# Homework_4

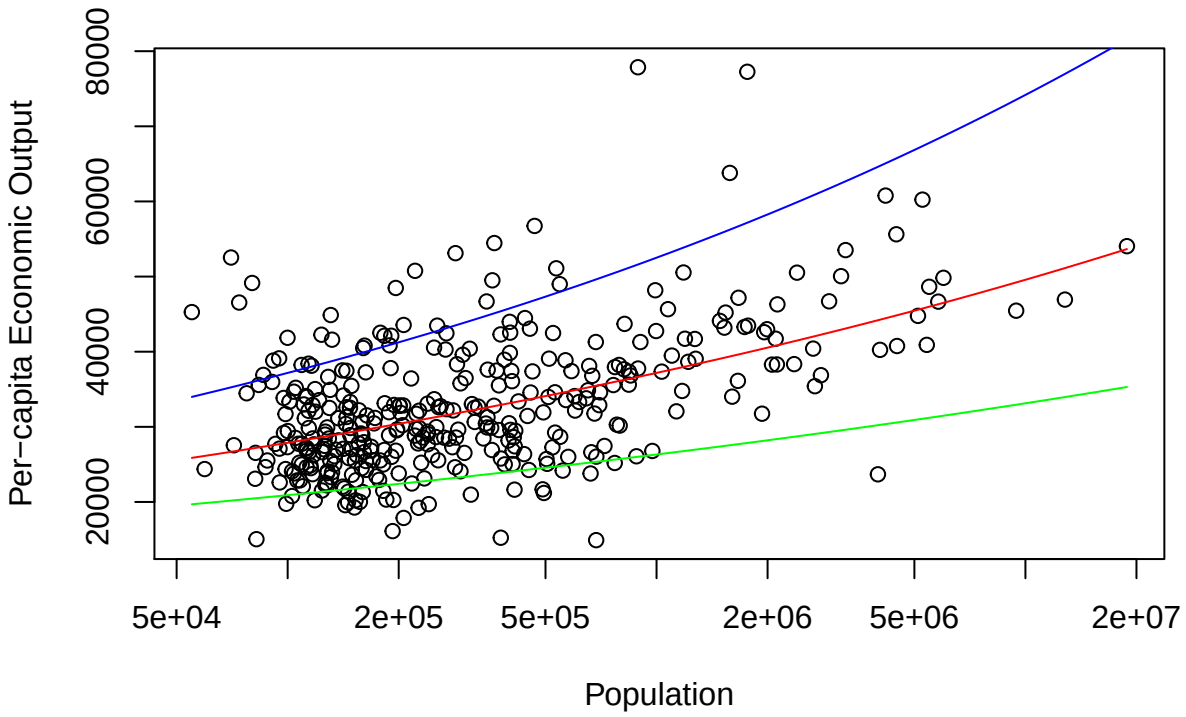
*Christine Chong cc4190*

*June 19, 2017*

```
gmp <- read.table("gmp-1.txt", header = TRUE)
gmp$pop <- gmp$gmp/gmp$pcgmp
```

i. Plot the data as in lecture, with per-capita GMP on the y-axis and population on the xaxis. Using the curve() function to add the model (1) using the default values provided in lecture. Add two more curves using $\beta_1 = 0.1$ and $\beta_1 = 0.15$ with the same value of $\beta_0$ from class. Make all three curves different colors using the col option. Note: you can also use ggplot for the graphs.

```
plot(gmp$pop, gmp$pcgmp, log = "x", xlab = "Population", ylab = "Per-capita Economic Output")
curve(6611*x^{1/8}, add = TRUE, col = "red")
curve(6611*x^{0.1}, add = TRUE, col = "green")
curve(6611*x^{0.15}, add = TRUE, col = "blue")
```




ii. Write a function called mse() which calculates the mean squared error of the model on a given dataset. mse() should have three arguments: (1) a numeric vector of length two, with the first component corresponding to $\beta_0$ and the second to $\beta_1$, (2) a numeric vector containing the population values (X), and (3) a numeric vector containing the values of the per-capita GMP (Y). The output of your function should be a single numeric value (the mean squared error). The second and third arguments should have as default values the variables pop and pcgmp, respectively, from the gmp dataset. Your function may not use a loop (neither for nor which). Check that using the default data your function returns

the following values:

```
mse<- function(betas, pop = gmp$pop, pcGMP = gmp$pcgmp){
  beta0 <- betas[1]
  beta1 <- betas[2]
  result <- mean((pcGMP - beta0*pop^beta1)^2)
  return (result)
}
mse(c(6611, 0.15))
```

```
## [1] 207057513
```

```
mse(c(5000, 0.10))
```

```
## [1] 298459915
```

