MSDA 605 - Fundamentals of Computational Mathematics Final Exam

Ben Arancibia

December 18, 2014

This final exam consists of three parts. The three parts are 1) Essential Concepts, 2) Coding, 3) Small Project.

Essential Concepts

1) What is the rank (number of lineraly indepdent rows) of the following matrix:

```
##      [,1] [,2] [,3] [,4]
## [1,]   -1    1    3    5
## [2,]    2   -1    5    7
## [3,]    6  -10   -1    3
```

First step is multiply first row by -2, results in the following matrix:

```
##      [,1] [,2] [,3] [,4]
## [1,]    2   -2   -6  -10
## [2,]    2   -1    5    7
## [3,]    6  -10   -1    3
```

Second step is subtract first row from second row:

```
##      [,1] [,2] [,3] [,4]
## [1,]    1   -1   -3   -5
## [2,]    0    1   11   17
## [3,]    6  -10   -1    3
```

Third step multiply row 1 by 6 then subtract first row from third row.

```
##      [,1] [,2] [,3] [,4]
## [1,]    1   -1   -3   -5
## [2,]    0    1   11   17
## [3,]    0   -4   17   33
```

Fourth step multiply row 1 by -1 and then second row by -4.

```
##      [,1] [,2] [,3] [,4]
## [1,]   -1    1    3    5
## [2,]    0   -4  -44   68
## [3,]    0   -4   17   33
```

Last step subtract second row from third row.

```
##      [,1] [,2] [,3] [,4]
## [1,]   -1    1    3    5
## [2,]    0    1   11   17
## [3,]    0    0   61  101
```

The matrix rank is 3. All rows are linearly indepdent.

2) What is the determinant of the following matrix:

```
##      [,1] [,2] [,3] [,4]
## [1,]   -1    1    3    5
## [2,]    2   -1    5    7
## [3,]    6  -10   -1    3
```

It is not possible to calculate the determinant of the matrix because it is not a square matrix.

3)Define orthonormal basis vectors. Please write down at least one orthonormal basis for the 5-dimensional vector space R5.

An Orthonormal Basis vector is when a orthogonal vector divided by its length = 1. An orthogonal vector is when

$$q1, ..., qn$$

have dot products equal to zero (

$$qi * qj$$

). Divide each vector by its length and the vectors become orthogonal unit vectors. The lengths are one.

A five dimenial orthonormal basis is the following:

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    0    0    0    0
## [2,]    0    1    0    0    0
## [3,]    0    0    1    0    0
## [4,]    0    0    0    1    0
## [5,]    0    0    0    0    1
```

(this is just the standard basis)

4)Given the following matrix, what is its characteristic polynomial?

```
##      [,1] [,2] [,3]
## [1,]    2   -1    4
## [2,]   -1   -2    6
## [3,]    1    0   -3
```

Characteristic polynomial of a square matrix is a polynomial, which is invariant under matrix similarity and has the eigenvalues as roots.

$$det(A - XI)$$

```
q4.1 = matrix(c(NA, -1, 1, -1, NA, 0, 4, 6, NA), nrow=3, ncol=3) # NA = X in this matrix
q4.1
```

```
##      [,1] [,2] [,3]
## [1,]   NA   -1    4
## [2,]   -1   NA    6
## [3,]    1    0   NA
```

Characteristic Polynomial =

$$-x^3 - 3x^2 + 9x + 17$$

5) What are its eigenvectors and eigenvalues of the following matrix?

