A1.R

yuantien

2022-01-21

```
#A1 Yuan Tien

library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──
```

```
## ✓ ggplot2 3.3.5     ✓ purrr   0.3.4
## ✓ tibble  3.1.6     ✓ dplyr   1.0.7
## ✓ tidyr   1.1.4     ✓ stringr 1.4.0
## ✓ readr   2.1.1     ✓ forcats 0.5.1
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
getwd()
```

```
## [1] "/Users/yuantien/Desktop/R/613/Data"
```

```
setwd("/Users/yuantien/Desktop/R/613/Data")
getwd()
```

```
## [1] "/Users/yuantien/Desktop/R/613/Data"
```

```
dathh2007 <- read.csv("dathh2007.csv")
dathh07 <- dathh2007
rm(dathh2007)

#1.1
class(dathh07$idmen)
```

```
## [1] "numeric"
```

```
a <- unique(dathh07$idmen) #find unique value
length(a) #10498
```

```
## [1] 10498
```

```
getwd()
```

```
## [1] "/Users/yuantien/Desktop/R/613/Data"
```

```
#1.2
dathh05 <- read.csv("dathh2005.csv")

length(dathh05$mstatus[dathh05$mstatus =="Couple, with Kids"]) #3374
```

```
## [1] 3374
```

```
table(dathh05$mstatus) #redo this with a more convenient way
```

```
##
##   Couple, No kids Couple, with Kids             Other            Single
##              2656              3374               275              2663
##     Single Parent
##               785
```

