1 Function Approximation Warmup

1.1 Exploring and downloading the data

```
rm(list=ls())           # Clear the workspace
set.seed(20866)
library(ggplot2)
library(sandwich)
library(car)
library(xtable)
library(aod)
library(systemfit)

## Loading required package:  Matrix
## Loading required package:  lmtest
## Loading required package:  zoo
##
## Attaching package:  'zoo'
##
## The following objects are masked from 'package:base':
##
##    as.Date, as.Date.numeric

library(MASS)
library(stargazer)

##
## Please cite as:
##
##  Hlavac, Marek (2014).  stargazer:  LaTeX code and ASCII text for
well-formatted regression and summary statistics tables.
##  R package version 5.1.  http://CRAN.R-project.org/package=stargazer

setwd("/Users/Tony/Downloads")

data <- read.csv("cps_00005.csv")
datamatrix <- as.matrix(read.csv("cps_00005.csv"))
datamatrix <- datamatrix[,-5:-8]
datamatrix <- datamatrix[,-2:-3]

AdjInc <- c(rep(NA, nrow(datamatrix)))

datamatrix <- cbind(datamatrix, AdjInc)

datamatrix <- datamatrix[datamatrix[,9]!= 0,]
datamatrix <- datamatrix[datamatrix[,9]!= 9999999,]
```

```r
incomeadjust <- function(data.m = datamatrix){

  for (i in 1:nrow(datamatrix)){

    year <- as.numeric(datamatrix[i,1])
    income <- as.numeric(datamatrix[i,9])

    if (year == 2004){

      AdjustedIncome <- income * 1.25
      datamatrix[i,10] = round(AdjustedIncome)
    }

    if (year == 2014){

      AdjustedIncome <- income
      datamatrix[i,10] = round(AdjustedIncome)
    }


  }


  top <- head(datamatrix, n=5)
  bottom<- tail(datamatrix, n=5)

  sample <- rbind(top,bottom)
  return(sample)

}

incomeadjust(datamatrix)
```

```
##            YEAR REGION AGE SEX RACE EDUC99 EMPSTAT HRSWORK INCWAGE AdjInc
##            2004     11  59   2  100     13      10       2   60000  75000
##            2004     11  49   1  100     10      10      20   32000  40000
##            2004     11  42   2  100     15      10      40   30000  37500
##            2004     11  68   2  100     15      10      20   18000  22500
##            2004     11  42   2  100     10      10      24   30000  37500
## [168263,] 2014     42  26   1  652     10      10      32   39000  39000
## [168264,] 2014     42  20   1  652     10      30       0    3480   3480
## [168265,] 2014     42  36   2  100     13      21       0   55300  55300
## [168266,] 2014     42  47   1  807     10      32       0   35000  35000
```

```
## [168267,] 2014     42  21    2  807     11      10      19   10300  10300
```

To find the CPI, I used the Bureau of Labor Statistics CPI Inflation Calculator, which told me that a dollar in 2004 has the same buying power as $1.25 in 2014. $Therefore, to adjust 2004 income to its 2014 equivalent, I wrote a function that multiplied all 2004 incom$

1.2 Make a new variable that is log wage income in your data

```
sample <- incomeadjust(datamatrix)
logVar <- c(rep(NA, nrow(sample)))
sample <- cbind(sample, logVar)



logVarf <- function(data.m = sample){

  for (i in 1:nrow(sample)){

    rowIncomeLog <- log(sample[i,10])
    sample[i,11] <- rowIncomeLog

  }

  ## return(datamatrix) Commenting out so it doesn't actually return this
  print(sample)

}

logVarf(sample)
```

```
##            YEAR REGION AGE SEX RACE EDUC99 EMPSTAT HRSWORK INCWAGE AdjInc
##            2004     11  59   2  100     13      10       2   60000  75000
##            2004     11  49   1  100     10      10      20   32000  40000
##            2004     11  42   2  100     15      10      40   30000  37500
##            2004     11  68   2  100     15      10      20   18000  22500
##            2004     11  42   2  100     10      10      24   30000  37500
## [168263,] 2014     42  26   1  652     10      10      32   39000  39000
## [168264,] 2014     42  20   1  652     10      30       0    3480   3480
## [168265,] 2014     42  36   2  100     13      21       0   55300  55300
## [168266,] 2014     42  47   1  807     10      32       0   35000  35000
## [168267,] 2014     42  21   2  807     11      10      19   10300  10300
##              logVar
##            11.225243
```

```
##           10.596635
##           10.532096
##           10.021271
##           10.532096
## [168263,] 10.571317
## [168264,]  8.154788
## [168265,] 10.920528
## [168266,] 10.463103
## [168267,]  9.239899
```

