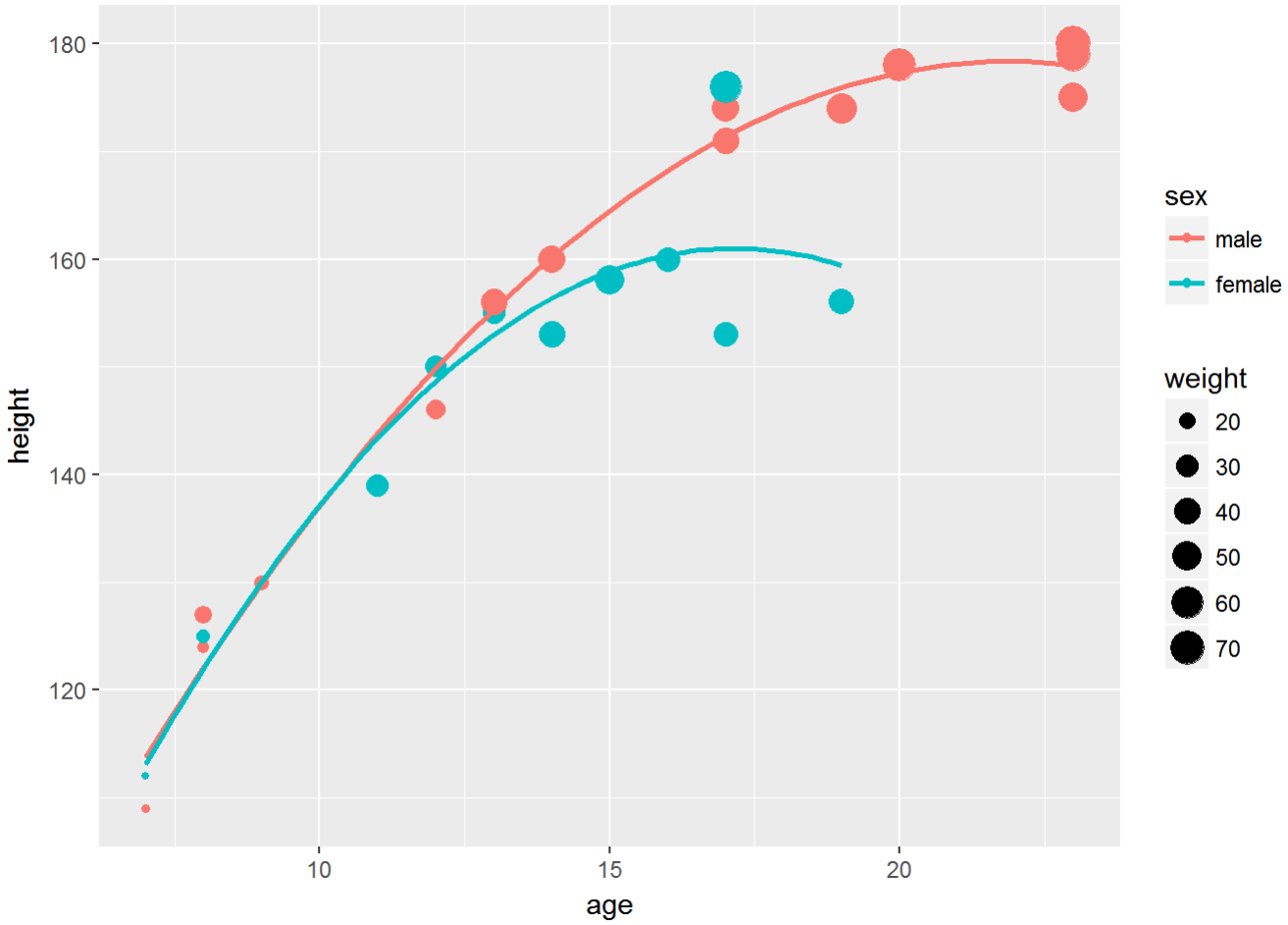

Figure 1: Relationship among age, height and weight by sex

ANALYSIS

A detaied analysis was performed on Principal Component Regression (PCR) and Partial Least Squares Regression (PLSR) using Monte Carlo cross validation (CV). In addition, Best subsets, Ridge regression, and Lasso regression were reevaluated with Monte Carlo CV to be consistent in the comparison.

Principal Component Regression (PCR)

A Monte Carlo CV has been employed to acquire the optimal number of components and to estimate the prediction accuracy of the PCR. The number of training samples has been chosen according to:

$$m = 3\sqrt{n} - 1$$

With only 25 observations (n=25), the size of training set is 14 and testing set is 11. The Monte Carlo CV is performed with sampling size of 100, and the results are summarized in Figure 2. The optimal number of component for the PCR was 1 since the sigle number of component came out most frequently at 31% of times.

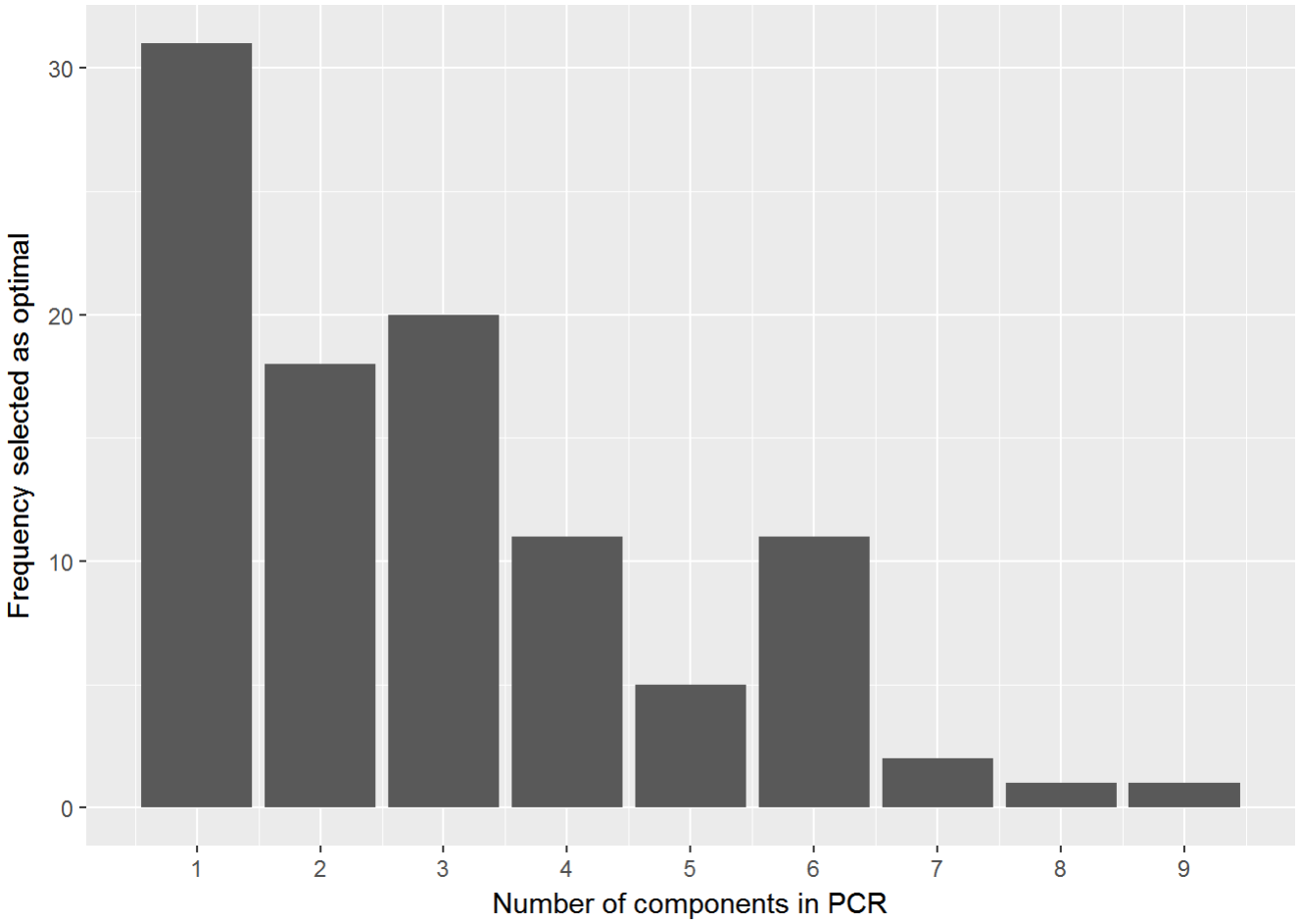

Figure 2: Frequency of number of component selected as optimal by PCR with the training dataset (Monte Carlo CV sampling size = 100)

The cross validated MSE values from the training set are plotted in Figure 3. The red line is the mean MSE for the respective number of components in PCR. The mean MSEs of number of components 1, 2 and 3 are the lowest with smallest variation. As more components are added after the third component, the mean MSEs and variation increases quickly, which indicates that the model variance is excessive and overfitting is occurring.