```
#1.3

datind08 <- read.csv("datind2008.csv")
b <- unique(datind08$idind)
length(b) #it shows 10825, but this individual level data has 25510 obs.
```

```
## [1] 10825
```

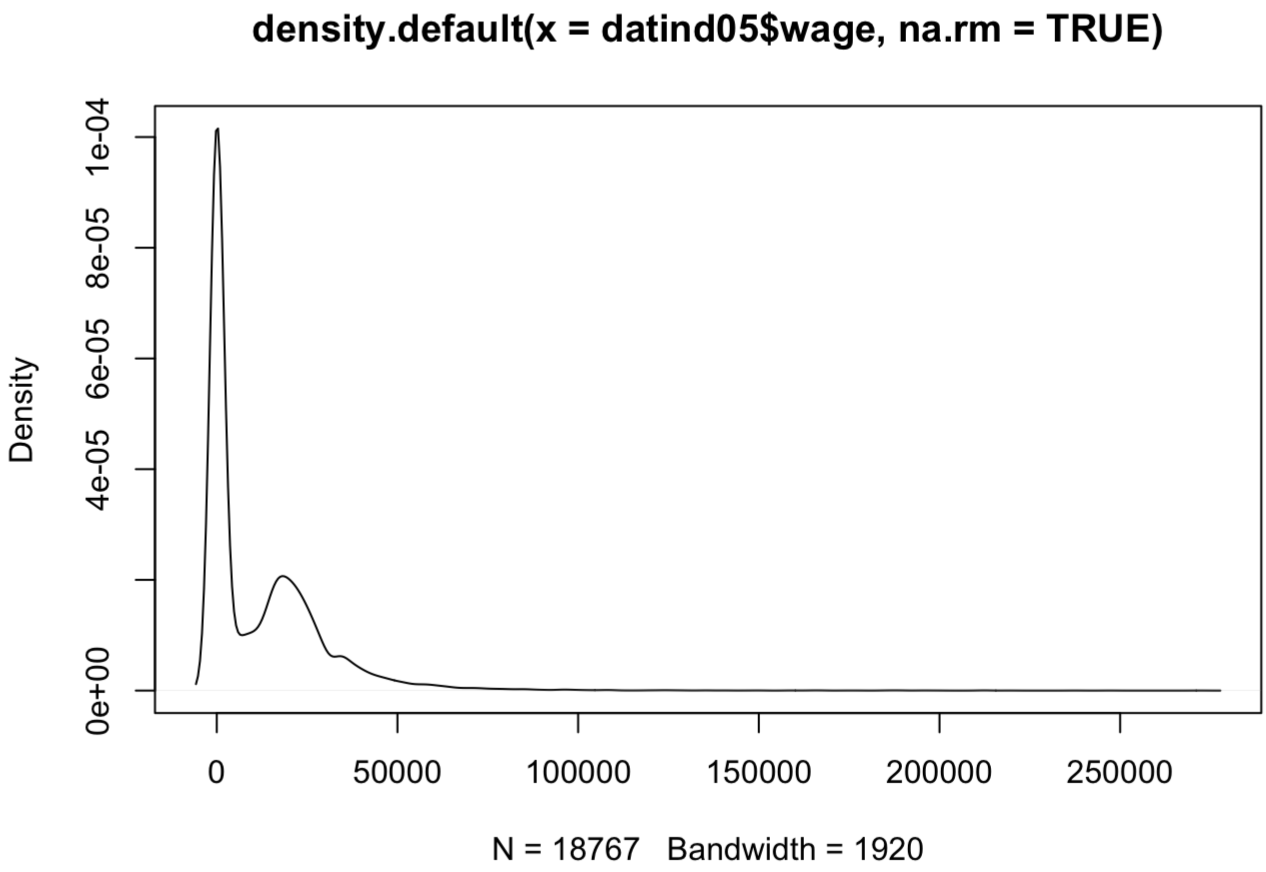
```
#1.4
datind16 <- read.csv("datind2016.csv")
a <- datind16 %>%
  filter(age>= 25 & age<=35) %>%
  nrow()
a #2765
```

```
## [1] 2765
```

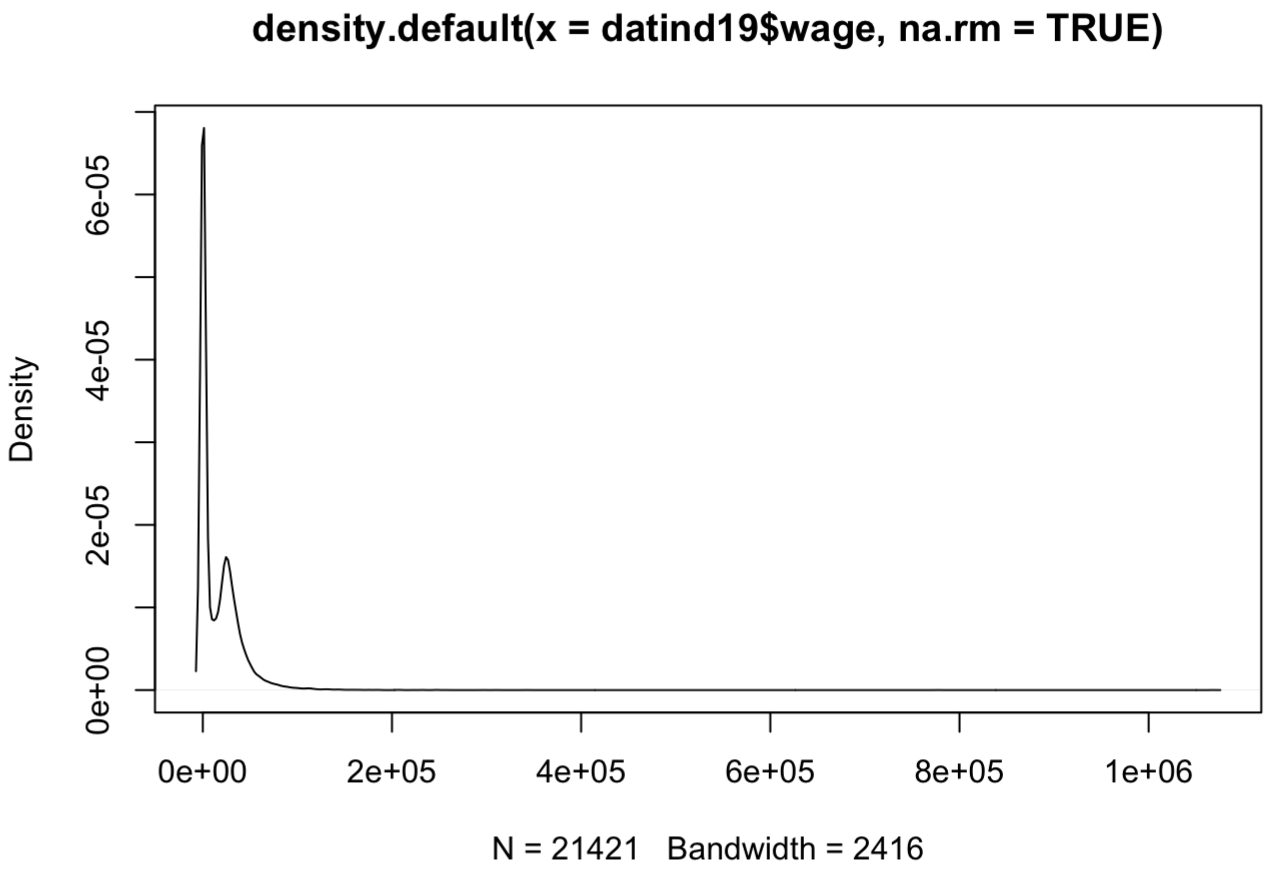
```
#1.5
datind09 <- read.csv("datind2009.csv")
CrossTable <- table(datind09$gender, datind09$profession)
CrossTable
```

```
##
##            0  11  12  13  21  22  23  31  33  34  35  37  38  42  43  44  45
##   Female  11  30   8  29  63  65   8  68  85 184  50 179  78 258 437   1 153
##   Male    19  57  19  78 213 114  48  98 107 142  59 260 368 110 117   2  95
##
##           46  47  48  52  53  54  55  56  62  63  64  65  67  68  69
##   Female 410  82  22 782  27 584 353 696  64  35  29  19 147 120  40
##   Male   340 429 215 169 182  98 101  74 443 520 246 159 237 177  82
```

```r
#1.6
datind05 <- read.csv("datind2005.csv")
datind19 <- read.csv("datind2019.csv")
plot(density(datind05$wage, na.rm = TRUE)) #plot the distribution
```




```r
