Econ 493 B1 - Winter 2023

Homework 1 - Solution

Exercise 1

Let $y_i = \beta_0 + \beta_1 x_i + \beta_2 z_i + u_i$ where y_i is the number of hot dogs sold at an amusement park on a given day, x_i is the number of admission tickets sold that day, z_i is the daily maximum temperature, and u_i is a random error.

Answer the following questions.

1. State whether **each** of y_i, x_i, z_i, β_0, β_1, and β_2 is a coefficient or a variable.

y_i, x_i, and z_i are variables. β_0, β_1, and β_2 are coefficients.

2. Determine the units of β_0, β_1, and β_2 and describe the interpretation of each.

Units are hot dogs. The coefficients measure the responsiveness of hot dog sales to the various variables.

3. What are your expectations for the signs of β_0, β_1, and β_2 (negative, zero, positive or unsure)? Justify your answer.

β_0: unsure. β_1: > 0. β_2: unsure (maybe 0?).

4. Is it reasonable to allow for a non-zero intercept? Explain.

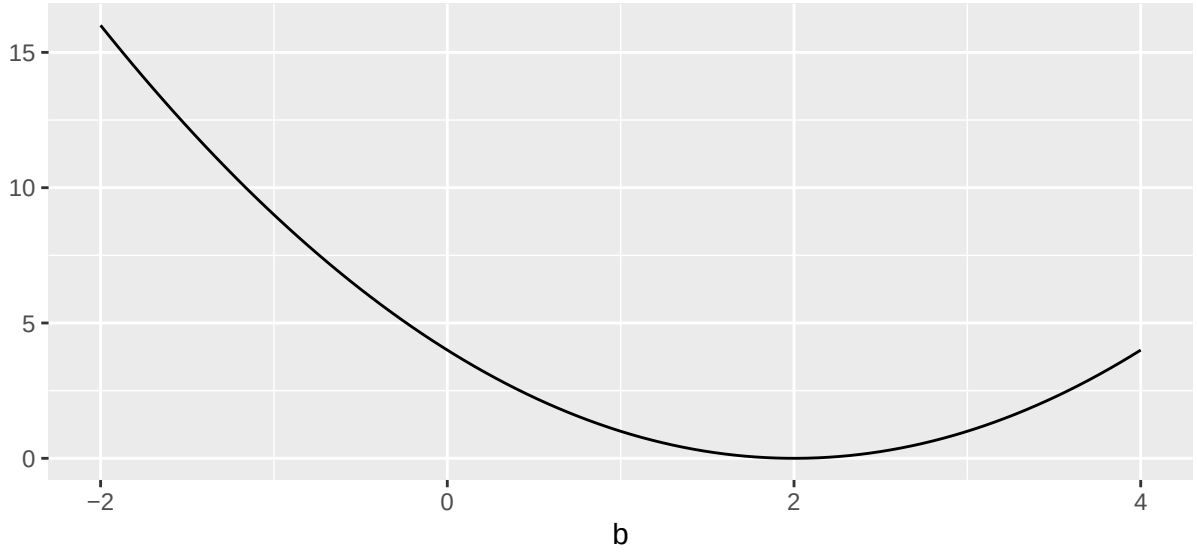
If admissions are 0, then hot dog sales should be 0 too and $\beta_0 = 0$. However, we always include a constant to allow for a better approximation.

Exercise 2

You have a sample size $n = 1$ with data $y_1 = 2$ and $x_1 = 1$. You are interested in the value of β in the regression $y = \beta x + u$. Note there is no intercept.

a. Plot the sum of squared residuals $(y_1 - bx_1)^2$ as a function of b.

```
# I will use R but solving it "by hand" is fine
# SSR function
SSR <- function(b){(2 - b*1)^2}
# plot
ggplot(data.frame(b = c(-2,4)), aes(x = b)) +
  stat_function(fun = SSR) +
  ylab("")
```



b. Show that the least squares estimate of β is $\hat{\beta}_{OLS} = 2$.

SSR is minimized at $b = 2$.

Mathematically: take first derivative of SSR with respect to b and equate to 0:

$$2(y_1 - bx_1)(-x_1) = 0$$

Next, solve for b:

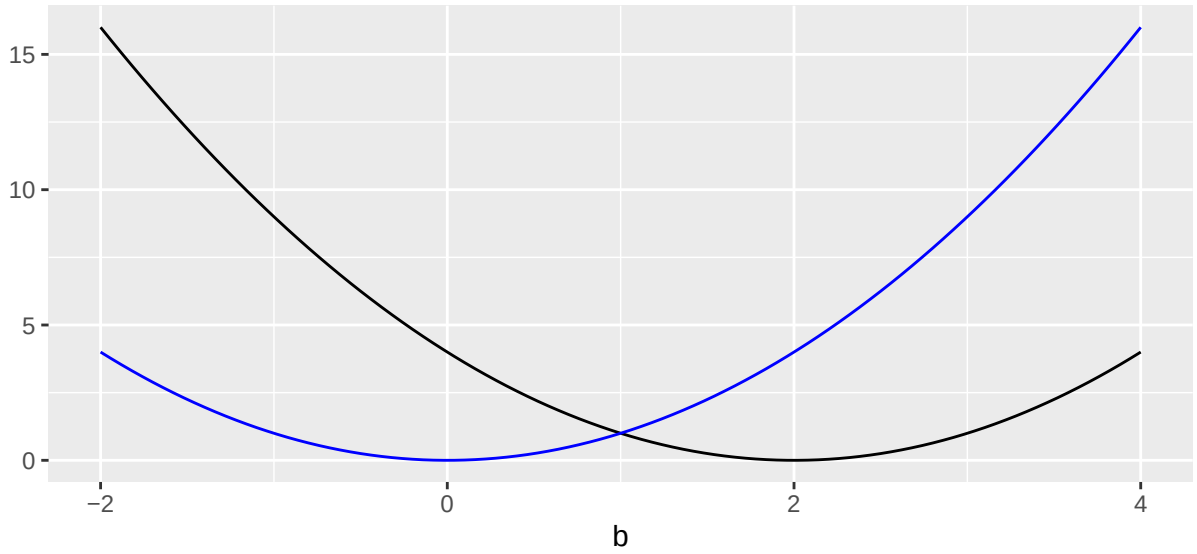
$$b = \frac{y_1 x_1}{x_1^2}$$

Finally, substitute $y_1 = 2$ and $x_1 = 1$ to obtain $b = 2$.