1.3 Construct "potential experience", which will be "Age - years of schooling - 5"

```
sample <- logVarf(sample)

##           YEAR REGION AGE SEX RACE EDUC99 EMPSTAT HRSWORK INCWAGE AdjInc
##           2004     11  59   2  100     13      10       2   60000  75000
##           2004     11  49   1  100     10      10      20   32000  40000
##           2004     11  42   2  100     15      10      40   30000  37500
##           2004     11  68   2  100     15      10      20   18000  22500
##           2004     11  42   2  100     10      10      24   30000  37500
## [168263,] 2014     42  26   1  652     10      10      32   39000  39000
## [168264,] 2014     42  20   1  652     10      30       0    3480   3480
## [168265,] 2014     42  36   2  100     13      21       0   55300  55300
## [168266,] 2014     42  47   1  807     10      32       0   35000  35000
## [168267,] 2014     42  21   2  807     11      10      19   10300  10300
##              logVar
##           11.225243
##           10.596635
##           10.532096
##           10.021271
##           10.532096
## [168263,] 10.571317
## [168264,]  8.154788
## [168265,] 10.920528
## [168266,] 10.463103
## [168267,]  9.239899

potExp <- c(rep(NA, nrow(sample)))
sample <- cbind(sample, potExp)

potExpf <- function(data.m = sample){

  for (i in 1:nrow(sample)){
```

```
indAge = as.numeric(sample[i,3])
indEduCode = as.numeric(sample[i,6])

if (indEduCode < 6){
  indYrsOfSch = 9
  indPotExp = indAge - indYrsOfSch - 5
  sample[i,12] = indPotExp

}

if (indEduCode == 6){
  indYrsOfSch = 10
  indPotExp = indAge - indYrsOfSch - 5
  sample[i,12] = indPotExp

}

if (indEduCode == 7){
  indYrsOfSch = 11
  indPotExp = indAge - indYrsOfSch - 5
  sample[i,12] = indPotExp

}

if (indEduCode == 8){
  indYrsOfSch = 12
  indPotExp = indAge - indYrsOfSch - 5
  sample[i,12] = indPotExp

}

if (indEduCode == 9){
  indYrsOfSch = 13
  indPotExp = indAge - indYrsOfSch - 5
  sample[i,12] = indPotExp

}

if (indEduCode == 10){
  indYrsOfSch = 13
  indPotExp = indAge - indYrsOfSch - 5
  sample[i,12] = indPotExp

}
```

```
    if (indEduCode == 11){
      indYrsOfSch = 14
      indPotExp = indAge - indYrsOfSch - 5
      sample[i,12] = indPotExp

    }

    if (indEduCode == 12){
      indYrsOfSch = 15
      indPotExp = indAge - indYrsOfSch - 5
      sample[i,12] = indPotExp

    }

    if (indEduCode == 13){
      indYrsOfSch = 15
      indPotExp = indAge - indYrsOfSch - 5
      sample[i,12] = indPotExp

    }

    if (indEduCode == 14){
      indYrsOfSch = 15
      indPotExp = indAge - indYrsOfSch - 5
      sample[i,12] = indPotExp

    }

    if (indEduCode == 15){
      indYrsOfSch = 17
      indPotExp = indAge - indYrsOfSch - 5
      sample[i,12] = indPotExp

    }

    if (indEduCode == 16){
      indYrsOfSch = 19
      indPotExp = indAge - indYrsOfSch - 5
      sample[i,12] = indPotExp

    }

    if (indEduCode == 17){
      indYrsOfSch = 19
      indPotExp = indAge - indYrsOfSch - 5
```

```
      sample[i,12] = indPotExp

    }

    if (indEduCode == 18){
      indYrsOfSch = 22
      indPotExp = indAge - indYrsOfSch - 5
      sample[i,12] = indPotExp

    }

  }

  ## return(datamatrix) Commenting out so it doesn't actually return this
  print(sample)

}

potExpf(sample)
```

```
##            YEAR REGION AGE SEX RACE EDUC99 EMPSTAT HRSWORK INCWAGE AdjInc
##            2004     11  59   2  100     13      10       2   60000  75000
##            2004     11  49   1  100     10      10      20   32000  40000
##            2004     11  42   2  100     15      10      40   30000  37500
##            2004     11  68   2  100     15      10      20   18000  22500
##            2004     11  42   2  100     10      10      24   30000  37500
## [168263,] 2014     42  26   1  652     10      10      32   39000  39000
## [168264,] 2014     42  20   1  652     10      30       0    3480   3480
## [168265,] 2014     42  36   2  100     13      21       0   55300  55300
## [168266,] 2014     42  47   1  807     10      32       0   35000  35000
## [168267,] 2014     42  21   2  807     11      10      19   10300  10300
##               logVar potExp
##            11.225243     39
##            10.596635     31
##            10.532096     20
##            10.021271     46
##            10.532096     24
## [168263,] 10.571317      8
## [168264,]  8.154788      2
## [168265,] 10.920528     16
## [168266,] 10.463103     29
## [168267,]  9.239899      2
```