plot(density(datind19$wage, na.rm = TRUE))
```



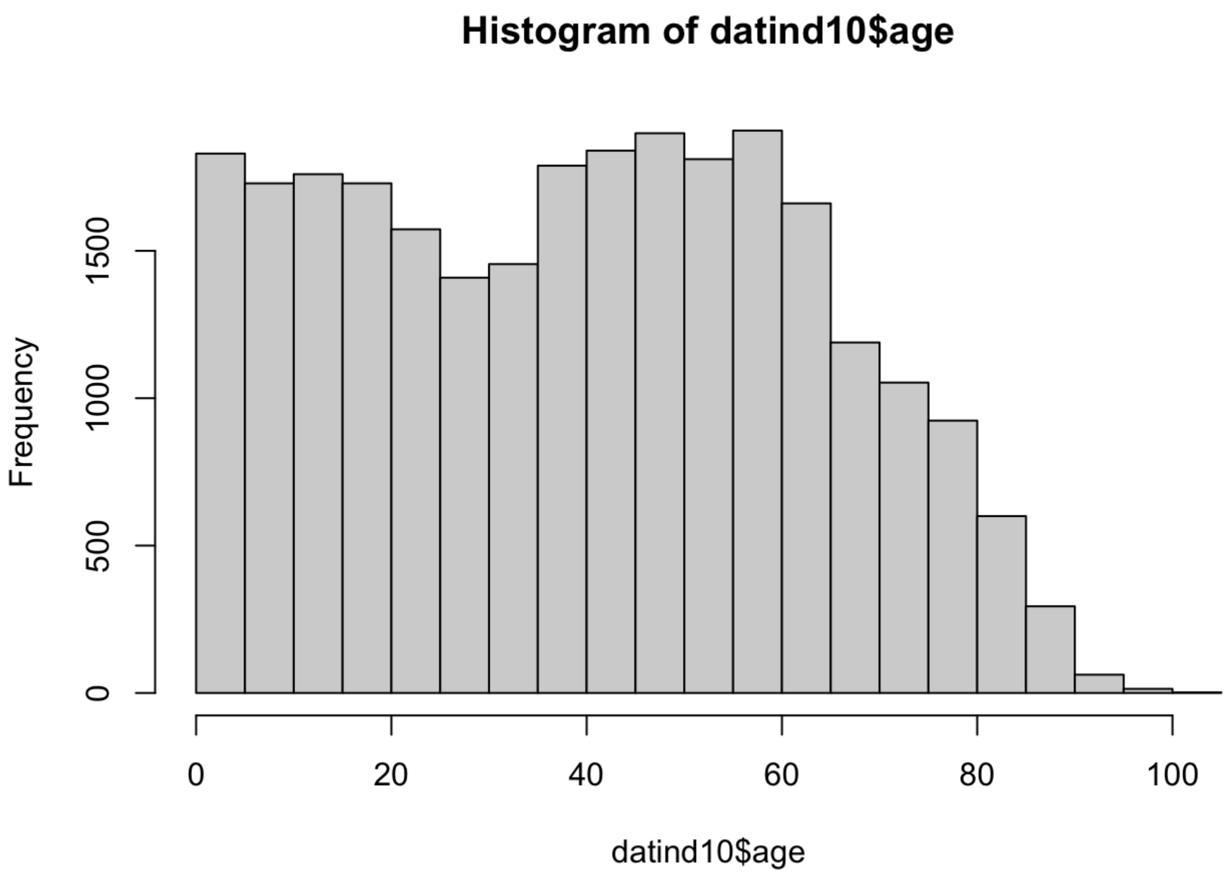

```
inter_decile <- function(x) {
  quantileX = quantile(x, prob = c(0.1, 0.9))
  ratio = quantileX[2]/quantileX[1] #because 2nd element represent the 90% and the 1s
t element represent the 10%
  return(ratio)
}


gini <- function(y) {
  n = length(y)
  a = 1/(n-1)
  b = (n+1)
  c = - 2*((sum((n+1-1:n)*y)))
  d = sum(y)
  return(a*(b-c/d))
} #this is sample gini coefficient. Reference: http://www3.nccu.edu.tw/~jthuang/Gini.
pdf page 2

dist_report <- function(x) {
  return(c(mean = mean(x), sd = sd(x), ratio = inter_decile(x), gini = gini(x)))
}
datind05_rm <- na.omit(datind05$wage) #clear out rows with NA in wage
datind19_rm <- na.omit(datind19$wage)
datind05_rm <- datind05_rm[datind05_rm != 0] #clear out wage = 0
datind19_rm <- datind19_rm[datind19_rm != 0]
dist_report(datind05_rm) #mean = 22443.029, sd = 18076.708, ratio = 8.896, gini = 2.0
01
```

```
##         mean           sd    ratio.90%         gini
## 22443.029118 18076.708882     8.896525     2.001934
```

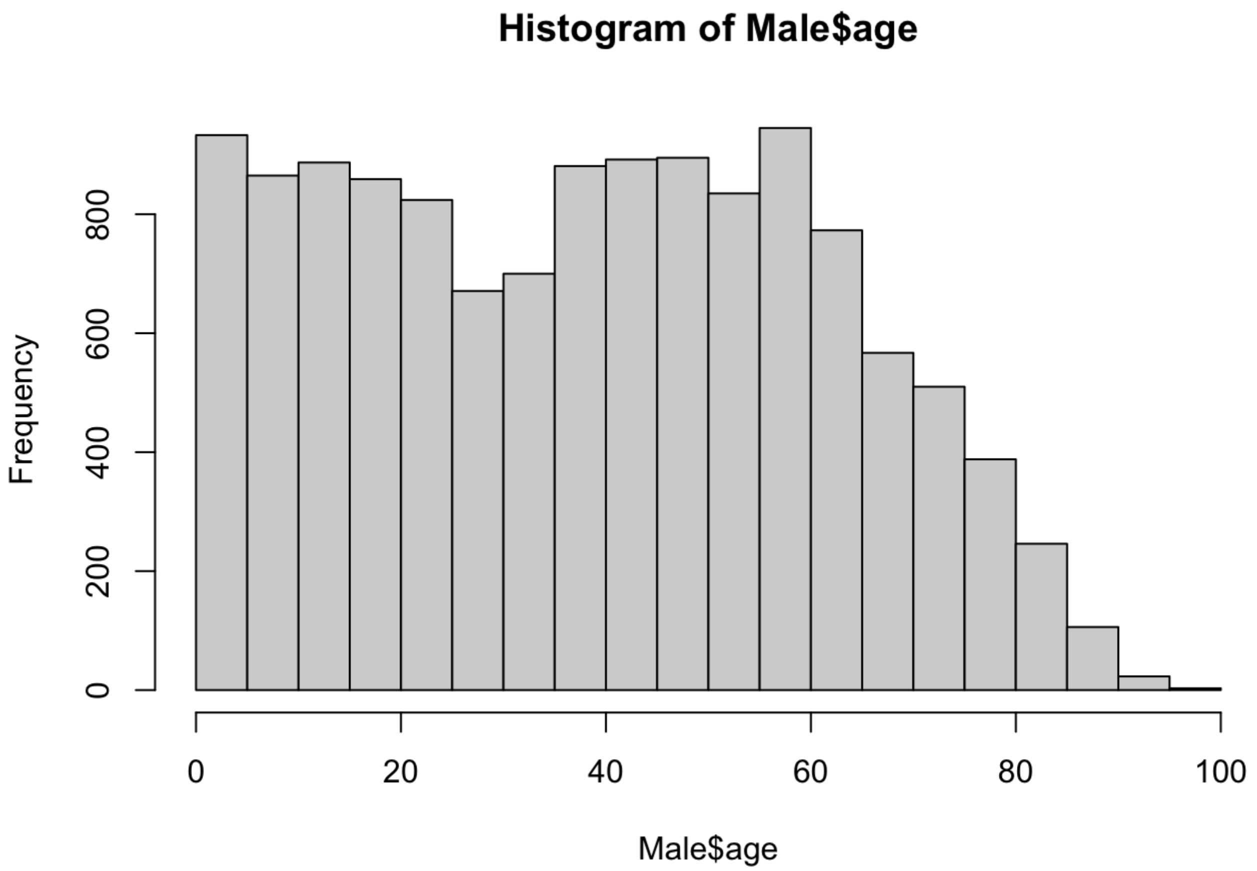
```
dist_report(datind19_rm) #mean = 27578.839, sd = 25107.187, ratio = 13.862, gini = 2.