Then, $\hat{\beta}_{OLS} = 2$.

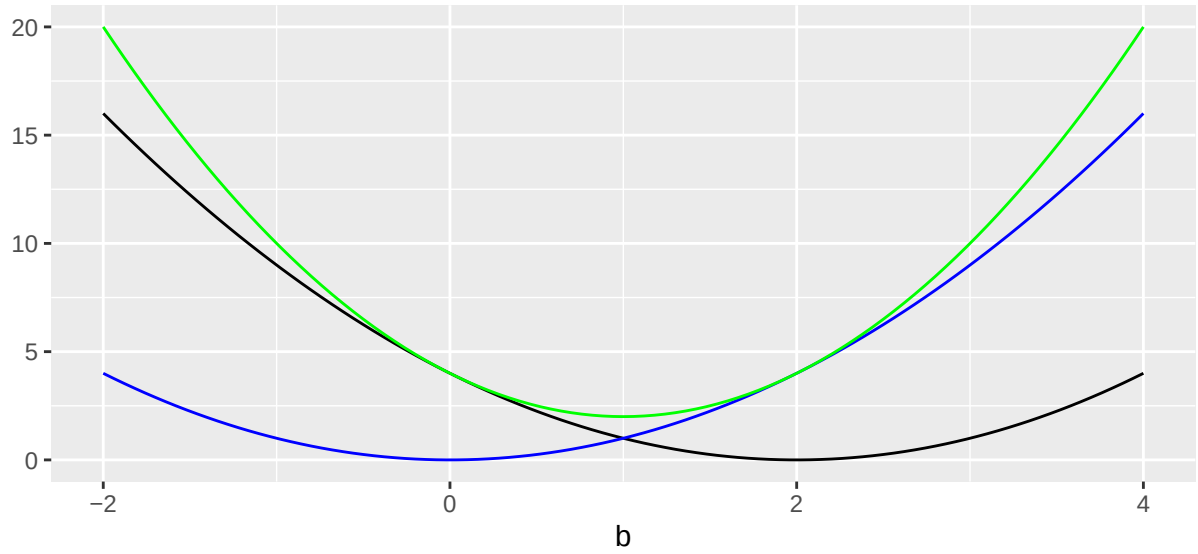
c. Using $\lambda_{ridge} = 1$, plot the ridge penalty term $\lambda_{ridge} b^2$ as a function b.

```
# ridge penalty, lambda = 1
ridge_pen <- function(b){1*(b^2)}
# plot
ggplot(data.frame(b = c(-2,4)), aes(x = b)) +
  stat_function(fun = SSR) +
  stat_function(fun = ridge_pen, color = "blue") +
  ylab("")
```



d. Using $\lambda_{ridge} = 1$, plot the ridge penalized sum of squared residuals $(y_1 - bx_1)^2 + \lambda_{ridge} b^2$.

```
# ridge penalty, lambda = 1
ridge <- function(b){(2 - b*1)^2 + 1*(b^2)}
# plot
ggplot(data.frame(b = c(-2,4)), aes(x = b)) +
  stat_function(fun = SSR) +
  stat_function(fun = ridge_pen, color = "blue") +
  stat_function(fun = ridge, color = "green") +
  ylab("")
```



e. Find the value of $\hat{\beta}_{ridge}$.

The ridge penalized sum of squared residuals appears to be minimized at $b = 1$.

Mathematically: take first derivative of the ridge penalized sum of squared residuals with respect to b and equate to 0:

$$2(y_1 - bx_1)(-x_1) + 2\lambda b = 0$$

Next, solve for b:

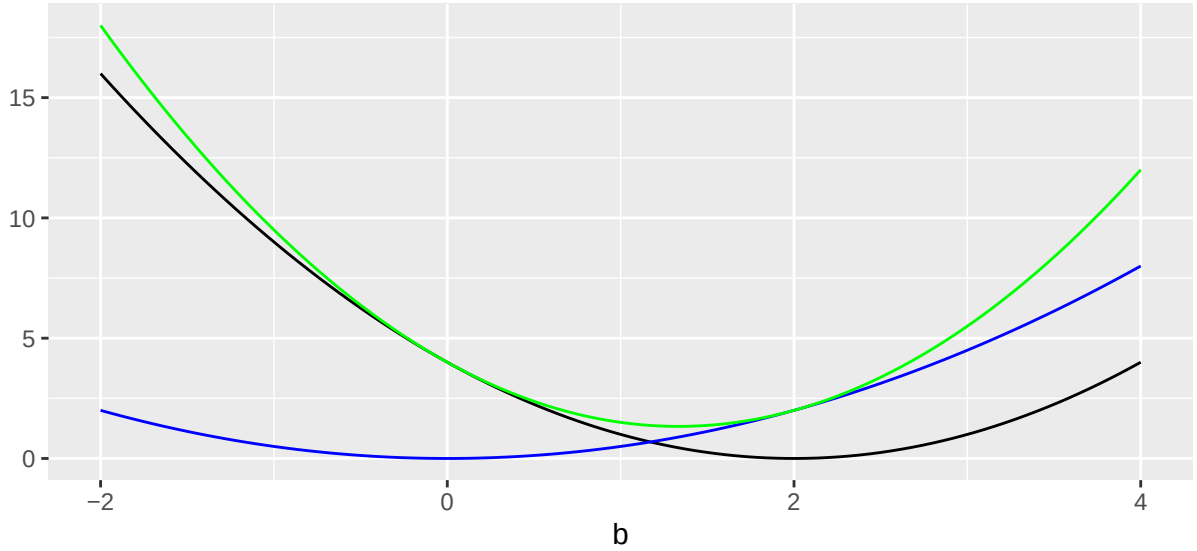
$$b = \frac{y_1 x_1}{\lambda + x_1^2}$$

Finally, substitute $\lambda = 1$, $y_1 = 2$ and $x_1 = 1$ to obtain $b = 1$.