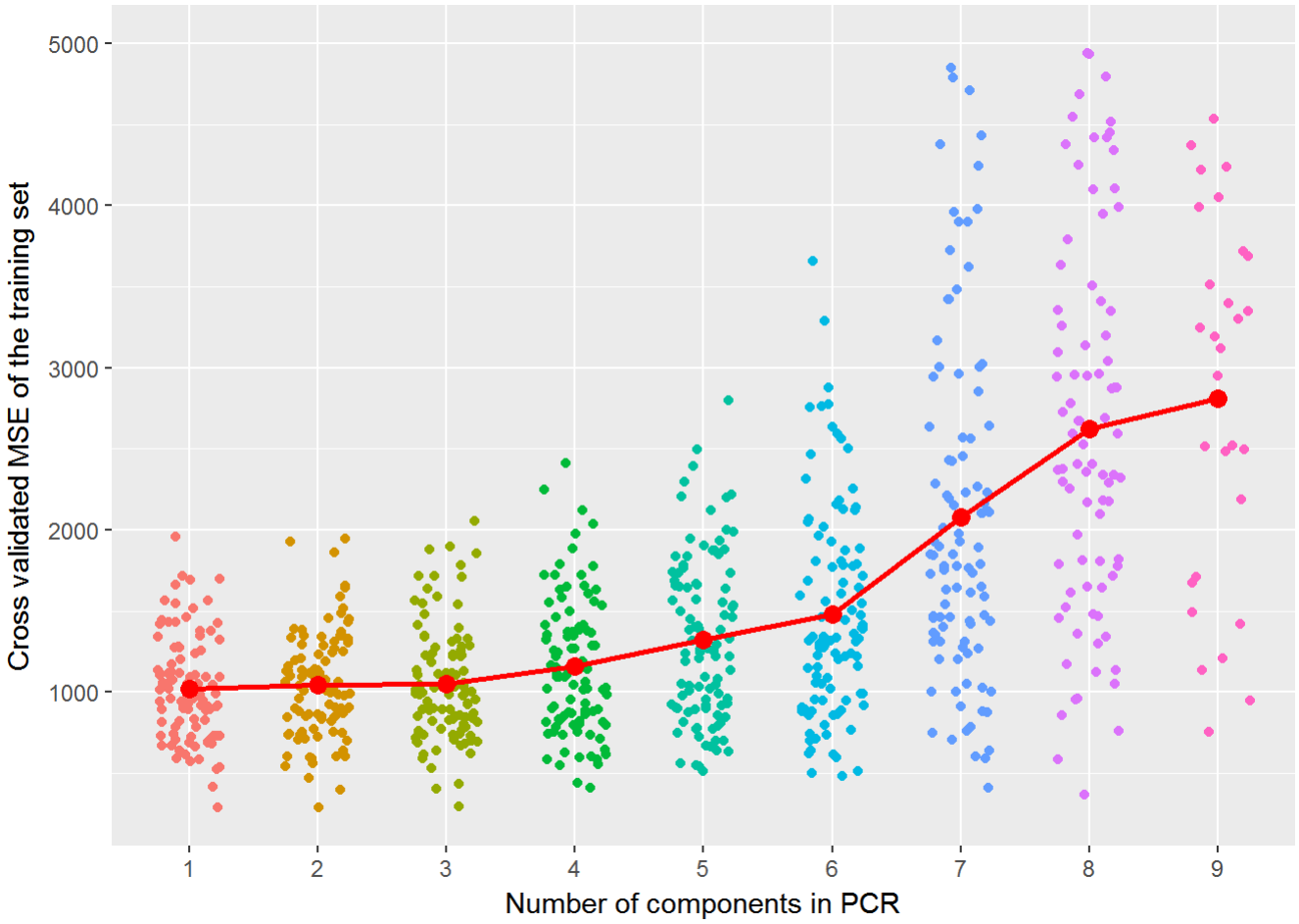

Figure 3: MSE of PCR using the training set with Monte Carlo CV (sampling size=100)

The prediction accuracy, measured as MSE, of the single component PCR using the test dataset is:

```
Prediction MSE = 1034 (where optimal number of PCR component is 1 )
```

Fitting the PCR model with the full dataset, we can observe that the first principal component explains 57.59% of the variation in the predictors and explains 31.16% of variation in the response variable ‘pemax’.

```
Data:   X dimension: 25 9
    Y dimension: 25 1
Fit method: svdpc
Number of components considered: 1
TRAINING: % variance explained
       1 comps
X        57.59
pemax    31.16
```

Partial Linear Squares Regression (PLSR)

With same seeding from PCR, Monte Carlo CV with sampling size of 100 was performed to obtain the optimal number of components and to estimate the prediction error of the PLSR. The results are shown in Figure 4 where the most frequent number of component is 1 at 63% of times.

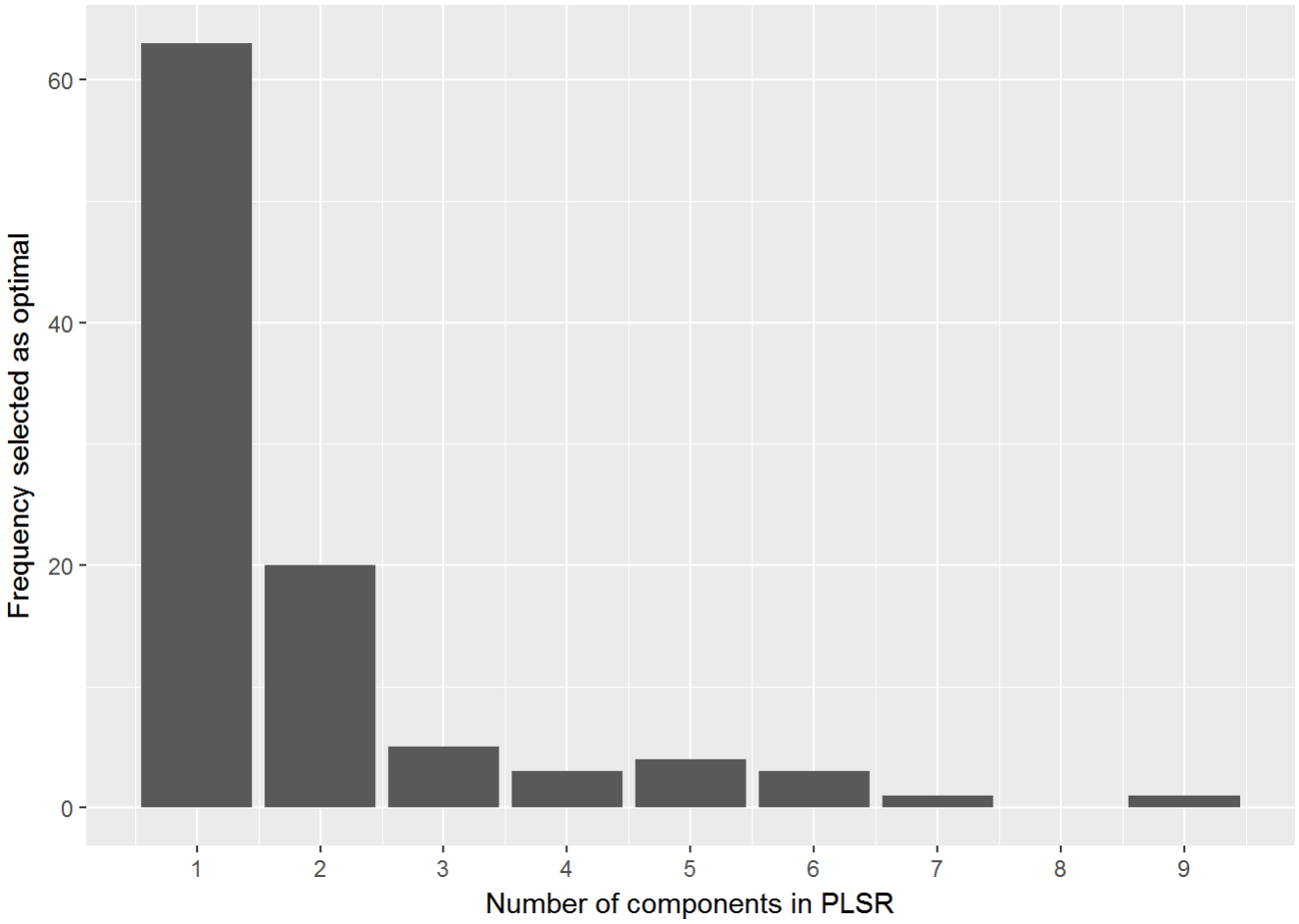

Figure 4: Frequency of number of component selected as optimal by PLSR with the training dataset (Monte Carlo CV sampling size = 100)

The cross validated MSE values from the training set is plotted in Figure 5 where the red line is the mean MSE for the respective PLSR number of component. The MSEs quickly increase as components are added to the PLSR model indicating increase in model variance and overfitting.

