1 Function Approximation Warmup

1.1 Exploring and downloading the data

```
rm(list=ls())           # Clear the workspace
set.seed(20866)
library("knitr")
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

opts_chunk$set(out.width ='\\textwidth')

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]


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



incomeadjust <- function(data.m = datamatrix, sampq = TRUE){

  AdjInc <- c(rep(NA, nrow(data.m)))
  data.m <- cbind(data.m, AdjInc)

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

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

    if (year == 2004){

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

    if (year == 2014){
      AdjustedIncome <- income
      data.m[i,10] = round(AdjustedIncome)
    }


  }

 if (sampq == TRUE){
  top <- head(data.m, n=15)
  bottom<- tail(data.m, n=15)

  sample1 <- rbind(top,bottom)
  row.names(sample1) <- NULL
  return(sample1)

 }

 if (sampq == FALSE){
   return(data.m)
 }
}

incomeadjust(datamatrix, TRUE)

##       YEAR REGION AGE SEX RACE EDUC99 EMPSTAT HRSWORK INCWAGE AdjInc
##  [1,] 2004     11  59   2  100     13      10       2   60000  75000
##  [2,] 2004     11  49   1  100     10      10      20   32000  40000
##  [3,] 2004     11  42   2  100     15      10      40   30000  37500
##  [4,] 2004     11  68   2  100     15      10      20   18000  22500
##  [5,] 2004     11  42   2  100     10      10      24   30000  37500
##  [6,] 2004     11  45   1  100     13      10      33   50000  62500
##  [7,] 2004     11  20   1  100     10      30       0   15000  18750
##  [8,] 2004     11  19   1  100     10      10      44   18000  22500
##  [9,] 2004     11  18   2  100      8      10      20   10000  12500
## [10,] 2004     11  59   2  100      8      10      25   20285  25356
## [11,] 2004     11  74   1  100     15      10      26   19000  23750
## [12,] 2004     11  73   2  100     14      10      32   24250  30312
## [13,] 2004     11  71   2  802     11      32       0    5270   6588
## [14,] 2004     11  47   2  802     17      10      30   20900  26125
## [15,] 2004     11  36   1  100     10      10      19   26048  32560
## [16,] 2014     42  58   1  651     11      10      40   50000  50000
## [17,] 2014     42  30   2  652     16      10      40   25000  25000
## [18,] 2014     42  30   1  652     13      12       0    5000   5000
## [19,] 2014     42  48   1  651     10      10      50   43160  43160
## [20,] 2014     42  42   2  651     10      10      80   55120  55120
## [21,] 2014     42  35   1  802     10      10      40   24000  24000
## [22,] 2014     42  50   1  804     10      10      40   14000  14000
## [23,] 2014     42  39   1  651     15      10      40   27000  27000
## [24,] 2014     42  26   1  651     10      10      15   18000  18000
## [25,] 2014     42  24   2  651     17      10      40   60000  60000
## [26,] 2014     42  26   1  652     10      10      32   39000  39000
## [27,] 2014     42  20   1  652     10      30       0    3480   3480
## [28,] 2014     42  36   2  100     13      21       0   55300  55300
## [29,] 2014     42  47   1  807     10      32       0   35000  35000
## [30,] 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 $1.25in2014.Therefore,toadjust2004incometoits2014equivalent,Iwroteafunctionthatmultipliedall2004incom$

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

```r
testdata <- incomeadjust(datamatrix, TRUE)

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

  logInc <- c(rep(NA, nrow(data.m)))
  data.m <- cbind(data.m, logInc)