Then, $\hat{\beta}_{ridge} = 1$.

f. Using $\lambda_{Ridge} = 0.5$, repeat (c) and (d). Find the value of $\hat{\beta}_{ridge}$.

```
# ridge penalty, lambda = .5
SSR <- function(b){(2 - b*1)^2}
ridge_pen <- function(b){.5*(b^2)}
ridge <- function(b){(2 - b*1)^2 + .5*(b^2)}
# plot
ggplot(data.frame(b = c(-2,4)), aes(x = b)) +
  stat_function(fun = SSR) +
  stat_function(fun = ridge_pen, color = "blue") +
  stat_function(fun = ridge, color = "green") +
  ylab("")
```

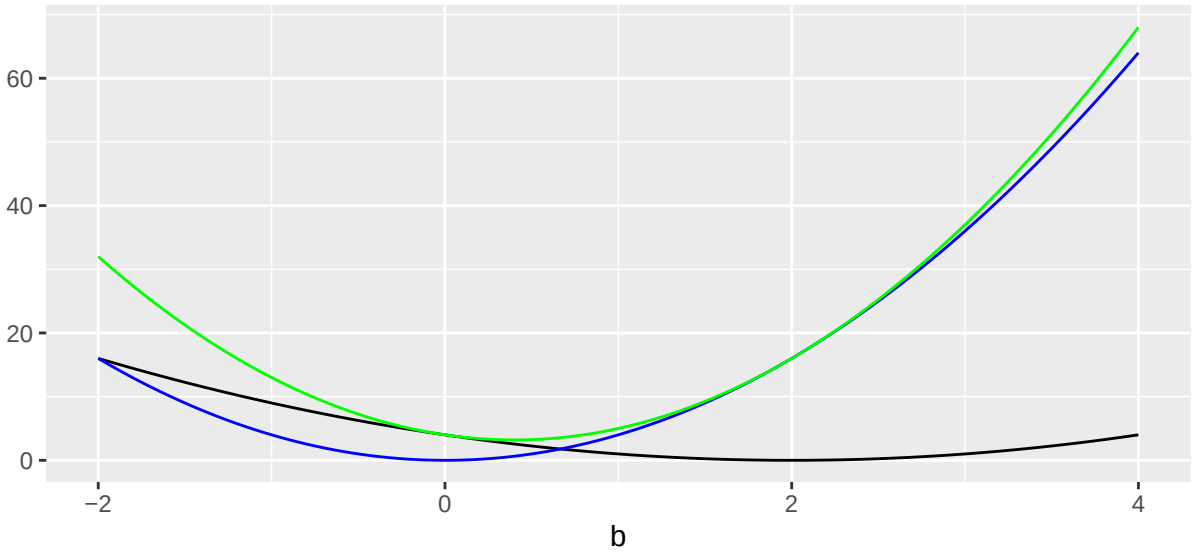


Substitute $\lambda = 0.5$, $y_1 = 2$ and $x_1 = 1$ to obtain $b = 4/3$.

Then, $\hat{\beta}_{ridge} = 4/3$.

g. Using $\lambda_{ridge} = 4$, repeat (c) and (d). Find the value of $\hat{\beta}_{ridge}$.

```
# ridge penalty, lambda = 4
SSR <- function(b){(2 - b*1)^2}
ridge_pen <- function(b){4*(b^2)}
ridge <- function(b){(2 - b*1)^2 + 4*(b^2)}
# plot
ggplot(data.frame(b = c(-2,4)), aes(x = b)) +
  stat_function(fun = SSR) +
  stat_function(fun = ridge_pen, color = "blue") +
  stat_function(fun = ridge, color = "green") +
  ylab("")
```



Substitute $\lambda = 4$, $y_1 = 2$ and $x_1 = 1$ to obtain $b = 2/5$.

Then, $\hat{\beta}_{ridge} = 2/5$.

h. Use the graphs that you produced in (a)-(d) for the various values of λ_{ridge} to explain why a larger value of λ_{ridge} results in more shrinkage of the OLS estimate.

As λ_{ridge} increases, the ridge penalty becomes steeper, shifting the minimizing value of the ridge-penalized SSR toward zero.

Exercise 3

You have a sample size $n = 1$ with data $y_1 = 2$ and $x_1 = 1$. You are interested in the value of β in the regression $y = \beta x + u$. Note there is no intercept.

a. Plot the sum of squared residuals $(y_1 - bx_1)^2$ as a function of b.