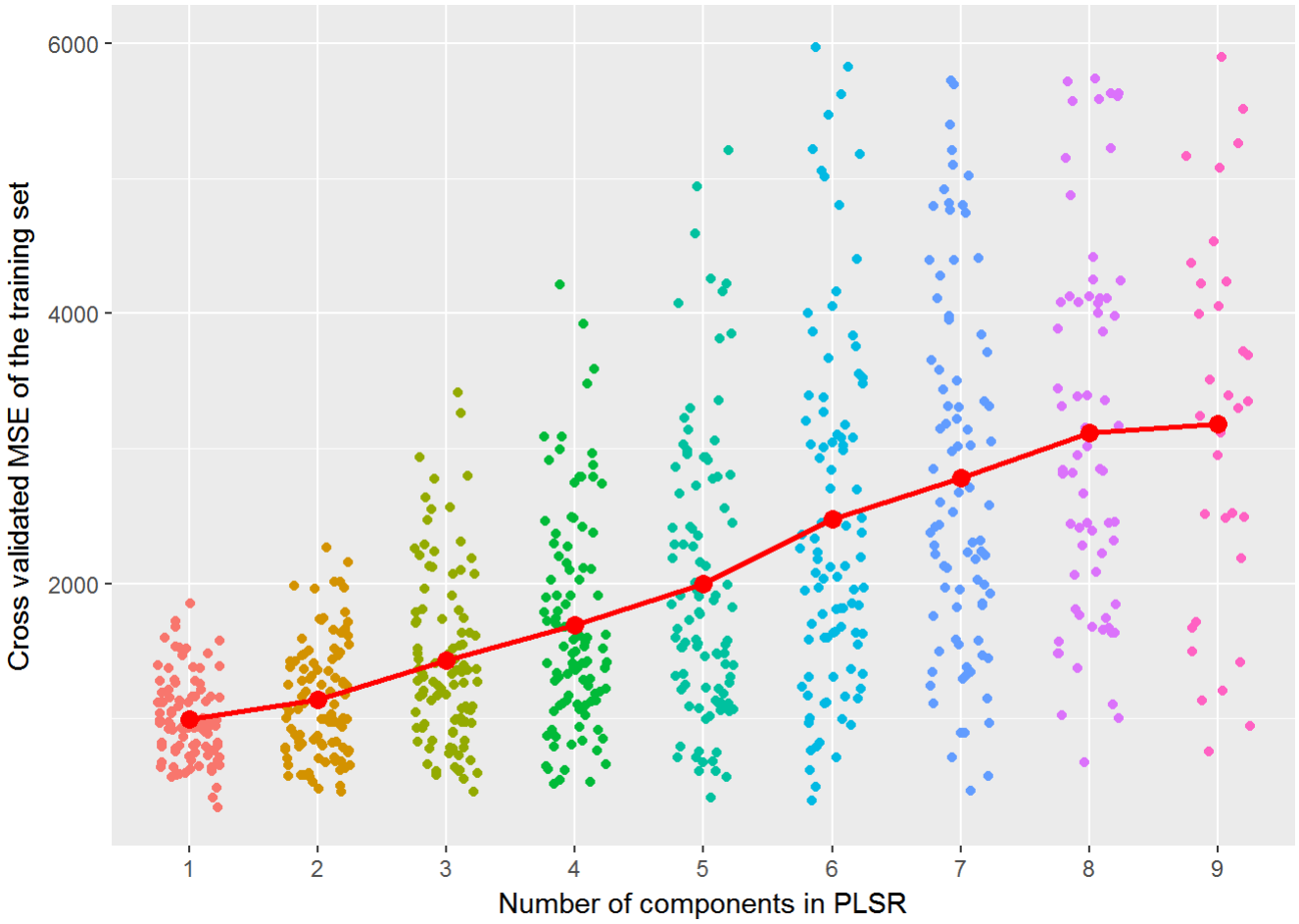

Figure 5: PLSR MSE of the training set from Monte Carlo CV simulation (sampling size=100)

The prediction accuracy, measured as MSE, of the single component PLSR using the test dataset is:

```
Prediction MSE = 974.6 (where optimal number of PLSR component is 1 )
```

Finally, the PLSR model is fitted with full dataset. With a single PLSR component, 56.79% of the variation in predictors is explained, and 37.49% of variation in response variable ‘pemax’ is explained.

It is noted that the percentage of explained variation in predictors are slightly lower for the PLSR in comparison with PCR. The reason for this is due to the fact that PCR is an unsupervised learner which finds principal components that explains the most amount of variation in only the predictors. On the other hand, explained variation in response variable is slightly higher for PLSR as the PLSR is supervised learner that takes into account the reponse variable.

```
Data:   X dimension: 25 9
    Y dimension: 25 1
Fit method: kernelpls
Number of components considered: 1
TRAINING: % variance explained
       1 comps
X        56.79
pemax    37.49
```

Monte Carlo CV for Best Subsets, Ridge and Lasso Regression

To be consistent in the comparison, the Monte Carlo CV was performed on Best subsets, Ridge and Lasso regression in this study where n-fold CV was employed previously.

Best Subsets

For the Best subsets, Monte Carlo CV was used to determine the optimal number of predictors as shown in Figure 6. As expected, the training MSE continually decreased with incresing number of predictors. The test MSE, however, increased as the predictors are added, which indicates the optimal number of predictor is 1 for the Best subsets.

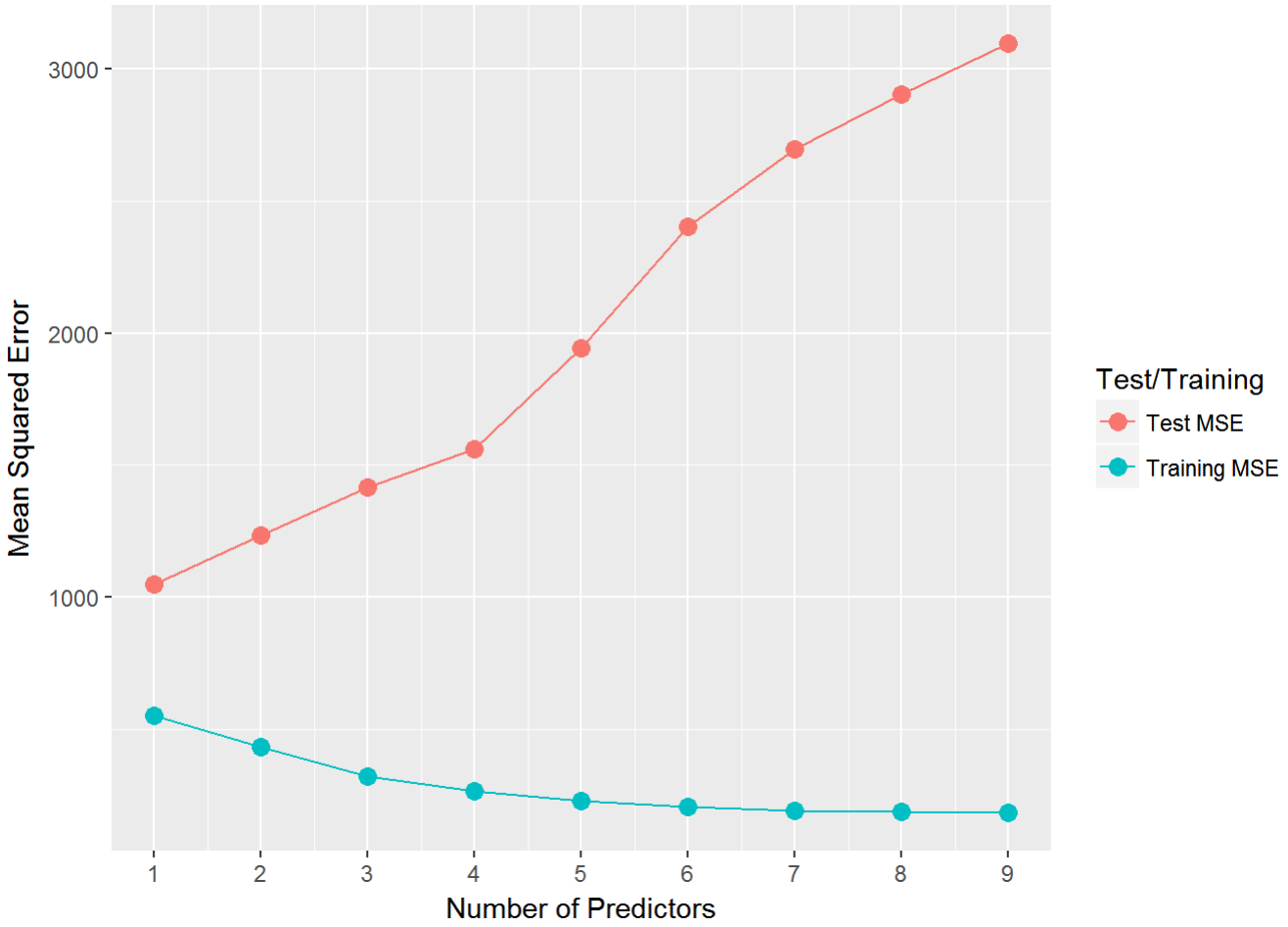

Figure 6: Test and training MSE using Best subsets regression with Monte Carlo simulation (100 samplings)

Using the full dataset with optimal predictor number of 1, only 'weight' is included in the Best subsets model.

```
(Intercept)      weight
     63.546       1.187
```

Ridge Regression

Similarly, the Monte Carlo CV with sampling size of 100 was applied to Ridge regression to determine the optimal λ_{min}:

```
Optimal lambda min = 205.3
```

Then the Ridge regression model was fitted using the full dataset, which resulted in coefficients versus $log(\lambda)$ as shown in Figure 7. As the model bias increased with increasing λ, the coefficients eventually became 0. For the chosen λ_{min}, however, all 9 coefficients are non-zero.