```
##      [,1] [,2] [,3]
## [1,]    2   -1    4
## [2,]   -1   -2    6
## [3,]    1    0   -3
```

Characteristic Polynomial =

$$-x^3 - 3x^2 + 9x + 17$$

derived from question 4. Find the roots for the characterisitic polynominal.

Eigenvalue of x1 (2-x) = 2.691 Eigenvalue of x2 (-2-x) = -4.18 Eigenvalue of x3 (-3-x) = -1.511

With those values if you plug them in the following are the results eigenvectors: x1: (86.43 , 1 , 15.18) x2: (0.36 , 1 , -0.30) x3: (0.16 , 1 , 0.11)

6)When would a model be said to have a high bias and when would it be said to have a high variance?

Non-linear modules such as polynominal fits will have a low bias because they are able to go arbitraily close to fi; however, they introduce a higher variance in the model as they may not fit any particular instance of the underlying data well. When there is a high variance but low bias, the model is referred to as overfitting, when there is a model of high bias but low variance, the model is known as underfitting. This is all part of the bias variance trade-off. Models with high bias tend to have low variance and models with high variance tend to have low bias.

7)Assuming that we are repeatedly sampling sets of numbers (each set is of size n) from an unknown probability density function. What can we say about the average value of each set?

Assuming that we are repeatedly sampling sets of numbers from an unknown probability density function dependson the type of random variable. Either for a continous or discrete random varibale the expect value of random variables states the average responses that should be obtained. The expected value of of PMF (discrete) can be calculated as the Sum of

$$P(x == i) * i)$$

. If the variable is a continouse random variable you can calculte the entire PDF with Integral of

$$xp(x)dx$$

.

8)What is the derivative of $e^{2xcos}2(x)$

Take constant out so becomes:

$$e^2 d/dx(xcos^2(x))$$

Product rule:

$$e^2((x)cos^2(x) + (cos^2(x))x)$$

Results in

$$e^2(1cos^2(x) + (-2cos(x)sin(x))x)$$

Simplfy:

$$e^2 cos(x)(cos(x) - 2xsin(x))$$

9)What is the derivative of log(sin(2x))?

Apply log rule which is

$$Log(b) = -ln(b)/ln(a)$$

Result:

$$ln(sin(2x))/ln(10)$$

10) What is the integral of

$$x^2 sin(x)dx$$

?

Integration by parts: =

$$x^2(-cos(x)) - \int 2x(-cos(x))$$

$$= x^2(-cos(x)) - \int 2xcos(x))$$

$$\int -2xcos(x) = -2(xsin(x) + cos(x))$$

$$= x^2(-cos(x) - (-2(xsin(x) + cos(x))))$$

Simplified Integral:

$$2(xsin(x) + cos(x)) - x^2 cos(x) + C$$

Compute with boundaries:

$$(0, \pi/2)$$

$$(0) : 2(xsin(0) + cos(0)) - x^2 cos(0) = 2$$

$$(\pi/2) : 2(xsin(\pi/2) + cos(\pi/2)) - x^2 cos(\pi/2) = \pi$$

Result:

$$\pi - 2$$

Mini Coding

2.1 Bayes Rule:

Working for a credit card company and 70% of customers have a good credit score. People with good credit have a loan default rate of 0.05. People with bad credit default on their loads at the rate of .10.

```
PRB <- (((.10*.30)+(.05*.30))/.7)
PRBA <- .10
PRA <- 0.5

PAB <- (PRBA*PRA)/PRB

PAB
```

```
## [1] 0.7777778
```

Supose there are two full bowls of cookies. Bowl 1 has 5 chocolate chip and 35 oatmeal raisin cookies. Bowl 2 has 20 of each. Fred picks a bowl at random and then picks a cookie at random. Cookie is chocolate chip, how probable is it that Fred pick it out of bowl 2.

```
PRB2 <- (((0.125*20)+(0.5*20))/40)
PRBA2 <- .5
PRA2 <- .125
PAB2 <- ((PRBA2*PRA2)/PRB2)

PAB2 *.05 #multipled by .5 because Fred has to select 1 of 2 bowls
```

```
## [1] 0.01
```

2.2 Central Limit Theorem

Assume a sample of size of 100 with mean of 50 and standard deviation 10. What is mean of sample and standard error?

```
x=rlnorm(100, log(50), log(10))

mean(x)
```

```
## [1] 430.2518
```

```
se <- sd(x)/sqrt(length(x))

se
```

```
## [1] 143.7357
```

Random sample of 100 observations with a mean of 40 and standard deviation of 25.