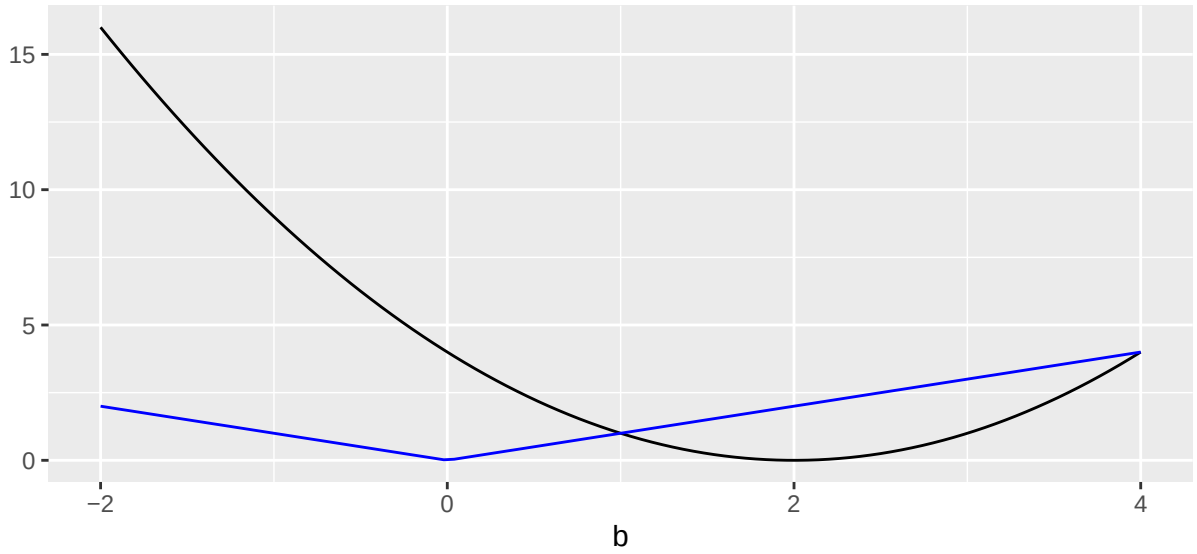
Same as Exercise 2.

b. Show that the least squares estimate of β is $\hat{\beta}_{OLS} = 2$.

Same as Exercise 2.

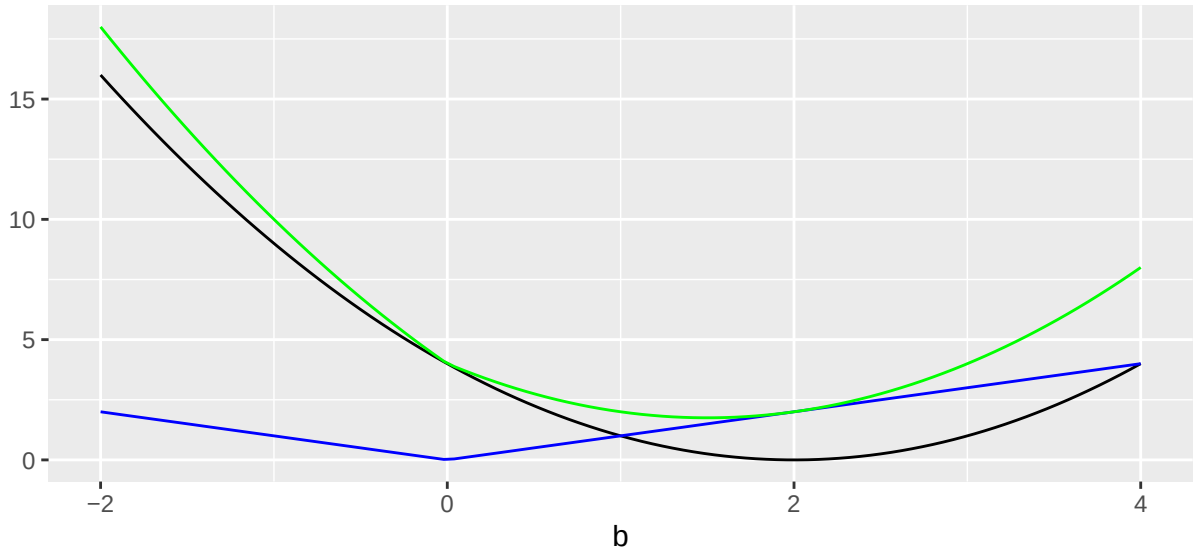
c. Using $\lambda_{lasso} = 1$, plot the lasso penalty term $\lambda_{lasso}|b|$ as a function b.

```
# lasso penalty, lambda = 1
SSR <- function(b){(2 - b*1)^2}
lasso_pen <- function(b){1*abs(b)}
# plot
ggplot(data.frame(b = c(-2,4)), aes(x = b)) +
  stat_function(fun = SSR) +
  stat_function(fun = lasso_pen, color = "blue") +
  ylab("")
```



d. Using $\lambda_{lasso} = 1$, plot the lasso penalized sum of squared residuals $(y_1 - bx_1)^2 + \lambda_{lasso}|b|$.

```
# lasso penalty, lambda = 1
lasso <- function(b){(2 - b*1)^2 + 1*abs(b)}
# plot
ggplot(data.frame(b = c(-2,4)), aes(x = b)) +
  stat_function(fun = SSR) +
  stat_function(fun = lasso_pen, color = "blue") +
  stat_function(fun = lasso, color = "green") +
  ylab("")
```



e. Find the value of $\hat{\beta}_{lasso}$. Assume $b > 0$.

The lasso penalized sum of squared residuals appears to be minimized somewhere between 1 and 2.

Mathematically: take first derivative of the lasso penalized sum of squared residuals with respect to b and equate to 0:

$$2(y_1 - bx_1)(-x_1) + \lambda \frac{b}{|b|} = 0$$

Given that $b > 0$ we have $\frac{b}{|b|} = 1$ and we can solve for b:

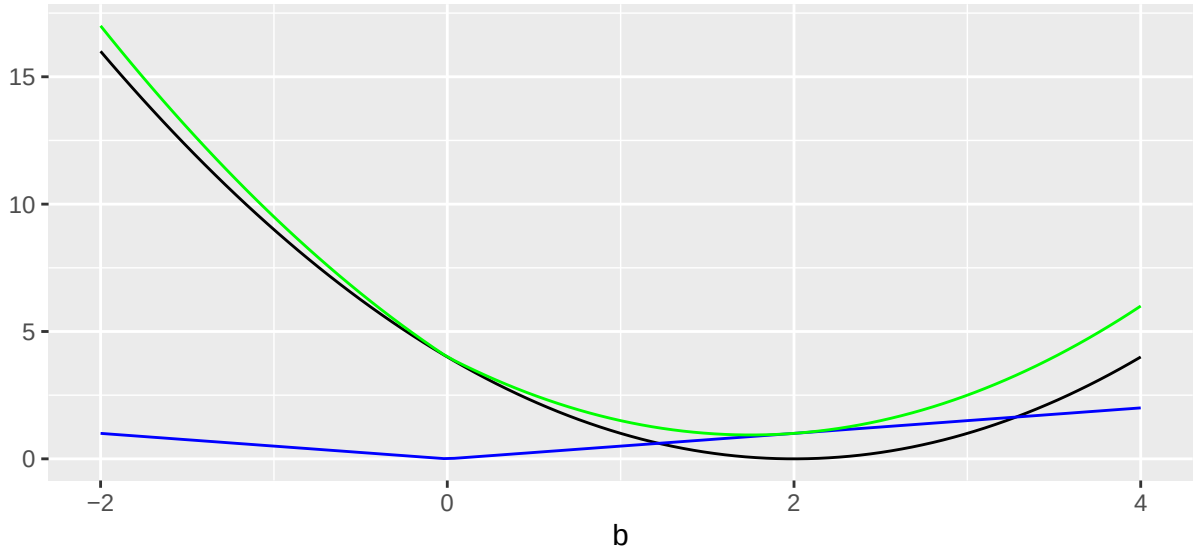
$$b = \frac{y_1 x_1}{x_1^2} - \frac{\lambda}{2x_1^2}$$

Finally, substitute $\lambda = 1$, $y_1 = 2$ and $x_1 = 1$ to obtain $b = 1.5$.

Then, $\hat{\beta}_{lasso} = 1.5$.

f. Using $\lambda_{lasso} = 0.5$, repeat (c) and (d). Find the value of $\hat{\beta}_{lasso}$.

```
# lasso penalty, lambda = .5
lasso_pen <- function(b){.5*abs(b)}
lasso <- function(b){(2 - b*1)^2 + .5*abs(b)}
# plot
ggplot(data.frame(b = c(-2,4)), aes(x = b)) +
  stat_function(fun = SSR) +
  stat_function(fun = lasso_pen, color = "blue") +
  stat_function(fun = lasso, color = "green") +
  ylab("")
```

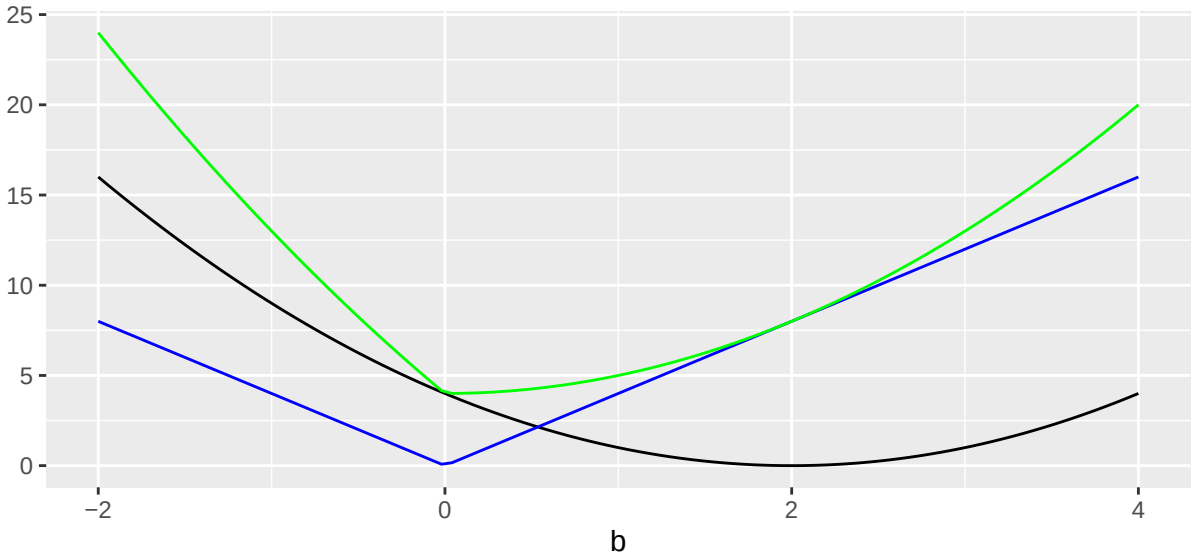


Substitute $\lambda = 0.5$, $y_1 = 2$ and $x_1 = 1$ to obtain $b = 1.75$.

Then, $\hat{\beta}_{lasso} = 1.75$.

g. Using $\lambda_{lasso} = 4$, repeat (c) and (d). Find the value of $\hat{\beta}_{lasso}$.

```
# lasso penalty, lambda = 4
lasso_pen <- function(b){4*abs(b)}
lasso <- function(b){(2 - b*1)^2 + 4*abs(b)}
# plot
ggplot(data.frame(b = c(-2,4)), aes(x = b)) +
  stat_function(fun = SSR) +
  stat_function(fun = lasso_pen, color = "blue") +
  stat_function(fun = lasso, color = "green") +
  ylab("")
```



Substitute $\lambda = 4$, $y_1 = 2$ and $x_1 = 1$ to obtain $b = 0$.

Then, $\hat{\beta}_{lasso} = 0$.

h. Use the graphs that you produced in (a)-(d) for the various values of λ_{lasso} to explain why a larger value of λ_{lasso} results in more shrinkage of the OLS estimate.

As λ_{lasso} increases, the lasso penalty becomes steeper, shifting the minimizing value of the lasso-penalized SSR toward zero.

Exercise 4 (R)

We will now perform model selection using information criteria and k-fold cross-validation on a simulated data set.