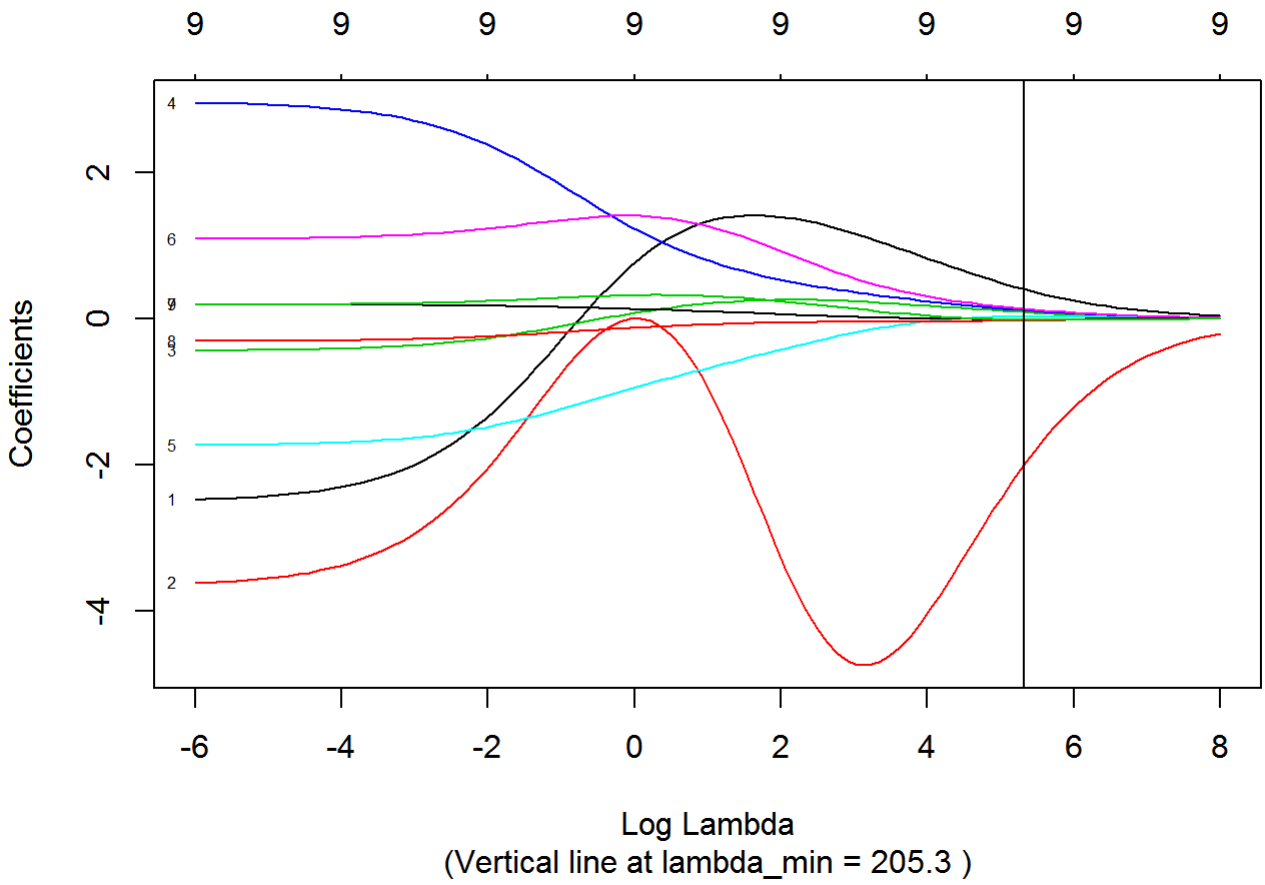

Figure 7: Coefficients vs. log(lambda) for Ridge Regression with full dataset

Lasso Regression

The Monte Carlo CV with sampling size of 100 was applied to Lasso regression to determine the optimal λ_{min}:

```
Optimal lambda min = 13.46
```

The Lasso regression was fitted using the full dataset, and the resulting coefficients versus $log(\lambda)$ is ploted in Figure 8. As the model bias increases, some coefficients rapidly decay. For the chosen λ_{min}, only single predictor, ‘weight’, is non-zero.

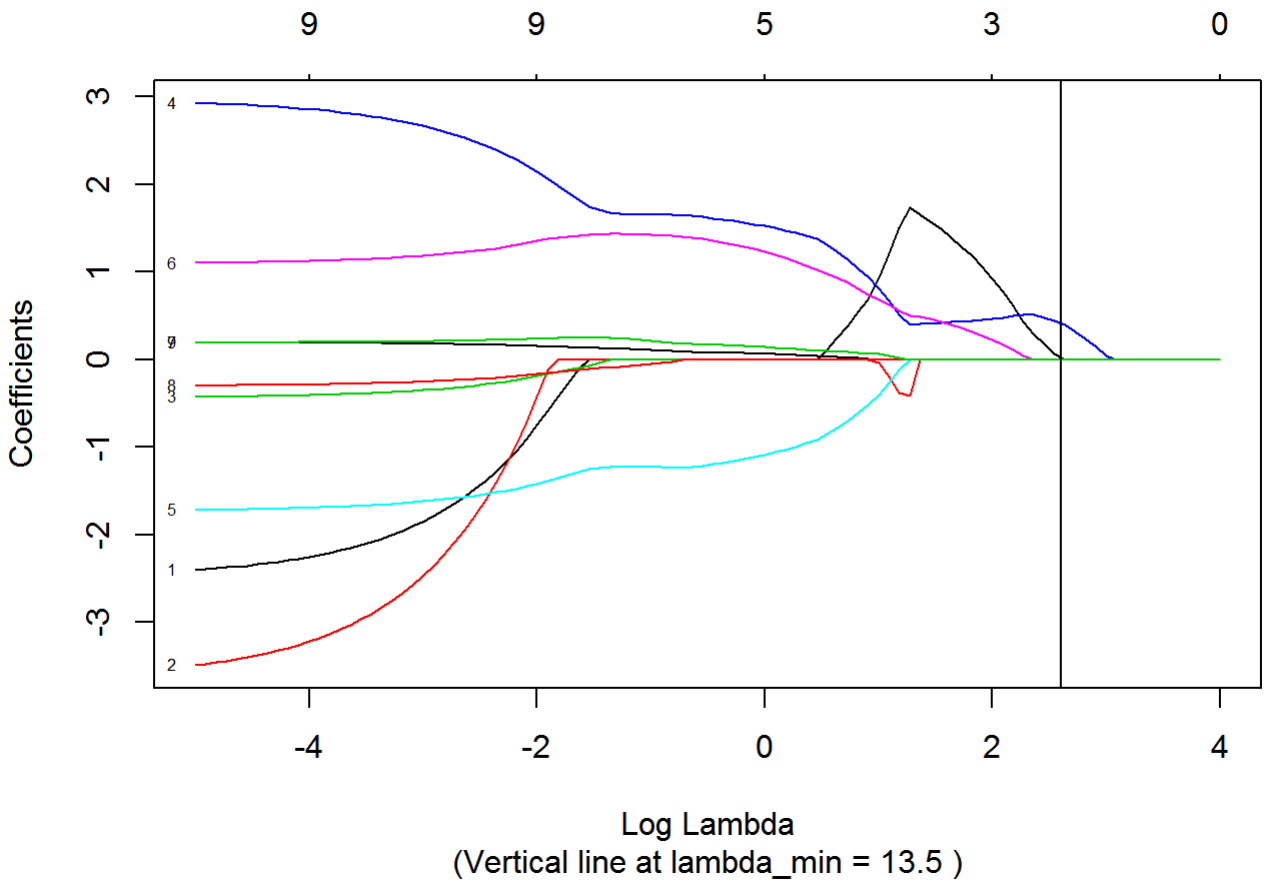

Figure 8: Coefficients vs. log(lambda) for Ridge Regression with full dataset

DISCUSSION AND COMPARATIVE TABLES AND PLOTS

The Test MSEs of all the model selection methods are summarized in Figure 9. The solid dots represent the means of Test MSEs, and the error bars represent the lower and higher quartiles of the repective methods. The mean Test MSE is the lowest for PLSR at 974.6. However, the PCR, PLSR and BS have comparable prediction accuracy considerting the variations involved by the error bars. The Test MSEs of Ridge and Lasso regressions have a little higher mean and larger variation than the other three methods.