```
x=rnorm(100,mean=40,sd=25)

mean(x)
```

```
## [1] 40.45218
```

```
sd(x)
```

```
## [1] 25.24123
```

Random sample of 100 obserations mean of 40 and standard deviation of 25. What is the probability that the mean of the sample will exceed 45?

```
x=rnorm(100,mean=40,sd=25)

y=mean(x)

1-pnorm(y, 40, 25)
```

```
## [1] 0.5272738
```

2.3 Sample from function.

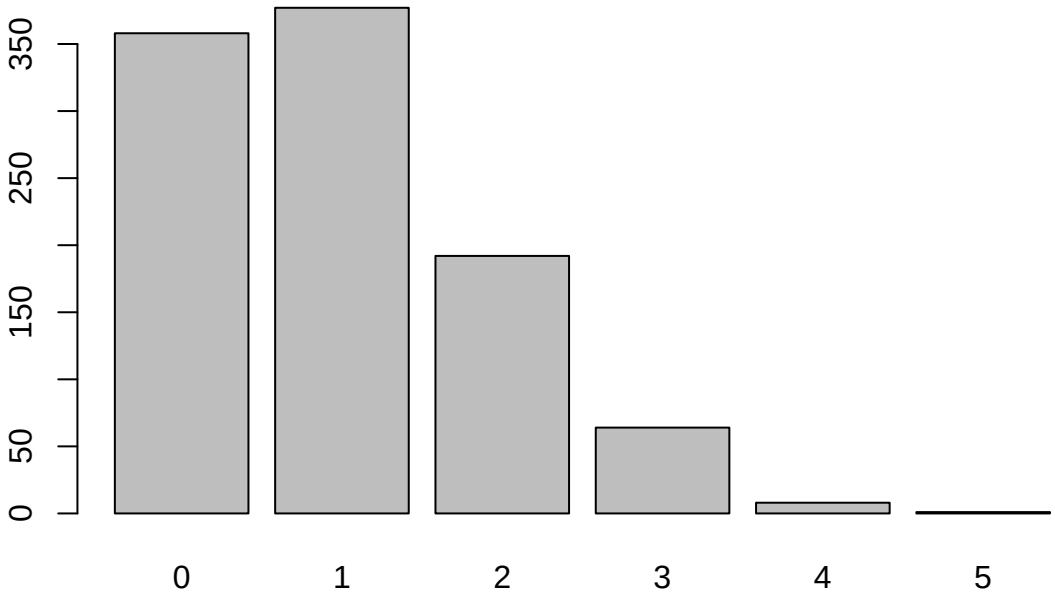
First create a binominal distribution to samepl p = 0.1, q =1-p and x[0,10]