Generate a simulated data set as follows:

```
set.seed(1234)
n.obs <- 100
x1 <- rnorm(n.obs)
y <- x1 - 2*x1^2 + rnorm(n.obs)
```

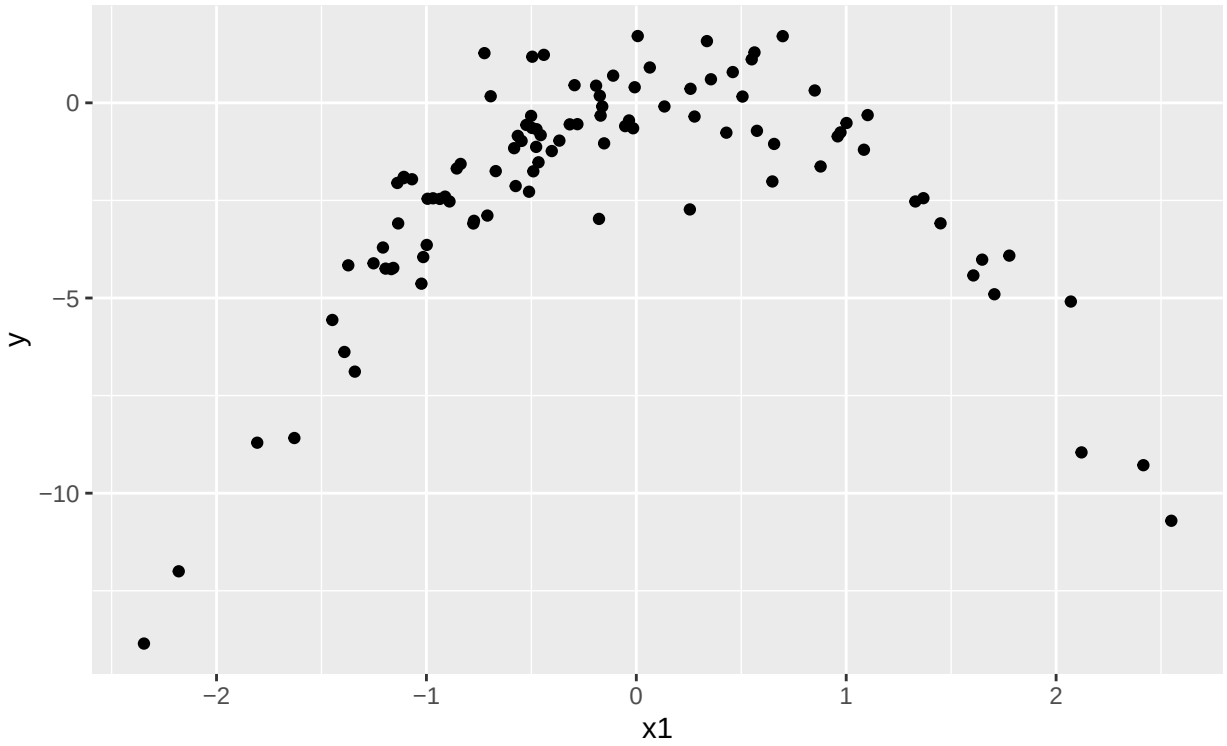
a. Write out the model used to generate the data in equation form.

The equation is

$$y_t = x_t - 2x_t^2 + \nu_t, \qquad \nu_t \sim \text{iid } (0, 1)$$

b. Create a scatterplot of x against y. Comment on what you find.

```
data <- data.frame(y,x1)
ggplot(data, aes(x=x1,y=y)) + geom_point()
```



There is an inverted U-shaped relationship between y and x.

c. Compute the BIC and AIC for the following four models estimated using least squares. Which model would you prefer?

```
x2 <- x1*x1
x3 <- x2*x1
x4 <- x3*x1
data.all <- data.frame(y,x1,x2,x3,x4)
# estimate models
model1 <- lm(y~x1,data=data.all)
model2 <- lm(y~x1+x2,data=data.all)
model3 <- lm(y~x1+x2+x3,data=data.all)
model4 <- lm(y~x1+x2+x3+x4,data=data.all)
# BIC
c(BIC(model1),BIC(model2),BIC(model3),BIC(model4))
```

```
## [1] 510.56 306.00 310.27 314.85
```

```
# AIC
c(AIC(model1),AIC(model2),AIC(model3),AIC(model4))
```

```
## [1] 502.75 295.58 297.24 299.22
```

Both BIC and AIC select model ii.

d. Compute the k-fold cross-validation errors that result from fitting the four models. Use $k = 5$. Which model would you prefer? Is this what you expected? Explain your answer.

```
# k-fold cross-validation
cv.error <- rep(NA,4)
```

```r
# model 1
model1.cv <- glm(y~x1)
cv.error[1] <- cv.glm(data.all,model1.cv,K=5)$delta[1]
# model 2
model2.cv <- glm(y~x1+x2)
cv.error[2] <- cv.glm(data.all,model2.cv,K=5)$delta[1]
# model 3
model3.cv <- glm(y~x1+x2+x3)
cv.error[3] <- cv.glm(data.all,model3.cv,K=5)$delta[1]
# model 4
model4.cv <- glm(y~x1+x2+x3+x4)
cv.error[4] <- cv.glm(data.all,model4.cv,K=5)$delta[1]

print(cv.error,digits=4)
```

```
## [1] 8.865 1.079 1.157 1.116
```

Model i shows the worst fit. The other models exhibit a similar fit.