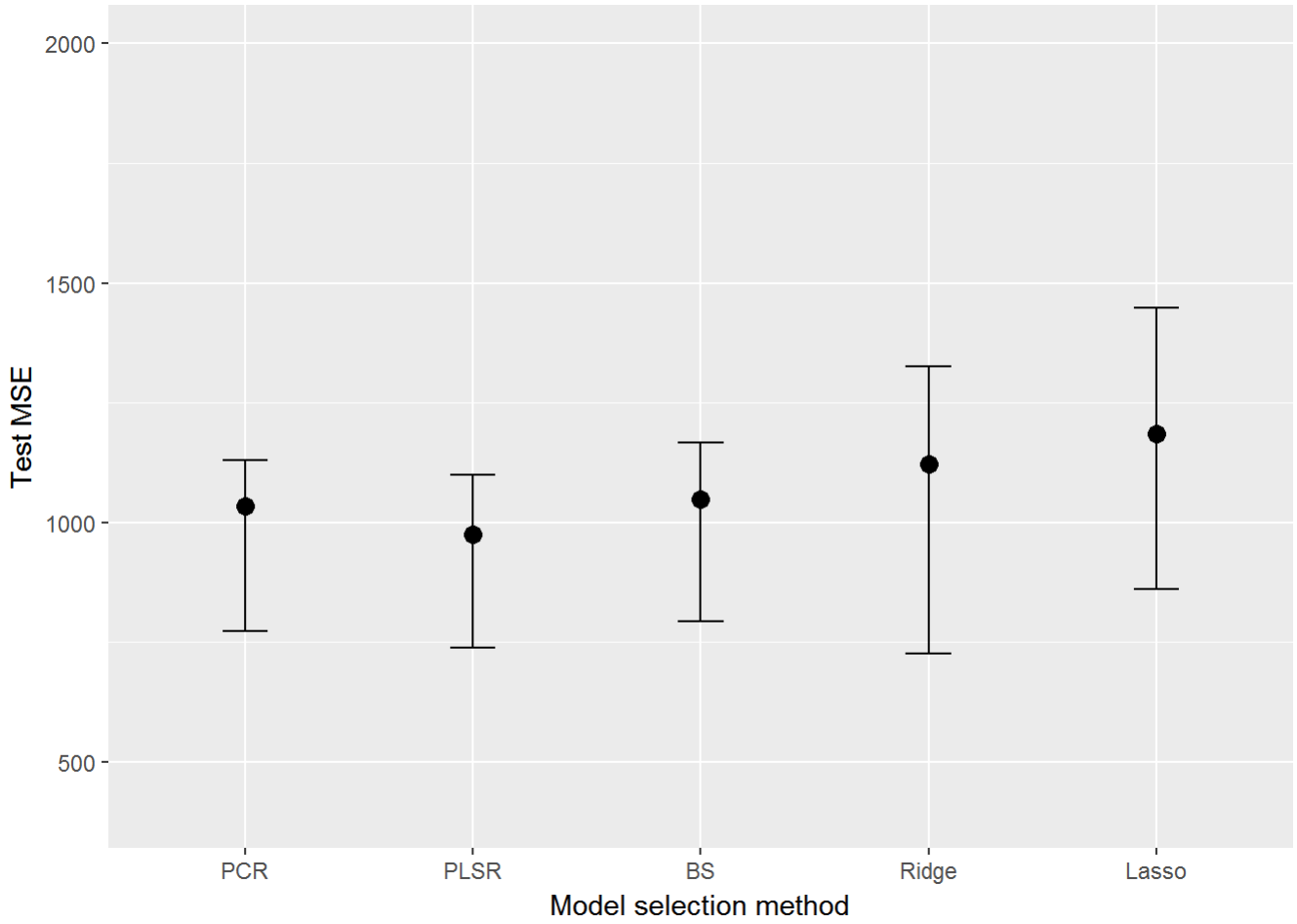

Figure 9: Comparison of Test MSE of different model selection (dot: mean, error bars: 1 and 3 quartile)

The coefficients and their values selected by Best subsets, Ridge, and Lasso regressions are summarized in Table 1. Both Best subsets and Lasso regressions selected 'weight' as the single coefficient for the model whereas Ridge regression required all 9 predictors.

Table 1: Number of coefficients and values selected by the Best subsets, Ridge regression and Lasso regression

	BS	Ridge	Lasso
(Intercept)	63.546	85.4812	93.0115
age	NA	0.4053	NA
sex	NA	-1.9968	NA
height	NA	0.0913	NA
weight	1.187	0.1177	0.4194
bmp	NA	0.0315	NA
fev1	NA	0.1355	NA
rv	NA	-0.0070	NA
frc	NA	-0.0254	NA
tlc	NA	-0.0110	NA
Num.Comp	1.000	9.0000	1.0000

Both the PCR and PLSR selected a single component for the model reducing the dimension of the problem from 9 to 1. It is, however, noted that both methods do not select predictors like Best subsets or Lasso regression, but still

require all 9 predictors for the chosen single component as shown in Table 2. Looking at the loadings for PCR and PLSR in Table 2, the predictors ‘age’, ‘height’, and ‘weight’ have similar amount of positive contribution to ‘pemax’. On the other hand, ‘rv’ and ‘frc’ have similar amount of negative impact on ‘pemax’.

Table 2: Loadings of the first component for PCR and PLSR

	PCR-Comp1	PLSR-Comp1
age	0.3620	0.3983
sex	-0.1385	-0.1648
height	0.3676	0.4031
weight	0.3893	0.4214
bmp	0.2972	0.2936
fev1	0.3047	0.3067
rv	-0.3804	-0.3752
frc	-0.3859	-0.3837
tlc	-0.2973	-0.2794

A spectral component plot has been generated to compare the contribution of each predictor on the response as shown in Figure 10. For each method, the loading was calculated as the normalized coefficients, such that sum of squared coefficients becomes 1.

It can be readily seen that ‘age’, ‘height’, ‘weight’, ‘bmp’ and ‘fev1’ have postive impact on ‘pemax’ whereas ‘sex’, ‘rv’, ‘frc’ and ‘tlc’ have negative impact on ‘pemax’. The two high peaks of 1 in ‘weight’ came from the Best subsets and Lasso regression where only a single component was selected. It is also interesting to note that Ridge regression had a very high peak for ‘sex’ in comparison to other models.

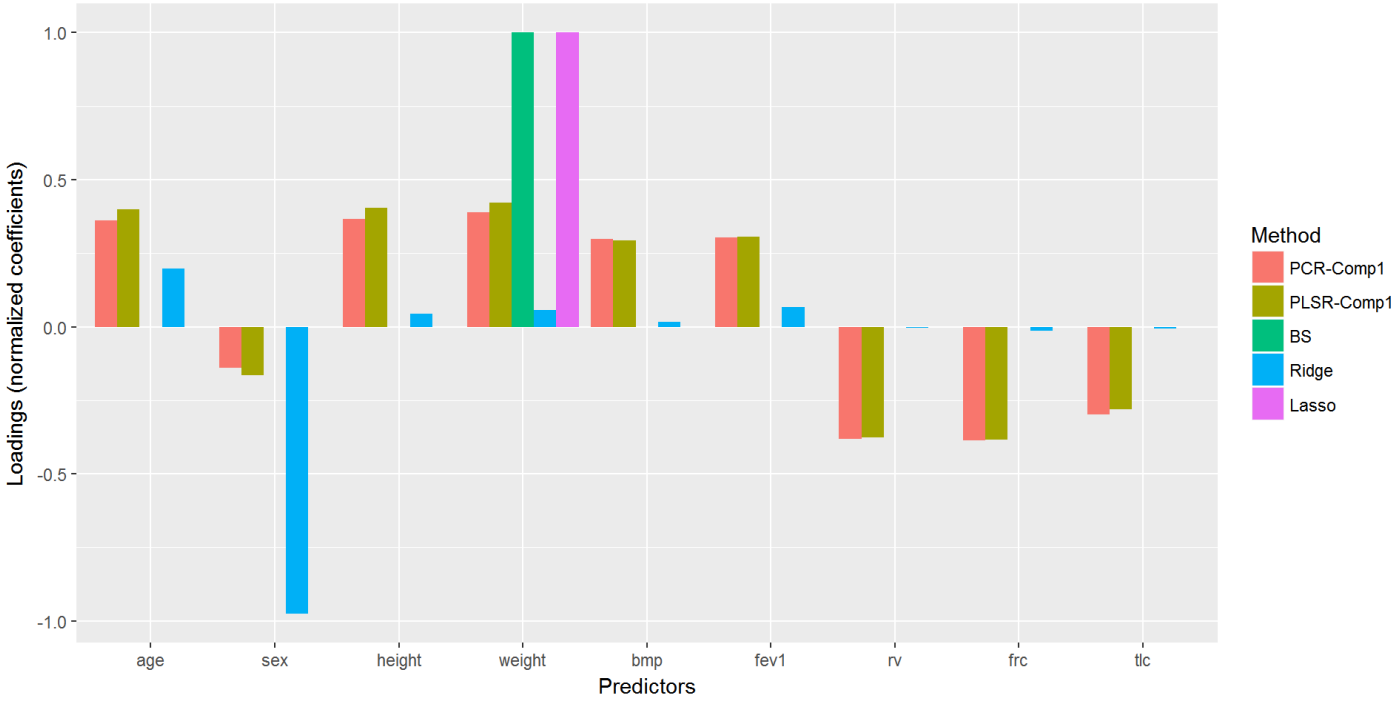

Figure 10: Spectral plot of coefficients (loadings) for different seletion methods

A fitted versus observed values of ‘pemax’ is plotted for all model selection methods in Figure 11 using the full dataset. A perfectly fitted values will fall on the line with slop of 1 shown as the dotted line in Figure 11. The linearly fitted slopes of Ridge and Lasso regressions deviate the most from the perfect slope of 1. On the other hand, slopes of Best subsets and PLSR are closer to the slope of 1, which indicates a better fit.