```
rbinom(1, 10, .1)
```

```
## [1] 0
```

Generate 1000 samples using this and plot a histogram.

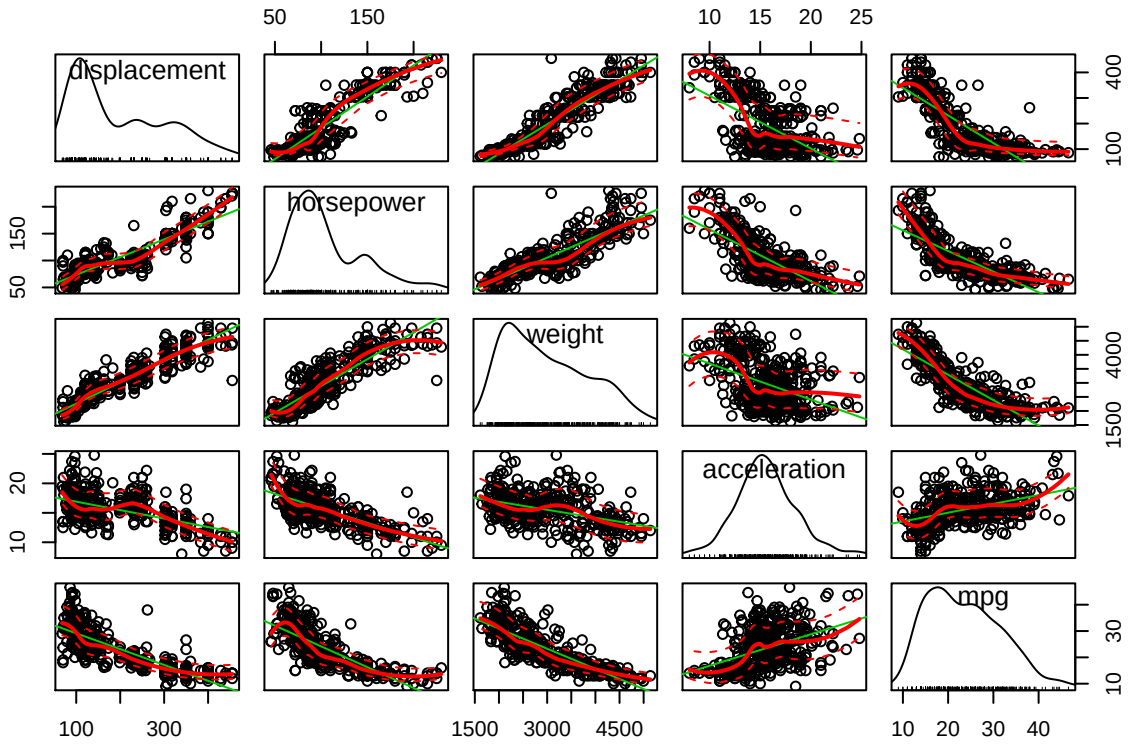
```
barplot(table(rbinom(1000,10,.1)))
```



2.4 Principal Components Analysis

Perform a principal components analysis by performing an SVD on 4 independent variables (mpg is dependent variable) and select top 2 directions. Create a scatter plot of the data and print out the two orthogonal vectors.

```
library(car)
data <- read.table("/Users/bcarancibia/CUNY_IS_605/FinalExam/auto-mpg.data")
names(data) <- c("displacement", "horsepower", "weight", "acceleration", "mpg")


scatterplotMatrix(data) #take a look at the data first
```



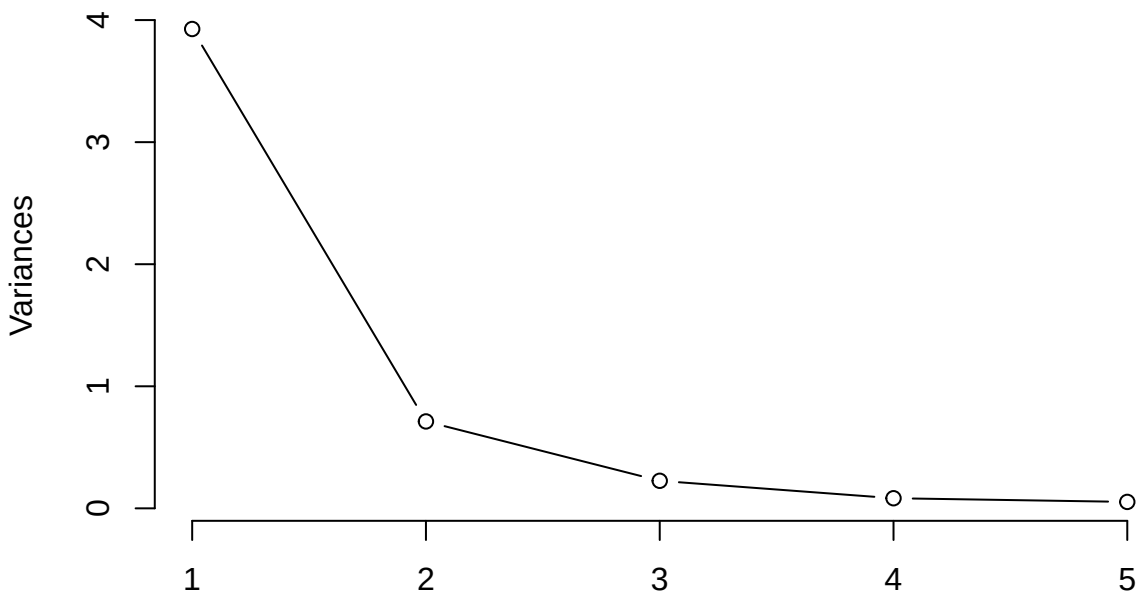

