Computing Project

Daniel Lee

April 12, 2016

```
library(data.table)
library(ff)

## Warning: package 'ff' was built under R version 3.2.4

## Loading required package: bit

## Attaching package bit

## package:bit (c) 2008-2012 Jens Oehlschlaegel (GPL-2)

## creators: bit bitwhich

## coercion: as.logical as.integer as.bit as.bitwhich which

## operator: ! & | xor != ==

## querying: print length any all min max range sum summary

## bit access: length<- [ [<- [[ [[<-

## for more help type ?bit

##
## Attaching package: 'bit'

## The following object is masked from 'package:data.table':
##
##     setattr

## The following object is masked from 'package:base':
##
##     xor

## Attaching package ff

## - getOption("fftempdir")=="C:/Users/Daniel/AppData/Local/Temp/Rtmpi0dXFx"

## - getOption("ffextension")=="ff"

## - getOption("ffdrop")==TRUE

## - getOption("fffinonexit")==TRUE

## - getOption("ffpagesize")==65536

## - getOption("ffcaching")=="mmnoflush"  -- consider "ffeachflush" if your
system stalls on large writes
```

```
## - getOption("ffbatchbytes")==63501762.56 -- consider a different value for
tuning your system

## - getOption("ffmaxbytes")==3175088128 -- consider a different value for
tuning your system

##
## Attaching package: 'ff'

## The following objects are masked from 'package:bit':
##
##     clone, clone.default, clone.list

## The following objects are masked from 'package:utils':
##
##     write.csv, write.csv2

## The following objects are masked from 'package:base':
##
##     is.factor, is.ordered

library(ggplot2)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(biglm)

## Warning: package 'biglm' was built under R version 3.2.4

## Loading required package: DBI

setwd("C:/Users/Daniel/Desktop/ph290/Assignment 3")
set.seed(1000)
```

Problem 1

Here are the 3 different commands of reading the zipped data

```
system.time(read.csv("ss13hus.csv.bz2", nrow = 10000))
```

```
##    user  system elapsed
##    1.74    0.02    1.75

system.time(scan(file = "ss13hus.csv.bz2", what = list(rep("character",
1000)),
                                skip = 1, sep = ",",
                                nlines = 10000))

##    user  system elapsed
##    1.50    0.02    1.52

con <- bzfile("ss13hus.csv.bz2", "rt")
system.time(readLines(con, n = 10000))

##    user  system elapsed
##    1.05    0.00    1.04

close(con)
```

Problem 2

The code below creates a vector of indexes to select the appropriate columns.

```
data_fields <- c("REGION", "ST", "ADJHSG", "ADJINC", "NP", "ACR", "BDSP",
"ELEP", "GASP",
                 "RMSP", "VEH", "WATP", "FINCP", "HINCP")
col_names <- read.table("ss13hus.csv.bz2", nrow = 1, sep = ",")
col_names_index <- c()
for(i in 1:length(col_names)){
  for(j in 1:length(data_fields)){
    if(data_fields[j] == col_names[i]){
      col_names_index <- c(col_names_index, i)
    }
    }
}
```

Sampling Index

I take a sample of one million from a vector of numbers ranging from 1 to 7219001. 7219001 is the number of rows in the dataset `ss13hus.csv.bz2`. This number was obtained using bash. However, I couldn't get the code to work in R. I included my attempt at coding bash into R below.

```
#failed R code for Bash
#line_number <- system2(paste('"cd /Desktop/ph290/Assignment 3"', 'bzip2 -dck
ss13hus.csv.bz2 | wc -l'))
full_sample_index <- sort(sample(1:7219001, size = 1000000, replace = FALSE))
```

One Million Random Samples

The code below is the actual sampling of one million random samples from `ss13hus.csv.bz2`. The function `subset_data` examines 10,000 rows from the dataset at a time. It selects the rows that match the number in the `full_sample_index_`. The sample with one million random samples is stored in `million_samples`.

```
subset_data <- function(data_set, sample_index, col_index){
  counter <- 1
  con <- bzfile(data_set, "rt")
  sample_matrix <- matrix(data = NA, nrow = length(sample_index), ncol =
length(col_index),
                          dimnames = list(c(), c("REGION", "ST", "ADJHSG",
"ADJINC", "NP", "ACR",
                                                 "BDSP", "ELEP", "GASP",
                                                 "RMSP", "VEH", "WATP",
"FINCP", "HINCP")))
  for(i in 1:ceiling(max(sample_index)/10000)){
    block <- read.csv(con, nrow = 10000, header = ifelse (i == 1, TRUE,
FALSE))
    block_index <- sample_index[which(sample_index <= 10000*i &
                                        sample_index > (i-1)*10000)] - (i-
1)*10000
    sample_matrix[counter:(counter+length(block_index)-1),] <-
as.matrix(block[block_index,

col_index])
    counter <- counter+length(block_index)
  }
  close(con)
  return(sample_matrix)
}

million_samples <- subset_data("ss13hus.csv.bz2",
full_sample_index,col_names_index)
```