mse(c(6611, 0.15)) [1] 207057513 mse(c(5000, 0.10)) [1] 298459914

iii. R has several built-in functions for optimization which we'll talk about later on in the course. One of the simplest, which we use today, is nlm(), or non-linear minimization. nlm() takes two required arguments, a function to minimize and a starting value for that functin. Run nlm() three times with your function mse() and three starting pairs for $\beta_0$ and $\beta_1$ as in nlm(mse, c(beta0 = 6611, beta1 = 1/8)) What do the output quantities minmum and estimate represent? Check ?nlm for help. What values does the call return for these? Note that you can access these elements using the dollar sign $''as in the following : nlm(mse, c(beta0 = 6611, beta1 = 1/8))$estimate Hint: In the minimization you may return an error message about replacing NA/Inf values. That's ok as long as the minimization still worked. When you write your solutions in Markdown, surround your code chunks with the following: There should be no printed warnings in your .html file! In my minimization, it was rare that the $\beta_0$ value moved much from its initial guess.

```
first_nlm<- nlm(mse, c(beta0 = 6611, beta1 = 1/8))
second_nlm<- nlm(mse, c(beta0 = 5000, beta1 = .10))
third_nlm<- nlm(mse, c(beta0 = 4000, beta1 = .5))
first_nlm_estimate<- nlm(mse, c(beta0 = 6611, beta1 = 1/8))$estimate
first_nlm_min<- nlm(mse, c(beta0 = 6611, beta1 = 1/8))$minimum
first_nlm
```

```
## $minimum
## [1] 61857061
##
## $estimate
## [1] 6611.0000000    0.1263177
##
## $gradient
## [1] 50.048517 -9.983778
##
## $code
## [1] 2
##
## $iterations
## [1] 3
```

```
second_nlm
```

```
## $minimum
## [1] 62521485
##
## $estimate
```

```
## [1] 5000.0000008    0.1475913
##
## $gradient
## [1] -1028.22561    11.38672
##
## $code
## [1] 2
##
## $iterations
## [1] 5
```

```
third_nlm
```

```
## $minimum
## [1] 1168662933
##
## $estimate
## [1] 3999.9965 -201.9333
##
## $gradient
## [1] 0 0
##
## $code
## [1] 1
##
## $iterations
## [1] 1
```

```
first_nlm_estimate
```

```
## [1] 6611.0000000    0.1263177
```

```
first_nlm_min
```

```
## [1] 61857061
```

The output quantity estimate seems to output the approximate betas that were inputted in based on the mse. The minimum seems to be the minimum in which the mse function works at. iv. Using nlm() and the mse() function you wrote, write a function plm() which estimates the parameters $\beta_0$ and $\beta_1$ of the model by minimizing the mean squared error. The function plm() should take the following four arguments: (1) an initial guess for $\beta_0$, (2) an initial guess for $\beta_1$, (3) a vector containing the population values (X), and (4) a vector containing the per-capita GMP values (Y). All arguments except for the initial guesses should have suitable default values. It should return a list with the following three components: (1) the final guess for $\beta_0$, (2) the final guess for $\beta_1$, and (3) the final value of the MSE. Your function must call those you wrote in earlier questions (as opposed to simply repeating the code) and the appropriate arguments to plm() should be passed on to them. What parameter estimate do you get when starting from $\beta_0 = 6611$ and $\beta_1 = 0.15$? From $\beta_0 = 5000$ and $\beta_1 = 0.10$? If these are not the same, why do they differ? Which estimate has a lower MSE?

```
plm<-function(b0, b1, popVec = gmp$pop, pcGMPVec = gmp$pcgmp){
  b0 <- b0
  b1 <- b1
  initial_nlm<- nlm(mse, c(beta0 = b0, beta1 = b1), popVec, pcGMPVec)
  minimum<- initial_nlm$minimum
  estimate <- initial_nlm$estimate
  nb0 <- estimate[1]
  nb1 <- estimate[2]
    return(c(nb0,nb1,minimum))
```

```
}
plm(6611, 0.15)
```

```
## [1] 6.611000e+03 1.263182e-01 6.185706e+07
```

```
plm(5000, 0.10)
```

```
## [1] 5.000000e+03 1.475913e-01 6.252148e+07
```

The input where beta 0 is 6601 and b1 is 0.15 has the lower MSE. These should differ slightly because they are testing different beta 0 and beta 1.

v. Let's practice the bootstrap in a simple example to convince ourselves, again, that it will work.