```
standard <- as.data.frame(scale(data))
car.pca <- prcomp(standard)
summary(car.pca)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5
## Standard deviation     1.9816 0.8438 0.47500 0.28788 0.22966
## Proportion of Variance 0.7853 0.1424 0.04512 0.01658 0.01055
## Cumulative Proportion  0.7853 0.9277 0.97288 0.98945 1.00000
```

```
car.pca$sdev
```

```
## [1] 1.9816040 0.8438043 0.4749966 0.2878810 0.2296579
```

```
screeplot(car.pca, type ="lines")
```



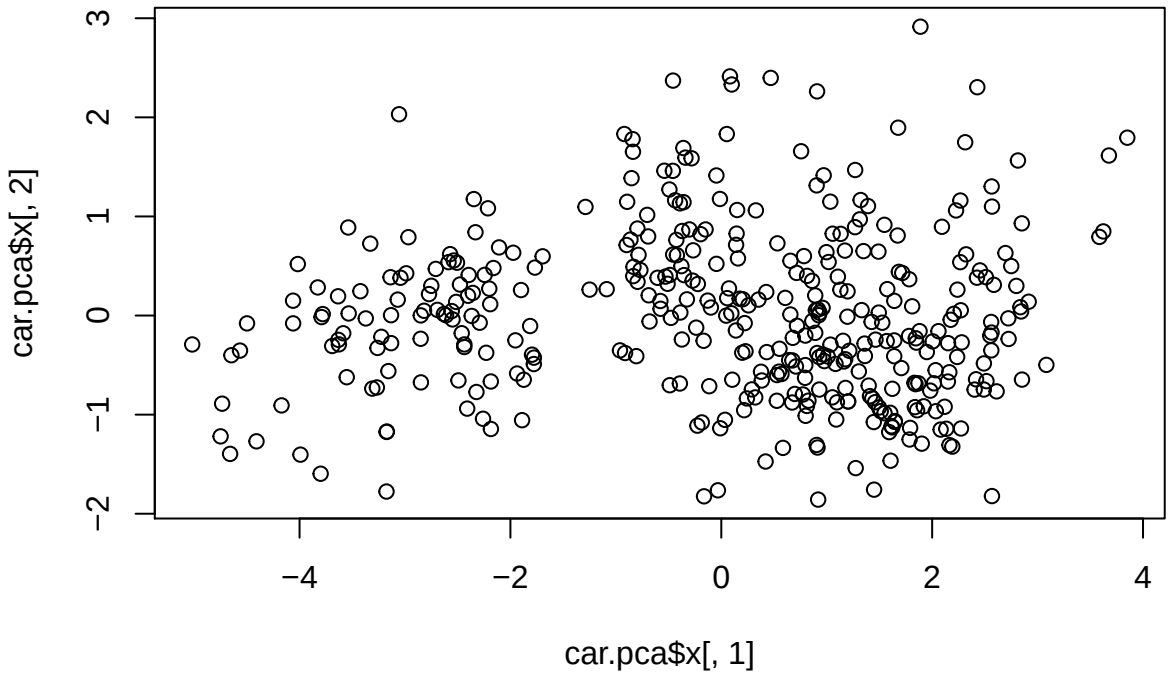

```
car.pca$rotation[,1]
```

```
## displacement   horsepower       weight acceleration          mpg
##   -0.4832332   -0.4844417   -0.4712207    0.3352350    0.4442640
```

```
car.pca$rotation[,2]
```

```
## displacement   horsepower       weight acceleration          mpg
##    0.1347900   -0.1242676    0.3263218    0.8761089   -0.3038692
```