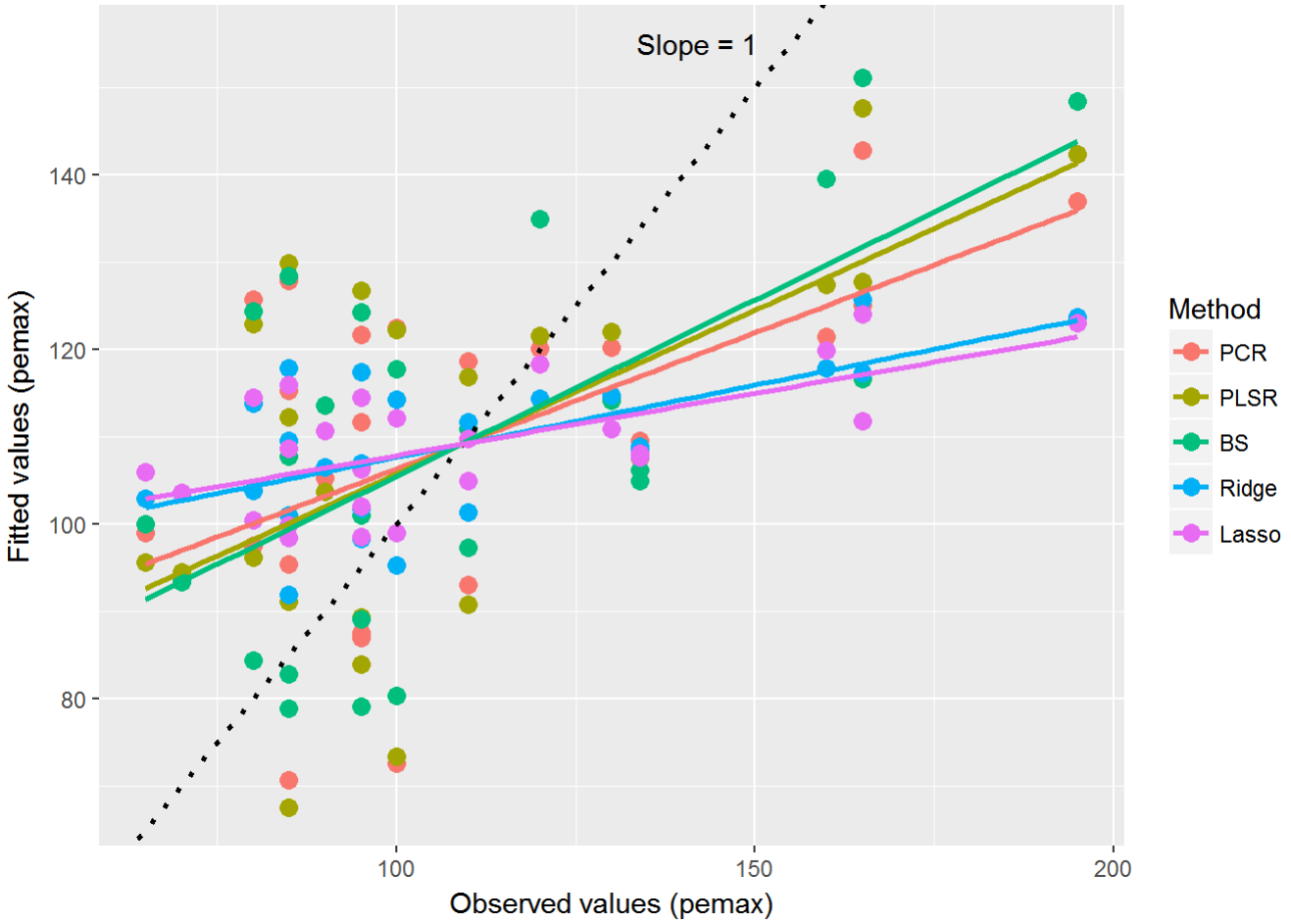

Figure 11: Comparison of fitted vs. observed values

The R^2 values of the model selection methods using the full dataset is given in Figure 12. The results are inline with fitted versus observed values plot in Figure 11. It is interesting to note the difference between the Best subsets and Lasso regression where both have only a single component ‘weight’.

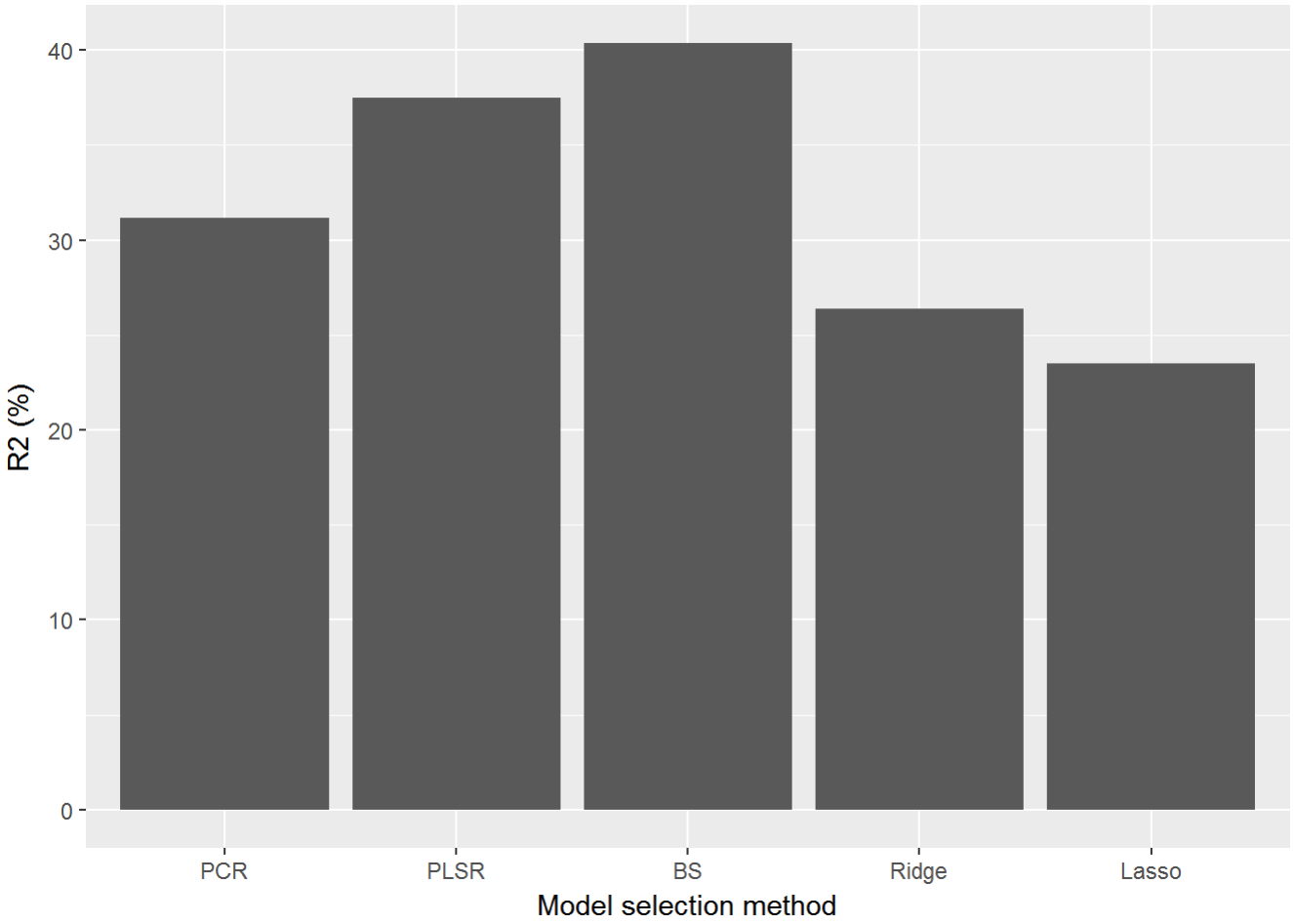

Figure 12: R2 values of the model selection method with the training dataset

It needs to be emphasized that Figures 11 and 12 do not truly represent the predictive performace of the models whereas the Test MSE given in Figure 9 provides a more accurate predictive performance measure.

CONCLUSIONS

A comparative study on five model selection methods, Principal Component regression, Partial Least Squares regression, Best subsets, Ridge regression and Lasso regression, have been performed on ‘cystfibr’ dataset from ‘ISwR’ library. The goal is to find relationship between reponse variable, ‘pemax’, and other 9 predictors. A Monte Carlo cross validation was used on all five selection methods. Following conclusions may be drawn from this study:

- Both the PCR and PLSR selected a single component as the optimal model. The Test MSEs of PCR and PLSR were 1034 and 974.6, respectively. The variation of ‘pemax’ explained by the PCR and PLSR were 31.16% and 37.49%, respectively.
- The PCR, PLSR and Best subsets selection method had the lowest Test MSE, and thus, showed better predictive capability than Ridge and Lasso regression.
- The predictors, ‘age’, ‘height’, ‘weight’, ‘bmp’ and ‘fev1’ have postive impact on the response ‘pemax’; and ‘sex’, ‘rv’, ‘frc’ and ‘tlc’ have negative impact on the ‘pemax’.
- If the predictive accuracy is critical, PLSR is recommended as it had the lowest Test MSE. However, PLSR requires measurements of all 9 predictors.
- If convenience of prediction is important, Best subsets model is recommended. It has comparable prediction accuracy with PLSR but only requires a single measurement of ‘weight’.