(a) Calculate the mean and standard deviation of the per-capita GMP values in your dataset using built-in function mean() and sd(). Using these values calculate the standard error of the mean.

```
  pcGMP_mean<- mean(gmp$pcgmp)
  pcGMP_sd<- sd(gmp$pcgmp)
  gmp_length <- length(gmp$pcgmp)
  standardErr <- pcGMP_sd/ sqrt(gmp_length)
  standardErr
```

```
## [1] 481.9195
```

(b) Write a function which takes in a vector indices of length n, where n is the number of per-capita GMP values in our dataset, and calculates the mean percapita GMP for the cities indicated by the indices. For example if indices = c(1, 5, 1, 3), then your function should calculate the mean c(24490, 37657, 24490, 24269), the per-capita GMP for the first, fifth, first again, and third cities.

```
cityMeanGMP<- function(index){
  max <- length(index)
  cities <- c()
  for(i in 1:max){
   cities[i] <- gmp[index[i],"pcgmp"]
  }
 mCities <- mean(cities)
 return (mCities)
}
cityMeanGMP(c(1,2,3))
```

```
## [1] 27216
```

(c) Using the function in (b) create a vector bootstrap.means, which has the mean per-capita GMP for one hundred bootstrap samples. Here you'll want to create a loop where each iteration performs a new bootstrap sample. In each iteration you use sample() to create an indices vector containing the indices of your bootstrap sample and then using indices calculate the mean using your function from (b).

```
B<- 100
n <- nrow(gmp)
bootstrap.means <- rep(NA,B)
for(b in 1:B){
  boot_vals<- sample(1:n, n, replace = TRUE)
  bootstrap.means[b] <- cityMeanGMP(boot_vals)

}
head(bootstrap.means, 5)
```

```
## [1] 33472.43 33228.13 32624.85 33427.66 32931.19
```

(d) Calculate the standard deviation of the bootstrap.means to approximate the 3 standard error from part (a). How well does this estimate match the value form part (a)?

```
bootsd<- sd(bootstrap.means)
bootsd
```

```
## [1] 464.8735
```

```
SE<- sd(gmp$pcgmp)/sqrt(gmp_length)
SE
```

```
## [1] 481.9195
```

vi. Write a function plm.bootstrap(), then calculate a bootstrap estimates of the standard errors of $\beta_0$ and $\beta_1$. It should take the same arguments as plm() along with an argument B for the number of bootstrap samples to draw. Let the default be B = 100. plm.bootstrap() should return standard errors for both parameters. This function should call your plm() function repeatedly. What standard errors do you get for the two parameters using initial conditions $\beta_0 = 6611$ and $\beta_1 = 0.15$? Initial conditions $\beta_0 = 5000$ and $\beta_1 = 0.10$

```
plm.bootstrap<-function(b0, b1, popVec = gmp$pop, pcGMPVec = gmp$pcgmp, B = 100){
  b0 <- b0
  b1 <- b1
  n <- nrow(gmp)
  boot_vals <- matrix(NA, nrow=B, ncol = n)
  for (b in 1:B){
    boot_vals[b, ] <- sample(1:n,n, replace=TRUE)
    resampled_rows <- boot_vals[b, ]
    resampled_data <- gmp[resampled_rows, ]
  }
  bootPopVec = resampled_data$pop
  bootPopGMPVec = resampled_data$pcgmp
  initial_nlm<- nlm(mse, c(beta0 = b0, beta1 = b1), bootPopVec, bootPopGMPVec)
  minimum<- initial_nlm$minimum
  estimate <- initial_nlm$estimate
  nb0 <- estimate[1]
  nb1 <- estimate[2]
  bootSE<- sd(bootPopGMPVec)/sqrt(n)
    return(c(nb0,nb1,minimum, bootSE))

}
plm.bootstrap(6611, 0.15)
```

```
## [1] 6.611000e+03 1.263594e-01 5.903411e+07 4.684031e+02
```

```
plm.bootstrap(5000, 0.10)
```

```
## [1] 5.000000e+03 1.488525e-01 7.405329e+07 5.145483e+02
```

The standard errors you get are around 480-500. These standard erros are close to the original ones. The estimates for B0 and B1 seem to be very close to the original along with the minimum.

vii. The file gmp-2013.txt contains measurements for 2013 (in contrast to measurements from 2006 in gmp.txt). Load it and use plm() and plm.bootstrap() to estimate the parameters for the model for 2013 and their standard errors. Have the parameters of the model changed significantly?

```
gmp2013 <- read.table("gmp-2013.txt", header = TRUE)
gmp2013$pop <- gmp2013$gmp/gmp2013$pcgmp
plm(6611, 0.15, gmp2013$pop, gmp2013$pcgmp)
```

```
## [1] 6.611000e+03 1.433688e-01 1.352105e+08
```

```
plm(5000, 0.10, gmp2013$pop, gmp2013$pcgmp )
```

```
## [1] 5.000000e+03 1.644270e-01 1.392087e+08
```

It seems like while the beta 0s are relatively the same, the beta 1s and the minimum have changed.The beta 1s have changed slightly but the minimums have changed siginificantly.(Previously the minimums were e+07 now they are e+08.)