```
plot(car.pca$x[,1],car.pca$x[,2])
```



2.5

```
library(boot)
```

```
##
## Attaching package: 'boot'
##
## The following object is masked from 'package:car':
##
##     logit
```

```
data = c(8,10,7,12,13,8,10,50)
mean_data = function(x,indices){
  return(mean(x[indices]))
}
boot.out = boot(data, mean_data,1000000)
boot.ci(boot.out)
```

```
## Warning in boot.ci(boot.out): bootstrap variances needed for studentized
## intervals
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot.out)
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 5.42, 24.08 )   ( 4.25, 20.88 )
##
```

```
## Level     Percentile            BCa
## 95%   ( 8.62, 25.25 )   ( 9.12, 30.50 )
## Calculations and Intervals on Original Scale
```

We see the resulting confidence intervals:

Normal center is 14.74 and range is 18.66 Basic center is 12.57 and range is 16.63 Percentile center is 16.94 and range is 16.63 BCa center is 19.81 and range is 21.38

As seen from the results of the bootstrapping sample, the range is the same on basic and percentile confidence intervals.

Mini Project

The first step in this miniproject is to load in the data.

```
xdata <- as.matrix(read.table("/Users/bcarancibia/CUNY_IS_605/FinalExam/mini-project-data/ex3x.dat", hea
y <- as.matrix(read.table("/Users/bcarancibia/CUNY_IS_605/FinalExam/mini-project-data/ex3y.dat", header=

names(xdata) <- c("square feet", "number bedrooms")
names(y) <- c("price of house")

iterations=5000

#standardize the x

standard_function <- function(x, mean.val=NA) {
  if(is.matrix(x)) return(apply(x, 2, standard_function, mean.val=mean.val))
    if(is.data.frame(x)) return(data.frame(apply(x, 2, standard_function, mean.val=mean.val)))
    if(is.na(mean.val)) mean.val <- mean(x)
    sd.val <- sd(x)
    if(all(sd.val == 0)) return(x) # if all the values are the same
    (x - mean.val) / sd.val
}

x_scaled <- standard_function(xdata)
```

Add dummy variable to the and then perform gradient descent. First is alpha = 0.001

```
x_scaled <- cbind(1,x_scaled)
x <- x_scaled
names(x) <- c("dummy","square feet", "number bedrooms")


cost <- function(x, y, theta) {
  sum((x %*% theta - y)^2 ) / (2*length(y))
}

alpha <- 0.001
num_iters <- 5000

cost_history <- double(num_iters)
theta_history <- list(num_iters)

theta <- matrix(c(0,0,0), nrow=3)
```

```
for (i in 1:num_iters) {
  error <- (x %*% theta - y)
  delta <- t(x) %*% error / length(y)
  theta <- theta - alpha * delta
  cost_history[i] <- cost(x, y, theta)
  theta_history[[i]] <- theta
}

print(theta)
```

```
##                X3.9990000e.05
##                   336840.1967
## X2.1040000e.03    104566.7034
## X3.0000000e.00       554.9449
```

```
xdata <- as.matrix(read.table("/Users/bcarancibia/CUNY_IS_605/FinalExam/mini-project-data/ex3x.dat", hea
y <- as.matrix(read.table("/Users/bcarancibia/CUNY_IS_605/FinalExam/mini-project-data/ex3y.dat", header=

names(xdata) <- c("square feet", "number bedrooms")
names(y) <- c("price of house")

iterations=5000

#standardize the x

standard_function <- function(x, mean.val=NA) {
  if(is.matrix(x)) return(apply(x, 2, standard_function, mean.val=mean.val))
  if(is.data.frame(x)) return(data.frame(apply(x, 2, standard_function, mean.val=mean.val)))
    if(is.na(mean.val)) mean.val <- mean(x)
    sd.val <- sd(x)
    if(all(sd.val == 0)) return(x) # if all the values are the same
    (x - mean.val) / sd.val
}

x_scaled <- standard_function(xdata)
```

Add dummy variable to the and then perform gradient descent. First is alpha = 0.001