APPENDIX

All R codes used in producing the results are included below:

```
################################
### Initial Setup

knitr::opts_chunk$set(comment=NA, echo=FALSE, warning=FALSE, message=FALSE,
                      fig.align="center")
options(digits=4)

rm(list=ls())

################################
### Get Data

library(ISwR)
data("cystfibr")

cat("\nNumber of NA's: ", sum(is.na(cystfibr)),"\n\n")

mydata = cystfibr
mydata$sex = factor(mydata$sex, levels=c(0,1), labels=c("male","female"))
str(mydata)

####################################
### EXPLORATORY DATA ANALYSIS
####################################

library(ggplot2)

# Figure 1: Relationship among age, height and weight by sex

ggplot(data=mydata) +
   geom_point(aes(x=age, y=height, size=weight, color=sex)) +
   geom_smooth(aes(x=age, y=height, color=sex), span=10, se=FALSE)

####################################
### ANALYSIS
####################################

####################################
### Principal Component Regression
####################################

# Perform Monte Carlo simulation to obtain distribution of ncomp that minimize MSEP

library(pls)
x = model.matrix(pemax ~ ., data=cystfibr)[,-1]
y = cystfibr$pemax

n_Monte = 100
ncomponent = rep(0, n_Monte)
pred.MSE_pcr = rep(NA, n_Monte)
MSEP.trainCV = matrix(NA, n_Monte, 9)

set.seed(3)
```

```
for(i in c(1:n_Monte)) {

   #Sampling for Monte Carlo simulation
   train = sample(c(1:nrow(cystfibr)), round(3*sqrt(nrow(cystfibr))-1))
   test = -train
   y.test = y[test]

   # PCR 5 fold CV on training dataset
   pcr.fit.train = pcr(pemax ~ ., data=cystfibr, subset=train,
                       scale=TRUE, validation="CV", segments=5)

   MSEP.train = MSEP(pcr.fit.train)
   ncomp.min = which.min(MSEP.train$val[1,,-1])

   MSEP.trainCV[i,] = MSEP.train$val[1,,-1] # MSE of CV error for each loop i
   ncomponent[i] = ncomp.min  # Optimal (lowest) number of components per loop i

   pcr.pred = predict(pcr.fit.train, x[test,], ncomp=ncomp.min)
   pred.MSE_pcr[i] = mean((pcr.pred - y.test)^2)  # Predicted MSE using test dataset per i
}

############################################################
### Get optimal number of components from training set

library(ggplot2)

freq.ncomp = table(ncomponent)

# Figure 2: Frequency of number of component selected as optimal from training dataset (100 Mont
e Carlo simulations)

ggplot() +
   geom_bar(aes(x=ncomponent)) +
   scale_x_continuous(labels=c(1:9), breaks=c(1:9)) +
   labs(x="Number of components in PCR", y="Frequency selected as optimal")

################################
### Get MSEP of train dataset

library(reshape2)

# Mean of MSE CV in the training dataset
mean.MSEP.trainCV = apply(MSEP.trainCV, 2, mean)

# CV MSE of the training dataset
MSEP.trainCV_df = data.frame(MSEP.trainCV)
colnames(MSEP.trainCV_df) = c(1:9)

melt_MSEP.trainCV_df = melt(MSEP.trainCV_df, variable.name="ncomp", value.name="MSEP")

# Checkpoint (hidden) -------
freq.ncomp
mean.MSEP.trainCV
MSEP.trainCV_df
```

```
melt_MSEP.trainCV_df
#---------------------------

# Figure 3: PCR MSE of the training set from Monte Carlo CV simulation (sampling size=100)

ggplot(data=melt_MSEP.trainCV_df) +
#   geom_boxplot(aes(x=ncomp, y=MSEP)) +
   geom_jitter(aes(x=ncomp, y=MSEP, color=ncomp), width=0.25) +
   scale_y_continuous(limits=c(NA,5000)) +
   stat_summary(aes(x=ncomp, y=MSEP, group=1),
                fun.y="mean", color="red", size=1, geom="line") +
   stat_summary(aes(x=ncomp, y=MSEP, group=1),
                fun.y="mean", color="red", size=3, geom="point") +
   scale_color_discrete(name="", breaks="") +
   labs(x="Number of components in PCR", y="Cross validated MSE of the training set")

################################################
### Calculate prediction MSE using test dataset

index.max = which.max(freq.ncomp)  # Get the index of most frequent ncomp
id.pcr = as.integer(names(freq.ncomp)[index.max])  # Get optimal number of component

pred.pcr = pred.MSE_pcr[which(ncomponent == id.pcr)]
avg.pred.pcr = mean(pred.MSE_pcr[which(ncomponent == id.pcr)])

cat("Prediction MSE =", avg.pred.pcr,
    "(where optimal number of PCR component is", id.pcr, ")")

index.max
names(freq.ncomp)[index.max]
length(pred.MSE_pcr)
length(pred.MSE_pcr[which(ncomponent == id.pcr)])
length(pred.MSE_pcr[which(ncomponent == names(freq.ncomp)[index.max])])
freq.ncomp

pcr.fit.full = pcr(pemax ~ ., data=cystfibr, scale=TRUE, ncomp=id.pcr)
summary(pcr.fit.full)

##########################################
### Partial Linear Squares Regression
##########################################

# Perform Monte Carlo simulation to obtain distribution of ncomp that minimize MSEP

ncomponent = rep(0, n_Monte)
pred.MSE_plsr = rep(NA, n_Monte)
MSEP.trainCV = matrix(NA, n_Monte, 9)

set.seed(3)
for(i in c(1:n_Monte)) {

   #Sampling for Monte Carlo simulation
   train = sample(c(1:nrow(cystfibr)), round(3*sqrt(nrow(cystfibr))-1))
   test = -train
```

```
   y.test = y[test]

   # plsr 5 fold CV on training dataset
   plsr.fit.train = plsr(pemax ~ ., data=cystfibr, subset=train,
                         scale=TRUE, validation="CV", segments=5)

   MSEP.train = MSEP(plsr.fit.train)
   ncomp.min = which.min(MSEP.train$val[1,,-1])

   MSEP.trainCV[i,] = MSEP.train$val[1,,-1] # MSE of CV error for each loop i
   ncomponent[i] = ncomp.min  # Optimal (lowest) number of components per loop i

   plsr.pred = predict(plsr.fit.train, x[test,], ncomp=ncomp.min)
   pred.MSE_plsr[i] = mean((plsr.pred - y.test)^2)  # Predicted MSE
}

############################################################
### Get optimal number of components from training set

freq.ncomp = table(ncomponent)

# Figure 4: Frequency of number of component selected as optimal from training dataset using PLS
R (100 Monte Carlo simulations)

ggplot() +
   geom_bar(aes(x=ncomponent)) +
   scale_x_continuous(labels=c(1:9), breaks=c(1:9)) +
   labs(x="Number of components in PLSR", y="Frequency selected as optimal")

################################
### Get MSEP of train dataset

# Mean of MSE CV in the training dataset
mean.MSEP.trainCV = apply(MSEP.trainCV, 2, mean)

# CV MSE of the training dataset
MSEP.trainCV_df = data.frame(MSEP.trainCV)
colnames(MSEP.trainCV_df) = c(1:9)

melt_MSEP.trainCV_df = melt(MSEP.trainCV_df, variable.name="ncomp", value.name="MSEP")

# Checkpoint (hidden) -------
freq.ncomp
mean.MSEP.trainCV
MSEP.trainCV_df
melt_MSEP.trainCV_df
#---------------------------

# Figure 5: PLSR MSE of the training set from Monte Carlo CV simulation (sampling size=100)

ggplot(data=melt_MSEP.trainCV_df) +
#   geom_boxplot(aes(x=ncomp, y=MSEP)) +
   geom_jitter(aes(x=ncomp, y=MSEP, color=ncomp), width=0.25) +
   scale_y_continuous(limits=c(NA,6000)) +
```

```
   stat_summary(aes(x=ncomp, y=MSEP, group=1),
                fun.y="mean", color="red", size=1, geom="line") +
   stat_summary(aes(x=ncomp, y=MSEP, group=1),
                fun.y="mean", color="red", size=3, geom="point") +
   scale_color_discrete(name="", breaks="") +
   labs(x="Number of components in PLSR", y="Cross validated MSE of the training set")

##############################################
### Calculate prediction MSE using test dataset

index.max = which.max(freq.ncomp)  # Get the index of most frequent ncomp
id.plsr = as.integer(names(freq.ncomp)[index.max])  # Get optimal number of component

pred.plsr = pred.MSE_plsr[which(ncomponent == id.plsr)]
avg.pred.plsr = mean(pred.MSE_plsr[which(ncomponent == id.plsr)])

cat("Prediction MSE =", avg.pred.plsr,
    "(where optimal number of PLSR component is", id.plsr, ")")

plsr.fit.full = plsr(pemax ~ ., data=cystfibr, scale=TRUE, ncomp=id.plsr)
summary(plsr.fit.full)

##################################################
### Best Subsets Regression (with Monte Carlo CV)
##################################################

library(leaps)

npred = ncol(cystfibr)-1  # Number of predictors

cv.errors = matrix(NA, n_Monte, npred)
train.errors = matrix(NA, n_Monte, npred)

set.seed(3)
for(j in c(1:n_Monte)) {

   #Sampling for Monte Carlo simulation
   train = sample(c(1:nrow(cystfibr)), round(3*sqrt(nrow(cystfibr))-1))
   test = -train

   best.subset = regsubsets(pemax ~ ., data=cystfibr[train,], nvmax=npred)
   train.errors[j,] = summary(best.subset)$rss / best.subset$nn

   x.test = model.matrix(pemax ~ ., data=cystfibr[test,])
   y.test = y[test]

   for(i in 1:npred) {
      coefi = coef(best.subset, id=i)
      pred = x.test[, names(coefi)] %*% coefi
      cv.errors[j,i] = mean((y.test - pred)^2)
   }
}
```

```
mean.cv.errors = apply(cv.errors, 2, mean)
mean.train.errors = apply(train.errors, 2, mean)

# Best subset optimal number of predictors
index.best = which.min(mean.cv.errors)

# Test MSE of Best Subsets
pred.BS = cv.errors[,index.best]