e. Comment on the statistical significance of the coefficient estimates that results from fitting each of the models in (c) using least squares. Do these results agree with the conclusions drawn based on information criteria (c) and the cross-validation results (d)?

```r
# tests
summary(model1)
```

```
##
## Call:
## lm(formula = y ~ x1, data = data.all)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.794  -1.008   0.766   1.694   3.806
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -2.098      0.297   -7.07  2.3e-10 ***
## x1             0.409      0.293    1.40     0.17
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.93 on 98 degrees of freedom
## Multiple R-squared:  0.0195, Adjusted R-squared:  0.00951
## F-statistic: 1.95 on 1 and 98 DF,  p-value: 0.166
```

```r
summary(model2)
```

```
##
## Call:
## lm(formula = y ~ x1 + x2, data = data.all)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9893 -0.6581 -0.0033  0.5682  2.9551
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.1395     0.1351    1.03      0.3
## x1            1.0010     0.1060    9.45  2.1e-15 ***
## x2           -2.0959     0.0799  -26.24  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.03 on 97 degrees of freedom
## Multiple R-squared:  0.879,  Adjusted R-squared:  0.876
## F-statistic:  352 on 2 and 97 DF,  p-value: <2e-16
```

```
summary(model3)
```

```
##
## Call:
## lm(formula = y ~ x1 + x2 + x3, data = data.all)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -2.959 -0.661  0.039  0.535  2.916
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.1325     0.1361    0.97     0.33
## x1            0.9126     0.1879    4.86  4.6e-06 ***
## x2           -2.1064     0.0822  -25.62  < 2e-16 ***
## x3            0.0323     0.0566    0.57     0.57
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.04 on 96 degrees of freedom
## Multiple R-squared:  0.879,  Adjusted R-squared:  0.876
## F-statistic:  233 on 3 and 96 DF,  p-value: <2e-16
```

```
summary(model4)
```

```
##
## Call:
## lm(formula = y ~ x1 + x2 + x3 + x4, data = data.all)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9473 -0.6536  0.0398  0.5358  2.9149
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.11872    0.16543    0.72     0.47
## x1           0.91080    0.18924    4.81  5.6e-06 ***
## x2          -2.07577    0.22284   -9.32  4.8e-15 ***
## x3           0.03423    0.05837    0.59     0.56
## x4          -0.00644    0.04357   -0.15     0.88
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.04 on 95 degrees of freedom
## Multiple R-squared:  0.879,  Adjusted R-squared:  0.874
## F-statistic:  173 on 4 and 95 DF,  p-value: <2e-16
```

Yes. Only x and x^2 are significant.

Exercise 5 (R)

The data set `males1987.csv` contains 545 observations of young working males in the United States with some professional and personal characteristics for the year 1987. Variable names and descriptions are provided in `males1987.txt`. We want to explain log-wages.

a. Create a matrix $\boldsymbol{X}$ (545×9) with the 7 explanatory variables described above plus experience and schooling squared. Scale the matrix $\boldsymbol{X}$ such that all variables have the same variance. Create a vector $\boldsymbol{y}$ (545×1) with log wage.

```
#
set.seed(1234)

# load data
DATA <-read.csv("males1987.csv", header = TRUE)
head(DATA, 3)
```

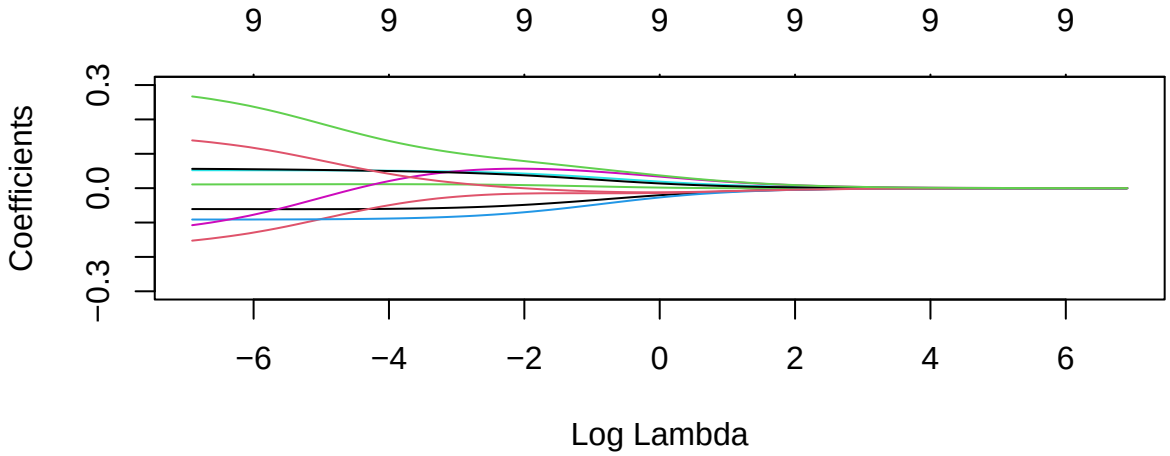
```
##   BLACK EXPER HISP  HOURS MAR SCHOOL UNION LOGWAGE EXPER2
## 1     0     8    0 50.769   0     14     0  1.6692     64
## 2     0    11    0 47.615   0     13     0  1.8203    121
## 3     0    11    0 45.000   1     12     0  2.8732    121
```

```
n.obs <- dim(DATA)[1]

# get data
y <- matrix(DATA[,8],n.obs,1)
x <- matrix(0,n.obs,9)
for (i in 1:7){x[,i] <- DATA[,i]}
x[,8] <- DATA[,2]^2
x[,9] <- DATA[,6]^2
x <- scale(x)
data.all <- data.frame(x,y)
attach(data.all)
```