```
x_scaled <- cbind(1,x_scaled)
x <- x_scaled
names(x) <- c("dummy","square feet", "number bedrooms")


cost <- function(x, y, theta) {
  sum((x %*% theta - y)^2 ) / (2*length(y))
}

alpha <- 0.001
num_iters <- 5000

cost_history <- double(num_iters)
theta_history <- list(num_iters)
```

```
theta <- matrix(c(0,0,0), nrow=3)

for (i in 1:num_iters) {
  error <- (x %*% theta - y)
  delta <- t(x) %*% error / length(y)
  theta <- theta - alpha * delta
  cost_history[i] <- cost(x, y, theta)
  theta_history[[i]] <- theta
}

print(theta)
```

```
##                 X3.9990000e.05
##                    336840.1967
## X2.1040000e.03     104566.7034
## X3.0000000e.00        554.9449
```

Alpha = 0.01

```
cost <- function(x, y, theta) {
  sum((x %*% theta - y)^2 ) / (2*length(y))
}

alpha <- 0.01
num_iters <- 5000

cost_history <- double(num_iters)
theta_history <- list(num_iters)

theta <- matrix(c(0,0,0), nrow=3)

for (i in 1:num_iters) {
  error <- (x %*% theta - y)
  delta <- t(x) %*% error / length(y)
  theta <- theta - alpha * delta
  cost_history[i] <- cost(x, y, theta)
  theta_history[[i]] <- theta
}

print(theta)
```

```
##                 X3.9990000e.05
##                     339119.457
## X2.1040000e.03      111467.179
## X3.0000000e.00       -6295.027
```

Alpha = .1

```
cost <- function(x, y, theta) {
  sum((x %*% theta - y)^2 ) / (2*length(y))
}
```

```
alpha <- 0.1
num_iters <- 5000

cost_history <- double(num_iters)
theta_history <- list(num_iters)

theta <- matrix(c(0,0,0), nrow=3)

for (i in 1:num_iters) {
  error <- (x %*% theta - y)
  delta <- t(x) %*% error / length(y)
  theta <- theta - alpha * delta
  cost_history[i] <- cost(x, y, theta)
  theta_history[[i]] <- theta
}

print(theta)
```

```
##                 X3.9990000e.05
##                     339119.457
## X2.1040000e.03      111467.179
## X3.0000000e.00       -6295.027
```

Alpha = 1

```
cost <- function(x, y, theta) {
  sum((x %*% theta - y)^2 ) / (2*length(y))
}

alpha <- 1
num_iters <- 5000

cost_history <- double(num_iters)
theta_history <- list(num_iters)

theta <- matrix(c(0,0,0), nrow=3)

for (i in 1:num_iters) {
  error <- (x %*% theta - y)
  delta <- t(x) %*% error / length(y)
  theta <- theta - alpha * delta
  cost_history[i] <- cost(x, y, theta)
  theta_history[[i]] <- theta
}