test.MSE_bestsub = mean.cv.errors[index.best]

# Figure 6: Test and training MSE using Best subsets regression with Monte Carlo simulation (100
 samplings)

ggplot() +
   geom_point(aes(x=c(1:npred),y=mean.cv.errors, color="Test MSE"), size=3) +
   geom_line(aes(x=c(1:npred),y=mean.cv.errors, color="Test MSE"), linetype="solid") +
   geom_point(aes(x=c(1:npred),y=mean.train.errors, color="Training MSE"), size=3) +
   geom_line(aes(x=c(1:npred),y=mean.train.errors, color="Training MSE"), linetype="solid") +
   scale_x_continuous(breaks=c(1:npred)) +
   labs(x="Number of Predictors", y="Mean Squared Error", color=c("Test/Training"))

##############################################################################
### Best subsets fit to full dataset with optimal number of predictors

reg.best = regsubsets(pemax ~ ., data=cystfibr, nvmax=npred)
coef(reg.best, index.best)

reg.best.lm = lm(pemax ~ weight, data=cystfibr)
summary_reg.best.lm = summary.lm(reg.best.lm)
summary_reg.best.lm$coef

#################################################
### Ridge Regression (with Monte Carlo CV)
#################################################

library(glmnet)

grid = exp(seq(8, -6, length=100))
n_Monte = 100
cv.errors = rep(NA, n_Monte)
bestlam_ridge = rep(NA, n_Monte)

set.seed(3)
for(j in c(1:n_Monte)) {

   #Sampling for Monte Carlo simulation
   train = sample(c(1:nrow(cystfibr)), round(3*sqrt(nrow(cystfibr))-1))
   test = -train
   y.test = y[test]

   ridge.cv.out = cv.glmnet(x[train,], y[train], alpha=0, nfolds=5, lambda=grid)
   bestlam_min_ridge = ridge.cv.out$lambda.min

   bestlam_ridge[j] = bestlam_min_ridge
```

```
   ridge.cv.pred = predict(ridge.cv.out, s=bestlam_ridge[j], newx=x[test,],
                           exact=T, x=x[train,], y=y[train])
   cv.errors[j] = mean((ridge.cv.pred - y.test)^2)
}

cv.output_ridge = data.frame("folds"=c(1:n_Monte), "lambda"=bestlam_ridge, "Test_MSE"=cv.errors)

avg.bestlam_ridge = mean(bestlam_ridge)

# Test MSE from Ridge Regression with lambda.min
pred.ridge = cv.errors
test.MSE_ridge.min = mean(cv.errors)

cat("Optimal lambda min =", avg.bestlam_ridge)

########################################
### Fit Ridge Regression to full dataset

out_ridge = glmnet(x, y, alpha=0, lambda=grid)
coef.bestlam_ridge = coef(out_ridge, s=avg.bestlam_ridge, exact=T, x=x, y=y)[,1]

#Figure.7, fig.ap="Figure 15: Coefficients vs. log(lambda) for Ridge Regression with full datase
t
plot(out_ridge, xvar="lambda", label=T,
     sub=paste("(Vertical line at lambda_min =", round(avg.bestlam_ridge, 1), ")"))
abline(v=log(avg.bestlam_ridge ))

###################################
### Lasso Rgression
###################################

grid = exp(seq(4, -5, length=100))

cv.errors = rep(NA, n_Monte)
bestlam_lasso = rep(NA, n_Monte)

set.seed(3)
for(j in c(1:n_Monte)) {

   #Sampling for Monte Carlo simulation
   train = sample(c(1:nrow(x)), round(3*sqrt(nrow(x))-1))
   test = -(train)
   y.test = y[test]

   lasso.cv.out = cv.glmnet(x[train,], y[train], alpha=1, nfolds=5, lambda=grid)
   bestlam_min_lasso = lasso.cv.out$lambda.min

   bestlam_lasso[j] = bestlam_min_lasso
   lasso.cv.pred = predict(lasso.cv.out, s=bestlam_lasso[j], newx=x[test,],
                           exact=T, x=x[train,], y=y[train])
   cv.errors[j] = mean((lasso.cv.pred - y.test)^2)
}
```

```
cv.output_lasso = data.frame("folds"=c(1:n_Monte), "lambda"=bestlam_lasso, "Test_MSE"=cv.errors)

avg.bestlam_lasso = mean(bestlam_lasso)
avg.cv.errors = mean(cv.errors)

# Test MSE from Lasso Regression with lambda.min
pred.lasso = cv.errors
test.MSE_lasso.min = avg.cv.errors

cat("Optimal lambda min =", avg.bestlam_lasso)

##########################################
### Fit Lasso Regression to full dataset

out_lasso = glmnet(x, y, alpha=1, lambda=grid)
coef.bestlam_lasso = coef(out_lasso, s=avg.bestlam_lasso, exact=T, x=x, y=y)[,1]

# Figure 8: Coefficients vs. log(lambda) for Ridge Regression with full dataset
plot(out_lasso, xvar="lambda", label=T,
     sub=paste("(Vertical line at lambda_min =", round(avg.bestlam_lasso, 1), ")"))
abline(v=log(avg.bestlam_lasso))

##########################################
### DISCUSSIONS
##########################################

MSE.all = list(pred.pcr, pred.plsr, pred.BS, pred.ridge, pred.lasso)
names(MSE.all) = c("PCR", "PLSR", "BS", "Ridge", "Lasso")

melt_test.MSE = melt(MSE.all)
colnames(melt_test.MSE) = c("Test.MSE", "Method")

melt_test.MSE = melt_test.MSE[c(2,1)]
melt_test.MSE$Method = factor(melt_test.MSE$Method, levels=c("PCR", "PLSR", "BS", "Ridge", "Lass
o"))

# Figure 9: Comparison of Test MSE of different model selection

ggplot(data=melt_test.MSE, aes(x=Method, y=Test.MSE)) +
   coord_cartesian(ylim = c(400, 2000)) +
   stat_summary(fun.y=mean, fun.ymax=max, fun.ymin=min, geom="point", size=3) +
   stat_summary(fun.ymax=function(z) {quantile(z,0.75)},
                fun.ymin=function(z) {quantile(z,0.25)},
                geom="errorbar", width=0.2) +
   labs(x="Model selection method", y="Test MSE")

library(knitr)

# Table of model coefficients
coef.table = matrix(NA, 10, 3, dimnames=list(reg.best$xnames, c("BS", "Ridge", "Lasso")))

coef.table[names(coef(reg.best, index.best)), 1] = coef(reg.best, index.best)
coef.table[names(coef.bestlam_ridge), 2] = coef.bestlam_ridge
coef.table[names(coef.bestlam_lasso), 3] = coef.bestlam_lasso
```

```
coef.table[coef.table == 0] = NA
n.comp = apply(!is.na(coef.table), 2, sum)-1

coef.table = rbind(coef.table, n.comp)
rownames(coef.table)[11] = "Num.Comp"

# Table 1: Number of coefficients and values selected by the Best subsets, Ridge regression and
 Lasso regression

kable(coef.table, format="html", table.attr = "style='width:40%;'",
      caption="Table 1: Number of coefficients and values selected by the Best subsets, Ridge re
gression and Lasso regression")

coef.table1 = cbind(pcr.fit.full$loadings[, 1], plsr.fit.full$loadings[, 1])
colnames(coef.table1) = c("PCR-Comp1", "PLSR-Comp1")

# Table 2: Loadings of the first component for PCR and PLSR
kable(coef.table1, format="html", table.attr="style='width:40%;'",
      caption="Table 2: Loadings of the first component for PCR and PLSR")


temp.table = coef.table[-c(1, 11),]
temp.table[is.na(temp.table)] = 0

ss = sqrt(apply(temp.table^2, 2, sum)) # Magnitude of each method

temp.table[,1] = temp.table[,1] / ss[1]
temp.table[,2] = temp.table[,2] / ss[2]
temp.table[,3] = temp.table[,3] / ss[3]

# Table of spectral components
spectral.table = cbind(coef.table1, temp.table)

melt_spectral.table = melt(spectral.table)
colnames(melt_spectral.table) = c("Predictors", "Method", "Loadings")

# Figure 10: Spectral plot of coefficients (loadings) for different seletion methods
ggplot(data=melt_spectral.table, aes(group=Method, fill=Method)) +
   geom_col(aes(x=Predictors, y=Loadings), position="dodge") +
   labs(y="Loadings (normalized coefficients)")


fitted.table = cbind(pcr.fit.full$fitted.values[,,1],
      plsr.fit.full$fitted.values[,,1],
      reg.best.lm$fitted.values,
      predict(out_ridge, s=avg.bestlam_ridge, newx=x, exact=T, x=x, y=y),
      predict(out_lasso, s=avg.bestlam_lasso, newx=x, exact=T, x=x, y=y))
colnames(fitted.table) = c("PCR", "PLSR", "BS", "Ridge", "Lasso")

melt_fitted.table = melt(fitted.table)
colnames(melt_fitted.table) = c("Data", "Method", "Fitted.value")
melt_fitted.table = cbind(melt_fitted.table, cystfibr$pemax)
```

```
colnames(melt_fitted.table)[4] = "Obs.value"

# Figure 11: Comparison of fitted vs. observed values

ggplot(data=melt_fitted.table) +
   geom_point(aes(x=Obs.value, y=Fitted.value, color=Method), size=3) +
   geom_abline(slope=1, size=1, linetype=3) +
   annotate("text", x=142, y=155, label="Slope = 1") +
   geom_smooth(aes(x=Obs.value, y=Fitted.value, color=Method), method="lm", se=F) +
   labs(x="Observed values (pemax)", y="Fitted values (pemax)")

mean.pemax = mean(cystfibr$pemax)
TSS = sum((cystfibr$pemax - mean.pemax)^2)

RSS.pcr = sum((pcr.fit.full$fitted.values[,,1] - cystfibr$pemax)^2)
RSS.plsr = sum((plsr.fit.full$fitted.values[,,1] - cystfibr$pemax)^2)
RSS.BS = sum((reg.best.lm$fitted.values - cystfibr$pemax)^2)
RSS.ridge = sum((predict(out_ridge, s=avg.bestlam_ridge, newx=x, exact=T, x=x, y=y) - cystfibr$p
emax)^2)
RSS.lasso = sum((predict(out_lasso, s=avg.bestlam_lasso, newx=x, exact=T, x=x, y=y) - cystfibr$p
emax)^2)

R2 = cbind(1-RSS.pcr/TSS, 1-RSS.plsr/TSS, 1-RSS.BS/TSS, 1-RSS.ridge/TSS, 1-RSS.lasso/TSS)
colnames(R2) = c("PCR", "PLSR", "BS", "Ridge", "Lasso")

melt.R2 = melt(R2)[,-1]
colnames(melt.R2) = c("Method", "R2")

# Figure 12: R2 values of the model selection method with the training dataset

ggplot(data=melt.R2) +
   geom_col(aes(x=Method, y=R2*100)) +
   labs(x="Model selection method", y="R2 (%)")
```