Save `million_samples` into CSV

The code below is for saving the file `million_samples` into a csv.

```
write.csv(million_samples, file = "million_samples.csv", quote = FALSE, na =
"NA")
```

Problem 3

Here are the three different commands of reading data created in step 2.

```
system.time(read.csv("million_samples.csv"))
```

```
##    user  system elapsed 
##    9.95    0.34   13.05
```

```
system.time(fread("million_samples.csv"))
```

```
##    user  system elapsed 
##    0.97    0.00    1.12
```

```
system.time(read.csv.ffdf(file = "million_samples.csv"))
```

```
##    user  system elapsed 
##    5.75    0.28    6.03
```

Problem 4

Clean up the dataset

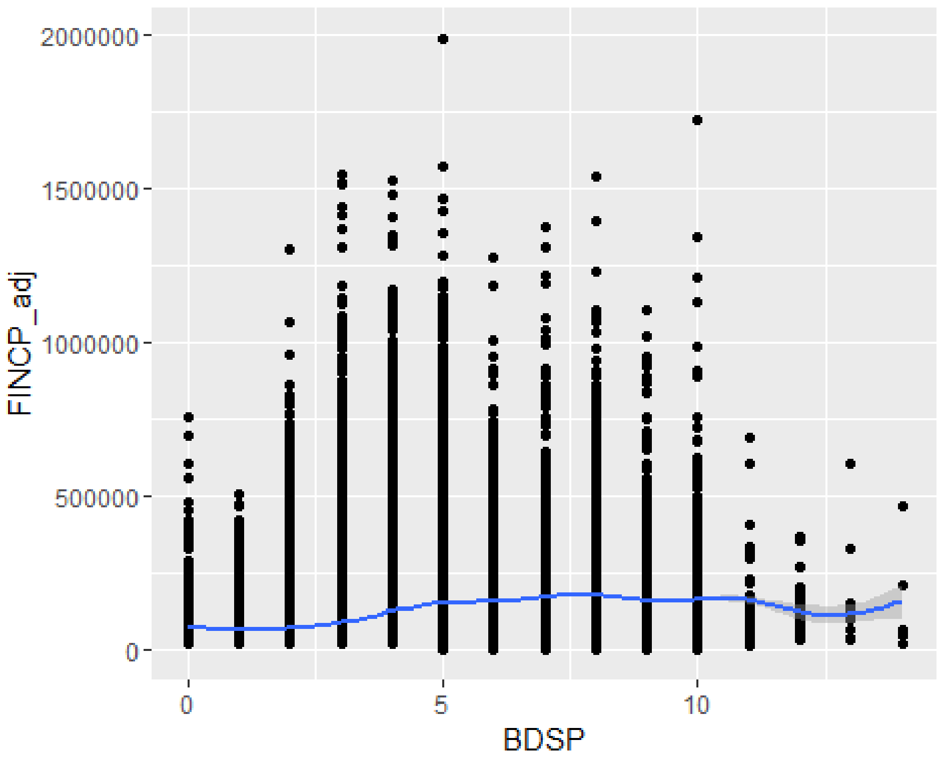
Here's the code to clean up the dataset. First, I removed all the rows with "NA". Then, I removed all the rows with negative family income. Then, I add a new column to dataset for adjusted family income called `FINCP_adj`. `FINCP_adj`is formed by multiplying `FINCP` and `ADJINC` and scaling by 0.000001. Then, if a household has greater than or equal to five bedrooms, I only selected rows with adjusted family income greater than $20,000. Finally, I removed all rows with more than 15 bedrooms. After I removed all the unwanted rows, I am left with 447,786 rows in the dataset.

```
million_samples_dt <- fread("million_samples.csv", na.strings = "NA")

row.has.na <- apply(million_samples_dt, 1, function(x){any(is.na(x))})

mil_samples_filtered <- million_samples_dt[!row.has.na,][FINCP >= 0][,
FINCP_adj :=

as.numeric(FINCP) *

as.numeric(ADJINC) *

0.000001][FINCP_adj >=

20000 |BDSP >=

5][BDSP < 15]
```

Scatter Plot

I created a scatter plot of `BDSP` and `FINCP_adj`. I also included a loess smoother with standard error shading.

```
p <- ggplot(mil_samples_filtered, aes(x = BDSP, y = FINCP_adj))
p + geom_point() + geom_smooth()
```



Problem 5

Linear Regression Model

```r
linear_model <- bigglm(FINCP_adj ~ BDSP + VEH, data = mil_samples_filtered)
summary(linear_model)
```

```
## Large data regression model: bigglm(FINCP_adj ~ BDSP + VEH, data = 
mil_samples_filtered)
## Sample size =  447786 
##                  Coef      (95%       CI)       SE p
## (Intercept)  3292.081  2319.168  4264.993 486.4562 0
## BDSP        20547.313 20291.750 20802.876 127.7814 0
## VEH         12259.334 12000.331 12518.336 129.5012 0
```