

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,]
```

```

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

  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)
    }

  }

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

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

  if (sampq == FALSE){
    return(datamatrix)
  }
}

```

```
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 1.25 in 2014. Therefore, to adjust 2004 income to its 2014 equivalent, I wrote a function that multiplied all 2004 income by 1.25.

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

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

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
  return(sample)

}

logVarf(sample)

##      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"

```
sample <- logVarf(sample)
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
  return(sample)
}

potExpf(sample)

```

##		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
## [1,] 11.225243    39
## [2,] 10.596635    31
## [3,] 10.532096    20
## [4,] 10.021271    46
## [5,] 10.532096    24
## [6,] 11.042922    25
## [7,]  9.838949     2
## [8,] 10.021271     1
## [9,]  9.433484     1
## [10,] 10.140771    42
## [11,] 10.075338    52
## [12,] 10.319299    53
## [13,]  8.793005    52
## [14,] 10.170648    23
## [15,] 10.390840    18
## [16,] 10.819778    39
## [17,] 10.126631     6
## [18,]  8.517193    10
## [19,] 10.672669    30
## [20,] 10.917268    24
## [21,] 10.085809    17
## [22,]  9.546813    32
## [23,] 10.203592    17
## [24,]  9.798127     8
## [25,] 11.002100     0
## [26,] 10.571317     8
## [27,]  8.154788     2
## [28,] 10.920528    16
## [29,] 10.463103    29
## [30,]  9.239899     2
```

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

```
library(stargazer)

sample <- potExpf(sample)

fit <- lm()

## Error in terms.formula(formula, data = data): argument is not a
```

valid model