041
```

```
##         mean           sd    ratio.90%         gini
## 27578.839302 25107.187196    13.862300     2.041109
```

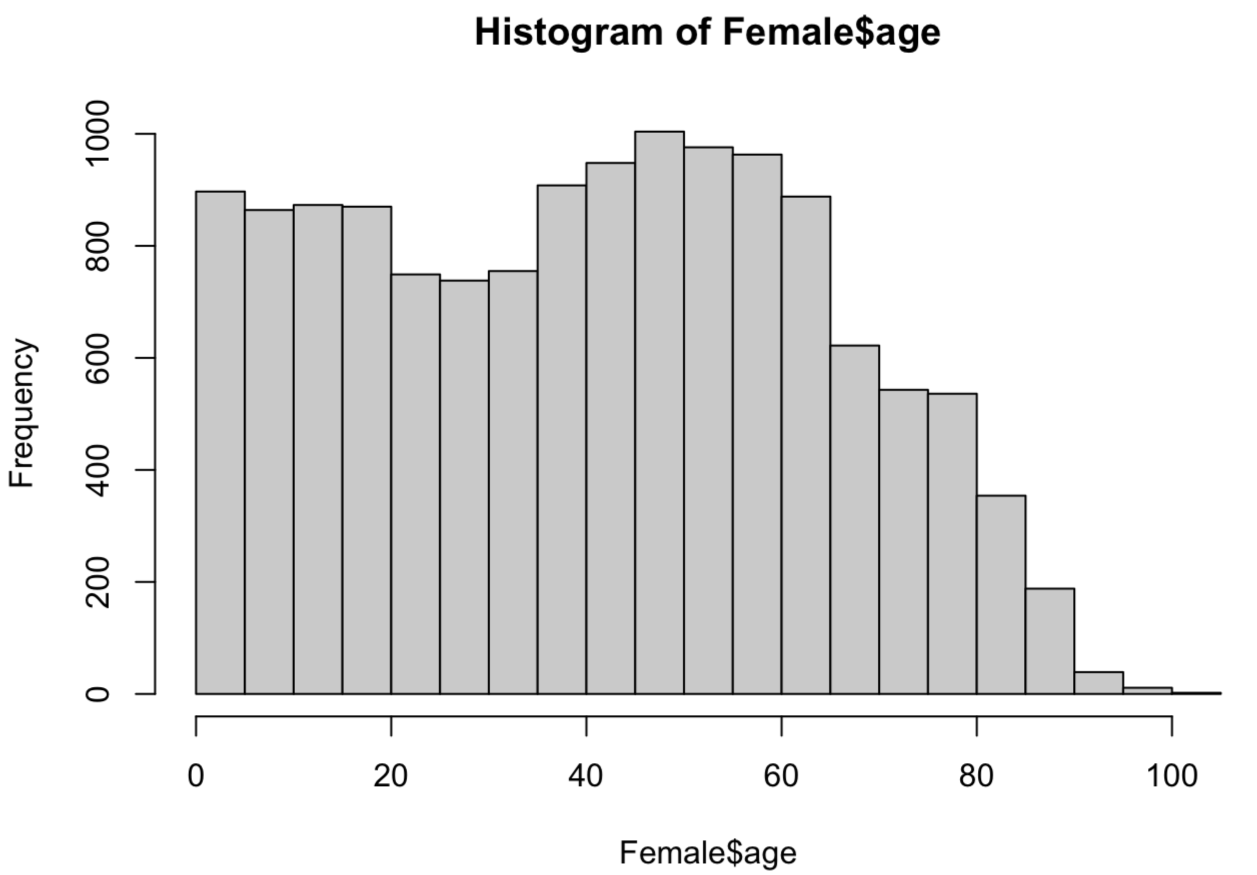
```
#1.7
datind10 <- read.csv("datind2010.csv")
hist(datind10$age)
```




```
Male <- datind10[datind10$gender == "Male",] #remember to put in Comma to select the
 rows we like
Female <- datind10[datind10$gender == "Female",]
hist(Male$age)
```




```
hist(Female$age)
```



```
# the most represented male group in the samples is around 60 years old, while the mo
st represented female group is around 50
#


#1.8
datind11 <- read.csv("datind2011.csv")
dathh11  <- read.csv("dathh2011.csv")
m11 <- datind11 %>%
  inner_join(dathh11, by = "idmen")
#merge household dataset which contains location to ind data
#I use inner join to delete individuals who do not appear in the household dataset
nrow(m11[m11$location == "Paris",]) #3531
```

```
## [1] 3531
```

```
#Exercise 2
#2.1
dind = list.files(pattern="datind")
for (i in 1:length(dind)) assign(dind[i], read.csv(dind[i])) #find data in my file an
d read multiple files

Mind <- do.call("rbind", list(datind2004.csv, datind2005.csv, datind2006.csv, datind2
007.csv, datind2008.csv, datind2009.csv,
                      datind2010.csv, datind2011.csv, datind2012.csv, datind2013.csv,
datind2014.csv, datind2015.csv,
                      datind2016.csv, datind2017.csv, datind2018.csv, datind2019.cs
v))

#2.2
dhh = list.files(pattern="dathh")

for (i in 1:length(dhh)) assign(dhh[i], read.csv(dhh[i]))

Mhh <- do.call("rbind", list(dathh2004.csv, dathh2005.csv, dathh2006.csv, dathh2007.c
sv, dathh2008.csv, dathh2009.csv,
                             dathh2010.csv, dathh2011.csv, dathh2012.csv, dathh2013.c
sv, dathh2014.csv, dathh2015.csv,
                             dathh2016.csv, dathh2017.csv, dathh2018.csv, dathh2019.c
sv))
#2.3
colnames(Mind)
```

```
##  [1] "X"          "idind"      "idmen"      "year"       "empstat"
##  [6] "respondent" "profession" "gender"     "age"        "wage"
```

```
colnames(Mhh)
```

```
## [1] "X"        "idmen"    "year"     "datent"   "myear"    "mstatus"  "move"
## [8] "location"
```