  for (i in 1:nrow(data.m)){

    rowIncomeLog <- log(data.m[i,10])
    data.m[i,11] <- rowIncomeLog

  }

  ## return(datamatrix) Commenting out so it doesn't actually return this
  return(data.m)

}

logVarf(testdata)
```

```
##       YEAR REGION AGE SEX RACE EDUC99 EMPSTAT HRSWORK INCWAGE AdjInc
##  [1,] 2004     11  59   2  100     13      10       2   60000  75000
##  [2,] 2004     11  49   1  100     10      10      20   32000  40000
##  [3,] 2004     11  42   2  100     15      10      40   30000  37500
##  [4,] 2004     11  68   2  100     15      10      20   18000  22500
##  [5,] 2004     11  42   2  100     10      10      24   30000  37500
##  [6,] 2004     11  45   1  100     13      10      33   50000  62500
##  [7,] 2004     11  20   1  100     10      30       0   15000  18750
##  [8,] 2004     11  19   1  100     10      10      44   18000  22500
##  [9,] 2004     11  18   2  100      8      10      20   10000  12500
## [10,] 2004     11  59   2  100      8      10      25   20285  25356
## [11,] 2004     11  74   1  100     15      10      26   19000  23750
## [12,] 2004     11  73   2  100     14      10      32   24250  30312
## [13,] 2004     11  71   2  802     11      32       0    5270   6588
## [14,] 2004     11  47   2  802     17      10      30   20900  26125
## [15,] 2004     11  36   1  100     10      10      19   26048  32560
## [16,] 2014     42  58   1  651     11      10      40   50000  50000
## [17,] 2014     42  30   2  652     16      10      40   25000  25000
## [18,] 2014     42  30   1  652     13      12       0    5000   5000
## [19,] 2014     42  48   1  651     10      10      50   43160  43160
## [20,] 2014     42  42   2  651     10      10      80   55120  55120
## [21,] 2014     42  35   1  802     10      10      40   24000  24000
## [22,] 2014     42  50   1  804     10      10      40   14000  14000
## [23,] 2014     42  39   1  651     15      10      40   27000  27000
## [24,] 2014     42  26   1  651     10      10      15   18000  18000
## [25,] 2014     42  24   2  651     17      10      40   60000  60000
## [26,] 2014     42  26   1  652     10      10      32   39000  39000
## [27,] 2014     42  20   1  652     10      30       0    3480   3480
## [28,] 2014     42  36   2  100     13      21       0   55300  55300
## [29,] 2014     42  47   1  807     10      32       0   35000  35000
## [30,] 2014     42  21   2  807     11      10      19   10300  10300
##          logInc
##  [1,] 11.225243
##  [2,] 10.596635
##  [3,] 10.532096
##  [4,] 10.021271
##  [5,] 10.532096
##  [6,] 11.042922
##  [7,]  9.838949
##  [8,] 10.021271
##  [9,]  9.433484
## [10,] 10.140771
## [11,] 10.075338
## [12,] 10.319299
## [13,]  8.793005
## [14,] 10.170648
## [15,] 10.390840
## [16,] 10.819778
## [17,] 10.126631
## [18,]  8.517193
## [19,] 10.672669
## [20,] 10.917268
## [21,] 10.085809
## [22,]  9.546813
## [23,] 10.203592
## [24,]  9.798127
## [25,] 11.002100
## [26,] 10.571317
## [27,]  8.154788
## [28,] 10.920528
## [29,] 10.463103
## [30,]  9.239899
```

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

```
sample1 <- logVarf(testdata)

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

  potExp <- c(rep(NA, nrow(data.m)))
  YrsOfSch <- c(rep(NA, nrow(data.m)))
  data.m <- cbind(data.m, potExp, YrsOfSch)

  for (i in 1:nrow(data.m)){

    indAge = as.numeric(data.m[i,3])
    indEduCode = as.numeric(data.m[i,6])

    if (indEduCode < 6){
      indYrsOfSch = 9
      indPotExp = indAge - indYrsOfSch - 5
      data.m[i,12] = indPotExp
      data.m[i,13] = indYrsOfSch

    }

    if (indEduCode == 6){
      indYrsOfSch = 10
      indPotExp = indAge - indYrsOfSch - 5
      data.m[i,12] = indPotExp
      data.m[i,13] = indYrsOfSch

    }

    if (indEduCode == 7){
      indYrsOfSch = 11
      indPotExp = indAge - indYrsOfSch - 5
      data.m[i,12] = indPotExp
      data.m[i,13] = indYrsOfSch

    }

    if (indEduCode == 8){
      indYrsOfSch = 12
      indPotExp = indAge - indYrsOfSch - 5
      data.m[i,12] = indPotExp
      data.m[i,13] = indYrsOfSch

    }
    if (indEduCode == 9){
      indYrsOfSch = 13
      indPotExp = indAge - indYrsOfSch - 5
      data.m[i,12] = indPotExp
      data.m[i,13] = indYrsOfSch

    }

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

    }

    if (indEduCode == 11){
      indYrsOfSch = 14
      indPotExp = indAge - indYrsOfSch - 5
      data.m[i,12] = indPotExp
      data.m[i,13] = indYrsOfSch