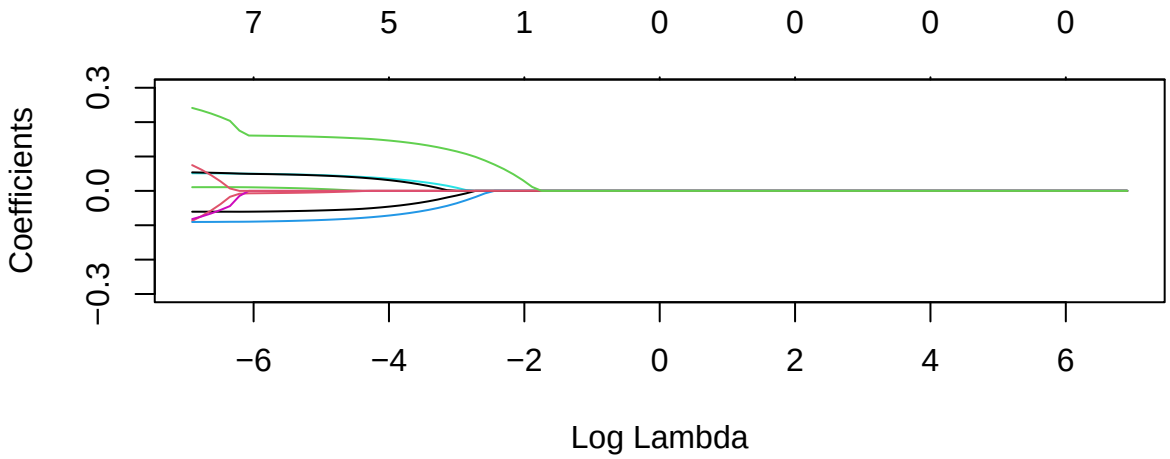
```
## The following object is masked _by_ .GlobalEnv:
##
##     y
```

b. Plot the ridge regression standardized coefficients for different values of λ. Plot the lasso standardized coefficients for different values of λ. Comment on your results.

```
# ridge reg for different values of lambda
grid <- 10^seq(3,-3,length=100)
ridge.mod <- glmnet(x,y,alpha=0,lambda=grid)

# plot ridge results
plot(ridge.mod,xvar="lambda",ylim=c(-.3,.3))
```




```
# lasso reg for different values of lambda
lasso.mod <- glmnet(x,y,alpha=1,lambda=grid)

# plot lasso results
plot(lasso.mod,xvar="lambda",ylim=c(-.3,.3))
```



c. Estimate the parameters by OLS using the full sample. Which variables are statistically significant at the 10% level.

```
# least squares
model1 <- lm(y~x,data=data.all)
```

```
summary(model1)
```

```
##
## Call:
## lm(formula = y ~ x, data = data.all)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -1.606 -0.277  0.042  0.254  1.445
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.8665     0.0179  104.07  < 2e-16 ***
## x1           -0.0606     0.0190   -3.19   0.0015 **
## x2           -0.1901     0.1760   -1.08   0.2804
## x3            0.0105     0.0188    0.56   0.5761
## x4           -0.0913     0.0181   -5.04  6.4e-07 ***
## x5            0.0524     0.0185    2.84   0.0047 **
## x6           -0.1315     0.1674   -0.79   0.4325
## x7            0.0574     0.0186    3.09   0.0021 **
## x8            0.1752     0.1850    0.95   0.3438
## x9            0.2905     0.1590    1.83   0.0682 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.419 on 535 degrees of freedom
## Multiple R-squared:  0.209,  Adjusted R-squared:  0.196
## F-statistic: 15.7 on 9 and 535 DF,  p-value: <2e-16
```

The significant variables are x_1, x_4, x_5, x_7, and x_9.

d. Perform best subset selection using the 9 explanatory variables and the full sample. Based on the BIC, which variables are included in the best model? Based on the AIC, which variables are included in the best model? Compare your results with OLS.

```
# best subset selection
model2 <- regsubsets(y~.,data=data.all,nvmax=9)
reg.summary <- summary(model2)

# coefficients associated with best model by bic
coef(model2,which.min(reg.summary$bic))
```

```
## (Intercept)          X1          X4          X5          X7          X9
##    1.866479   -0.065400   -0.092529    0.050748    0.052884    0.165096
```

```
# coefficients associated with best model by aic
coef(model2,which.min(reg.summary$cp))
```

```
## (Intercept)          X1          X4          X5          X6          X7
##    1.866479   -0.064474   -0.091210    0.050060   -0.181457    0.056982
```

```
##          X9
##    0.344596
```

e. Estimate the lasso coefficients using the full sample. Are any of the coefficients forced to be exactly 0? Compare your results with OLS and best subset selection.

```
lasso.cv <- cv.glmnet(x,y,alpha=1,nfolds=5)
lasso.lam <- lasso.cv$lambda.min
model4 <- glmnet(x,y,alpha=1,lambda=lasso.lam)
coef(model4)
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                     s0
## (Intercept)  1.866479
## V1          -0.060492
## V2          -0.147945
## V3           0.010674
## V4          -0.091003
## V5           0.052068
## V6          -0.136686
## V7           0.056608
## V8           0.132340
## V9           0.294727
```

f. Re-estimate all the models (parts c, d, and e) using a training set with (about) half of the observations. Obtain the test sample errors for all the models. Comment on your results.

```
cv.error <- rep(NA,4)

# split the sample into train/test sets
train <- sample(nrow(data.all),round(nrow(data.all)/2))
train[1:5]
```

```
## [1] 223 354 218 187 485
```

```
# least squares
model1.cv <- lm(y~x,data=data.all,subset=train)
cv.error[1] <- mean((y-predict(model1.cv,data.all))[-train]^2)

# best subset selection
regfit.train <- regsubsets(y~.,data=data.all,subset=train,nvmax=9)
reg.summary <- summary(regfit.train)

# bic
coef2 <- coef(regfit.train,id=which.min(reg.summary$bic))
test.mat <- model.matrix(y~.,data=data.all)
cv.error[2] <- mean((y-test.mat[,names(coef2)]%*%coef2)[-train]^2)

# aic
coef3 <- coef(regfit.train,id=which.min(reg.summary$cp))
```

```r
cv.error[3] <- mean((y-test.mat[,names(coef3)]%*%coef3)[-train]^2)

# lasso
lasso.cv <- cv.glmnet(x[train,],y[train],alpha=1,nfolds=5)
lasso.lam <- lasso.cv$lambda.min
cv.error[4] <- mean((y-predict(lasso.cv,s=lasso.lam,newx=x))[-train]^2)

# print test errors
print(cv.error,digits=4)
```

```
## [1] 0.1834 0.1873 0.1882 0.1837
```

All models exhibit a similar performance (the results may differ if you set a different seed).