```r
y = c(colnames(Mind), colnames(Mhh))
y[duplicated(y) == TRUE] #X, idmen, year
```

```
## [1] "X"     "idmen" "year"
```

```r
#find duplicated column names --> find variables that appear in both datasets)

#2.4
M <- inner_join(Mhh, Mind, by = c("idmen", "year"))
#I use innter_join because I believe those household ids that appear in both datasets
more reliable data

#2.5

M1 <- M #create M1 in case of unexpected accident
members_more_4 = function(x) {
  M2 = M1 %>%
    filter(year == x)
  z = table(M2$idmen)
  y = as.data.frame(z)
  nrow( y[ y$Freq >= 4, ] )
}
# I do this by year
# I create a frequency table by household -> turn to a dataframe -> calculate frequen
cy
year = 2004:2019
more_4_by_year = sapply(year, members_more_4)
sum(more_4_by_year) #37108
```

```
## [1] 37108
```

```r
#2.6
more_1_unemp = function(x) {
  M2 = M1 %>%
    filter(year == x)
  z = table(M2$idmen, M2$empstat)
  y = as.data.frame(z)
  h = y %>%
    filter(Var2 == "Unemployed")
  nrow( h[ h$Freq >=1, ] )
}
more_unemp_year = sapply(year, more_1_unemp)
sum(more_unemp_year) #17241
```

```
## [1] 17241
```

```r
#2.7
unique(M1$profession) #check professions, I am not sure if "X1" "X2" "HO" are profess
ions. Below, I assume they are
```

```
##  [1] "67" "56" ""   "38" "45" "34" "42" "46" "37" "54" "11" "63" "55" "48" "52"
## [16] "68" "23" "53" "31" "21" "22" "62" "43" "47" "33" "69" "65" "64" "12" "35"
## [31] "13" "44" "00" "X1" "X2" "HO" NA   "0"  "50" "36" "66" "61"
```

```
twoprof = function(x) {
  M2 = M1 %>%
    filter(year == x)
  z = table(M2$idmen, M2$profession)
  y = as.data.frame.matrix(z)
  nrow(y[y[,2:ncol(y)] >= 2,])
}
#By this, I return the rows that from column2 to column_n_professions where the frequ
ency >= 2
#why from column 2? because column 1 appears to be the freq of NA, the first professi
on 00 starts with 2nd column

two_prof_year <- sapply(year, twoprof)
sum(two_prof_year) #7509
```

```
## [1] 7509
```

```
#2.8
M1 %>%
  filter(mstatus == "Couple, with Kids") %>%
  nrow() #209382
```

```
## [1] 209382
```

```
#2.9
M1  %>%