    }

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

    }

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

    }

    if (indEduCode == 14){
      indYrsOfSch = 15
      indPotExp = indAge - indYrsOfSch - 5
      data.m[i,12] = indPotExp
      data.m[i,13] = indYrsOfSch

    }

    if (indEduCode == 15){
      indYrsOfSch = 17
      indPotExp = indAge - indYrsOfSch - 5
      data.m[i,12] = indPotExp
      data.m[i,13] = indYrsOfSch

    }

    if (indEduCode == 16){
      indYrsOfSch = 19
      indPotExp = indAge - indYrsOfSch - 5
      data.m[i,12] = indPotExp
      data.m[i,13] = indYrsOfSch

    }

    if (indEduCode == 17){
      indYrsOfSch = 19
      indPotExp = indAge - indYrsOfSch - 5
      data.m[i,12] = indPotExp
      data.m[i,13] = indYrsOfSch

    }

    if (indEduCode == 18){
      indYrsOfSch = 22
      indPotExp = indAge - indYrsOfSch - 5
      data.m[i,12] = indPotExp
      data.m[i,13] = indYrsOfSch

    }

  }

  ## return(datamatrix) Commenting out so it doesn't actually return this
  return(data.m)

}

potExpf(sample1)
```

```
##       YEAR REGION AGE SEX RACE EDUC99 EMPSTAT HRSWORK INCWAGE AdjInc
##  [1,] 2004     11  59   2  100     13      10       2   60000  75000
##  [2,] 2004     11  49   1  100     10      10      20   32000  40000
##  [3,] 2004     11  42   2  100     15      10      40   30000  37500
##  [4,] 2004     11  68   2  100     15      10      20   18000  22500
##  [5,] 2004     11  42   2  100     10      10      24   30000  37500
##  [6,] 2004     11  45   1  100     13      10      33   50000  62500
##  [7,] 2004     11  20   1  100     10      30       0   15000  18750
##  [8,] 2004     11  19   1  100     10      10      44   18000  22500
##  [9,] 2004     11  18   2  100      8      10      20   10000  12500
## [10,] 2004     11  59   2  100      8      10      25   20285  25356
## [11,] 2004     11  74   1  100     15      10      26   19000  23750
## [12,] 2004     11  73   2  100     14      10      32   24250  30312
## [13,] 2004     11  71   2  802     11      32       0    5270   6588
## [14,] 2004     11  47   2  802     17      10      30   20900  26125
## [15,] 2004     11  36   1  100     10      10      19   26048  32560
## [16,] 2014     42  58   1  651     11      10      40   50000  50000
## [17,] 2014     42  30   2  652     16      10      40   25000  25000
## [18,] 2014     42  30   1  652     13      12       0    5000   5000
## [19,] 2014     42  48   1  651     10      10      50   43160  43160
## [20,] 2014     42  42   2  651     10      10      80   55120  55120
## [21,] 2014     42  35   1  802     10      10      40   24000  24000
## [22,] 2014     42  50   1  804     10      10      40   14000  14000
## [23,] 2014     42  39   1  651     15      10      40   27000  27000
## [24,] 2014     42  26   1  651     10      10      15   18000  18000
## [25,] 2014     42  24   2  651     17      10      40   60000  60000
## [26,] 2014     42  26   1  652     10      10      32   39000  39000
## [27,] 2014     42  20   1  652     10      30       0    3480   3480
## [28,] 2014     42  36   2  100     13      21       0   55300  55300
## [29,] 2014     42  47   1  807     10      32       0   35000  35000
## [30,] 2014     42  21   2  807     11      10      19   10300  10300
##          logInc potExp YrsOfSch
##  [1,] 11.225243     39       15
##  [2,] 10.596635     31       13
##  [3,] 10.532096     20       17
##  [4,] 10.021271     46       17
##  [5,] 10.532096     24       13
##  [6,] 11.042922     25       15
##  [7,]  9.838949      2       13
##  [8,] 10.021271      1       13
##  [9,]  9.433484      1       12
## [10,] 10.140771     42       12
## [11,] 10.075338     52       17
## [12,] 10.319299     53       15
## [13,]  8.793005     52       14
## [14,] 10.170648     23       19
## [15,] 10.390840     18       13
## [16,] 10.819778     39       14
## [17,] 10.126631      6       19
## [18,]  8.517193     10       15
## [19,] 10.672669     30       13
## [20,] 10.917268     24       13
## [21,] 10.085809     17       13
## [22,]  9.546813     32       13
## [23,] 10.203592     17       17
## [24,]  9.798127      8       13
## [25,] 11.002100      0       19
## [26,] 10.571317      8       13
## [27,]  8.154788      2       13
## [28,] 10.920528     16       15
## [29,] 10.463103     29       13
## [30,]  9.239899      2       14
```

1.4 Make a table comparing the following regressions for 2014 and 2014