print(theta)
```

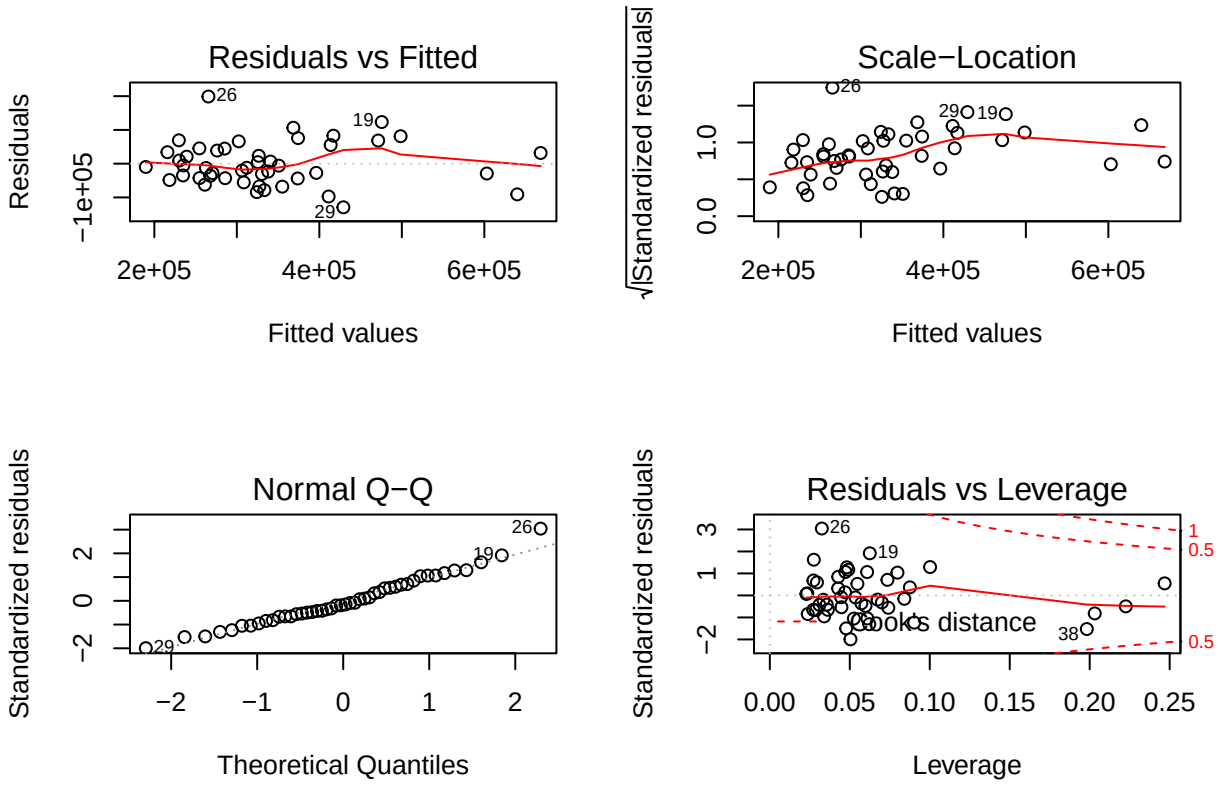
```
##                 X3.9990000e.05
##                     339119.457
## X2.1040000e.03      111467.179
## X3.0000000e.00       -6295.027
```

Do a regular linear regression.

```
xlr <- read.table("/Users/bcarancibia/CUNY_IS_605/FinalExam/mini-project-data/ex3x.dat", header=TRUE)
ylr <- read.table("/Users/bcarancibia/CUNY_IS_605/FinalExam/mini-project-data/ex3y.dat", header=TRUE)

names(xlr) <- c("square_feet", "number_bedrooms")
names(ylr) <- c("price_house")


linear_regression <- lm(ylr$price_house ~ xlr$square_feet + xlr$number_bedrooms)
linear_regression
```

```
##
## Call:
## lm(formula = ylr$price_house ~ xlr$square_feet + xlr$number_bedrooms)
##
## Coefficients:
##         (Intercept)      xlr$square_feet  xlr$number_bedrooms
##             87807.8                138.8              -8186.4
```

```
layout(matrix(c(1,2,3,4),2,2))

plot(linear_regression)
```



Ordinary Least Squares Model

```r
xols <- read.table("/Users/bcarancibia/CUNY_IS_605/FinalExam/mini-project-data/ex3x.dat", header=TRUE)
yols <- read.table("/Users/bcarancibia/CUNY_IS_605/FinalExam/mini-project-data/ex3y.dat", header=TRUE)

names(xols) <- c("square_feet", "number_bedrooms")
names(yols) <- c("price_house")

ols_model <- lm(formula = yols$price_house~xols$square_feet+xols$number_bedrooms)

ols_model
```

```
##
## Call:
## lm(formula = yols$price_house ~ xols$square_feet + xols$number_bedrooms)
##
## Coefficients:
##          (Intercept)      xols$square_feet  xols$number_bedrooms
##              87807.8                 138.8                -8186.4
```

```r
layout(matrix(c(1,2,3,4),2,2))
plot(ols_model)
```