  filter(location == "Paris") %>%
  nrow() #51904
```

```
## [1] 51904
```

```
#2.10
most_mem <- function(x) {
  M2 = M1 %>%
    filter(year == x)
  z = table(M2$idmen)
  y = as.data.frame(z)
  y[which.max(y$Freq),] #which.max will find me the index of maximum value
}
most_mem_year <- sapply(year, most_mem)
most_mem_year #the most in 2007 row 9903, and 2010 row 10991. Both have 14 members
```

```
##      [,1]             [,2]             [,3]             [,4]
## Var1 1208045118450100 1607839058220100 1607839058220100 2207811124040100
## Freq 10               11               10               14
##      [,5]             [,6]             [,7]             [,8]
## Var1 1700707001000100 1700707001000100 2510263102990100 1905191114960100
## Freq 10               11               14               10
##      [,9]             [,10]            [,11]            [,12]
## Var1 1905191114960100 2202243098040100 2106457101960100 3000896115750100
## Freq 10               10               9                12
##      [,13]            [,14]            [,15]            [,16]
## Var1 3000896115750100 3000896115750100 3000896115750100 2806477001000100
## Freq 12               12               11               9
```

```
most_mem(2007) #idem: 2207811124040100
```

```
##                  Var1 Freq
## 9903 2207811124040100   14
```

```
most_mem(2010) #idem: 2510263102990100
```

```
##                   Var1 Freq
## 10991 2510263102990100   14
```

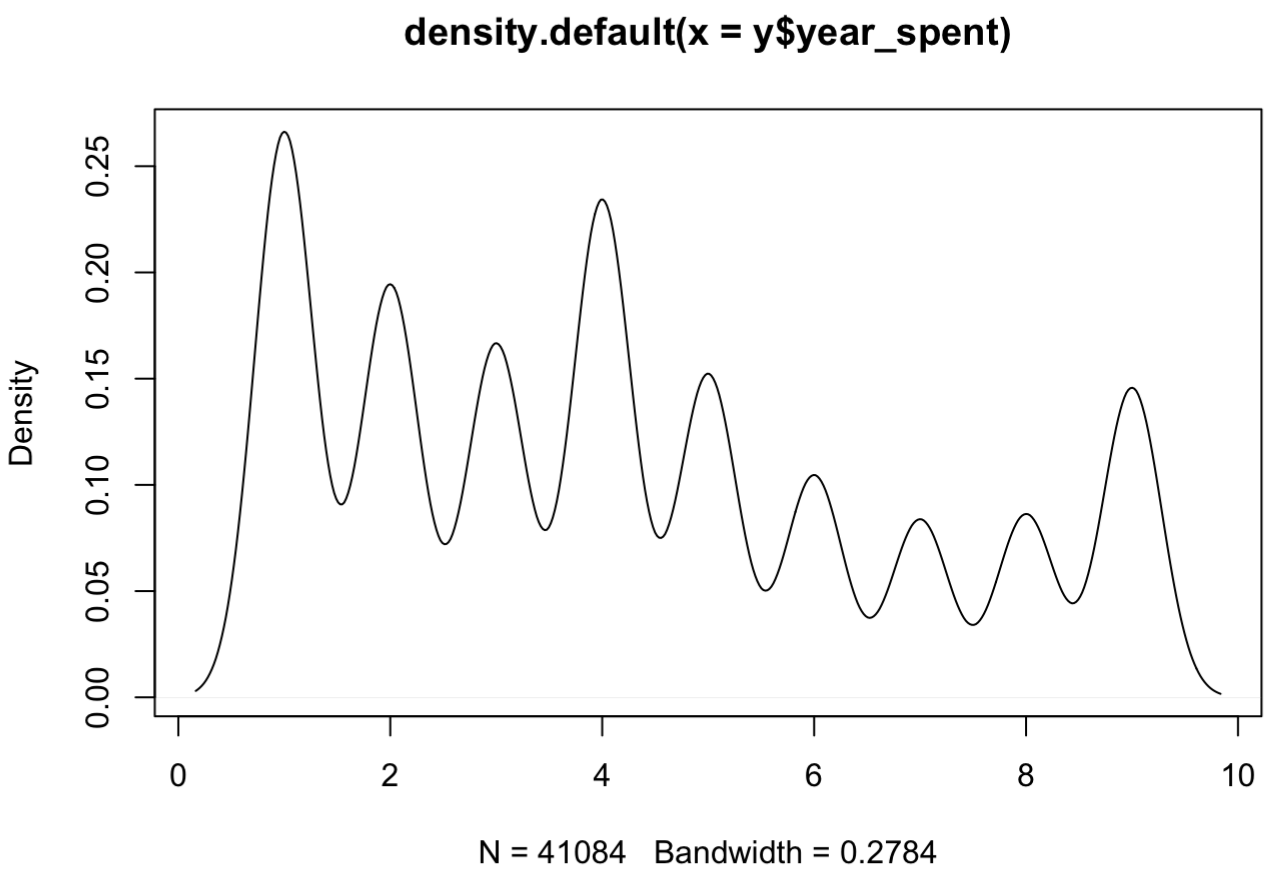
```
#2.11
M2 <- M1 %>%
  filter(year == 2010)
length(unique(M2$idmen)) #11048 households in 2010
```

```
## [1] 11048
```

```
M2 <- M1 %>%
  filter(year == 2011)
length(unique(M2$idmen)) #11360 households in 2011
```

```
## [1] 11360
```

```
#Exercise 3

#3.1
Mhh2 <- Mhh #I only work on the household dataset

z = table(Mhh2$idmen, Mhh2$year)
y = as.data.frame.matrix(z)
# the df is now sorted by idmen and showcase the year of entry (the earliest column w
ith 1) and the year of exit (the last column with 1)


y$year_spent <- rowSums(y)
#the columns are years, and the value is either 1 (participate) or 0 (not participat
e).