```
library(stargazer)

regData <- incomeadjust(datamatrix, FALSE)
regData <- logVarf(regData)
regData <- potExpf(regData)

regData <- as.data.frame(regData)

data04 <- regData[regData[,1]== 2004,]
data14 <- regData[regData[,1]== 2014,]
data04m <- data04[data04[,4]== 1,]
data04f <- data04[data04[,4]== 2,]
data14m <- data14[data14[,4]== 1,]
data14f <- data14[data14[,4]== 2,]

fit04 <- lm(data04$logInc ~ data04$YrsOfSch + data04$potExp
          + I(data04$potExp^2), data=data04 )

fit04m <- lm(data04m$logInc ~ data04m$YrsOfSch + data04m$potExp
          + I(data04m$potExp^2), data=data04m )

fit04f <- lm(data04f$logInc ~ data04f$YrsOfSch + data04f$potExp
          + I(data04f$potExp^2), data=data04f )

fit14 <- lm(data14$logInc ~ data14$YrsOfSch + data14$potExp
          + I(data14$potExp^2), data=data14 )

fit14m <- lm(data14m$logInc ~ data14m$YrsOfSch + data14m$potExp
          + I(data14m$potExp^2), data=data14m )

fit14f <- lm(data14f$logInc ~ data14f$YrsOfSch + data14f$potExp
          + I(data14f$potExp^2), data=data14f)
```

Table 1: Regressing Income on Yrs. of School and Exp.

	LogAdjInc	LogAdjInc	LogAdjInc
	(2004)	(2004 Males)	(2004 Females)
YrsOfSch	0.163***	0.160***	0.171***
	(0.001)	(0.002)	(0.002)
potExp	0.106***	0.120***	0.091***
	(0.001)	(0.001)	(0.001)
potExpˆ2	$-$0.002***	$-$0.002***	$-$0.002***
	(0.00002)	(0.00002)	(0.00002)
Constant	6.719***	6.869***	6.501***
	(0.019)	(0.024)	(0.030)
Observations	103,084	52,848	50,236
R^2	0.310	0.385	0.266
Adjusted R^2	0.310	0.385	0.266
Residual Std. Error	1.026 (df = 103080)	0.942 (df = 52844)	1.045 (df = 50232)
F Statistic	15,448.400*** (df = 3; 103080)	11,041.710*** (df = 3; 52844)	6,081.976*** (df = 3; 50232)
Note:			*p<0.1; **p<0.05; ***p<0.01

2 Function Approximation

2.1 Model for four-fifths of data, test on other fifth

```
fourfifthsdata <- regData[sample(nrow(regData),size =((4/5) * (nrow(regData))),replace=FALSE),]
fourfifthstop <- head(fourfifthsdata, n=10)
print(fourfifthstop)

##        YEAR REGION AGE SEX RACE EDUC99 EMPSTAT HRSWORK INCWAGE AdjInc
## 118566 2014     21  27   2  100      5      10      40    8000   8000
## 65939  2004     32  45   2  200     15      10      40   48000  60000
## 68036  2004     32  39   2  100     15      10      40   29000  36250
## 34552  2004     21  30   2  100     11      10      40   53000  66250
## 145324 2014     33  21   1  100     11      32       0   33000  33000
## 81157  2004     41  29   2  100     13      10      40   28000  35000
## 156704 2014     41  41   2  100     11      10      40   45000  45000
## 132185 2014     31  51   1  100     11      10      40   35000  35000
## 168216 2014     42  42   2  804     14      10      50  110000 110000
## 150695 2014     33  61   1  100      5      10      40   12000  12000
##           logInc potExp YrsOfSch
## 118566  8.987197     13        9
## 65939  11.002100     23       17
## 68036  10.498195     17       17
## 34552  11.101191     11       14
## 145324 10.404263      2       14
## 81157  10.463103      9       15
## 156704 10.714418     22       14
## 132185 10.463103     32       14
## 168216 11.608236     22       15
## 150695  9.392662     47        9
```