#So, by summing all the columns I get the years spent in survey
plot(density(y$year_spent)) #plot the distribution of time spent in the survey
```



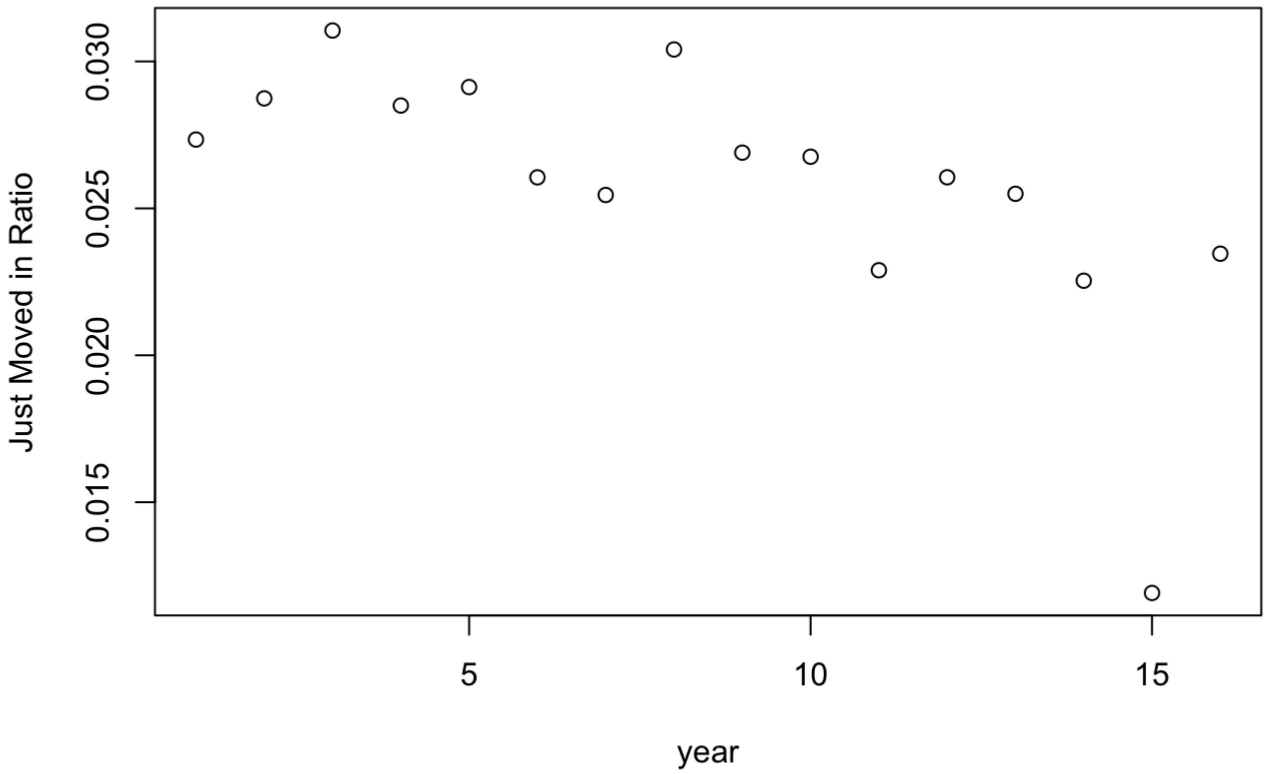

```
#3.2
Mhh2$move_in = Mhh2$year - Mhh2$datent == 0 # create variable "move_in" return 0 -> r
espondents moved in the same year as the survey

head(Mhh2$move_in, 10) # first 10 rows, all false
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
#the second part of question requires us to plot the share of "individuals"

h = as.data.frame.matrix( table(M1$idmen, M1$year) ) #create households members by ye
ar and household

dwelling <- function(x) {
  u <- h %>%
    select(members = as.character(x)) #I put as.character to make it work. Without it
R cannot execute my function.
  u$idmen <- rownames(u)
  u$idmen <- as.numeric(u$idmen)

  p <- Mhh2 %>%
    filter(year == x)

  K <- inner_join(p, u, by = "idmen") #here I create members count in 2004, and put i
t in hh dataset
  #duplicate household data by members
  n.times <- K$members
  N <- K[ rep( seq_len( nrow(K) ), n.times), ]
  z <- table(N$year, N$move_in)
  y <- as.data.frame.matrix(z)

  y$ratio <- y$"TRUE"/(y$"TRUE"+y$"FALSE")
  return(y)
}

dwelling(2004) #check if this work, and it works
```

```
##      FALSE TRUE      ratio
## 2004 21522  605 0.02734216
```

```
dwelling_year <- sapply(year, dwelling)
dwelling_year <- t(dwelling_year)
dwelling_year <- as.data.frame(dwelling_year)
dwelling_year$ratio <- as.numeric(dwelling_year$ratio)
plot(dwelling_year$ratio, xlab = "year", ylab = "Just Moved in Ratio")
```



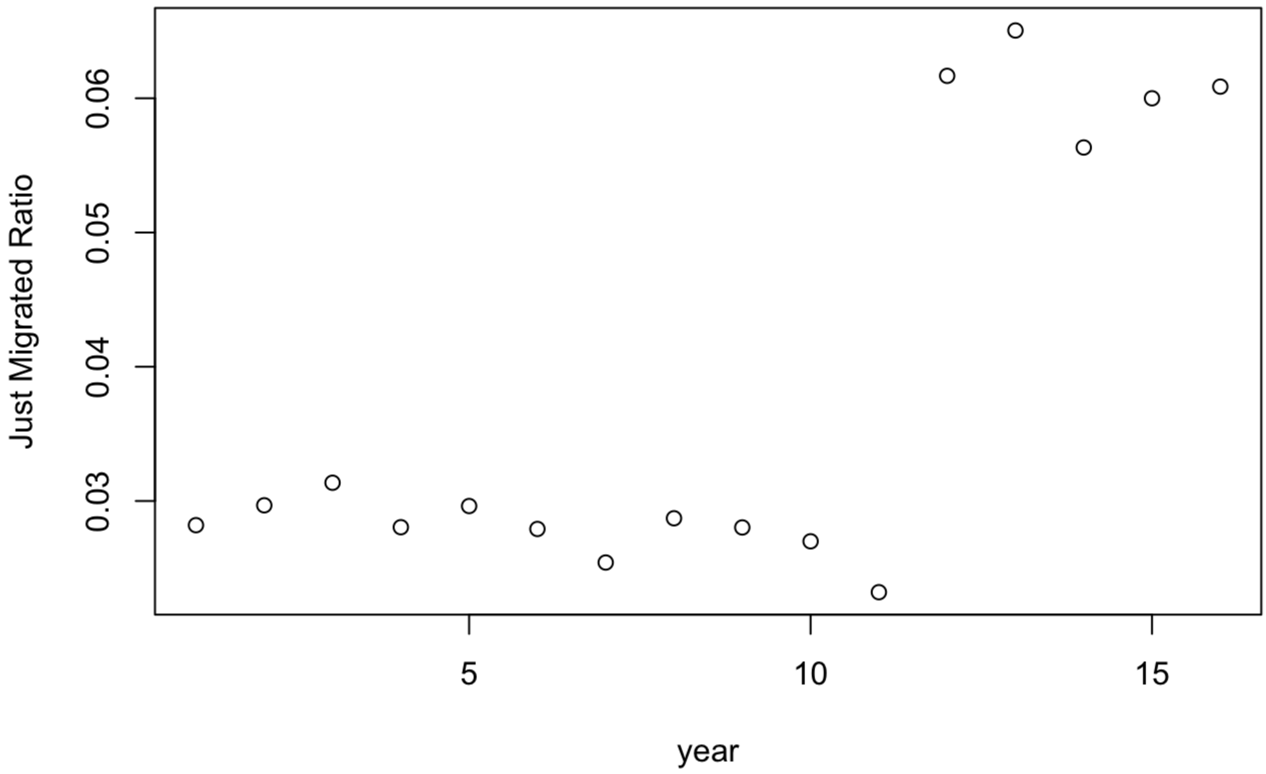

```
#Below is the code for plotting the "share of households" instead of individuals
#z <- table(Mhh2$year, Mhh2$move_in)
#y <- as.data.frame.matrix(z)
#y

#y$ratio <- y$"TRUE"/y$"FALSE"

#plot(y$ratio, xlab = "year", ylab = "just moved in ratio")
#axis(1, 4:19)

#3.3
#household migrated at the year of survey

Mhhold <- Mhh2 %>%
  filter(year <= 2014)
Mhhold$mig_survey = Mhhold$year - Mhhold$myear == 0

Mhhnew <- Mhh2 %>%
  filter(year >= 2015)

Mhhnew$mig_survey = Mhhnew$move == 2
#By this method I exert that when the household reports migration in year x, I assume
the whole family migrate that year

Mhh3 <- rbind(Mhhold, Mhhnew)
head(Mhh3$mig_survey, 10) #all false for the first 10 rows
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

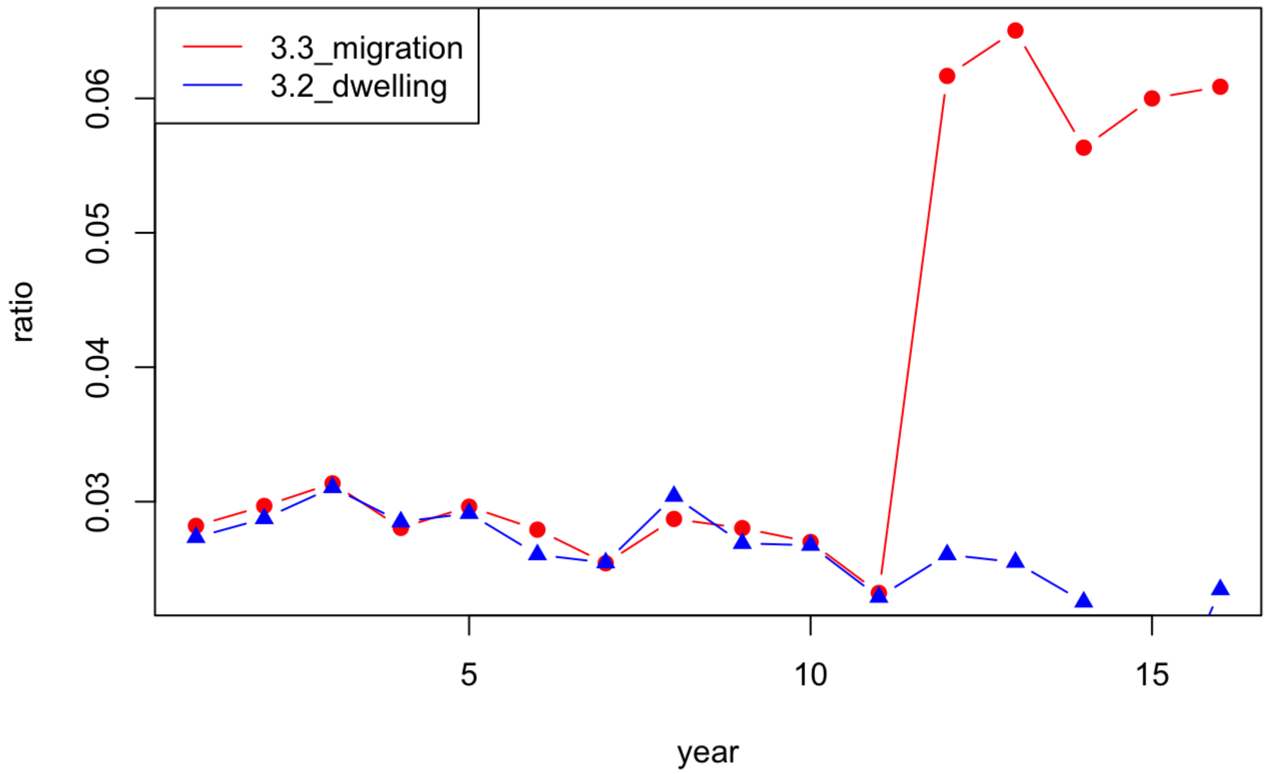
```
migrate <- function (x){
  u <- h %>%
    select(members = as.character(x))
  u$idmen <- rownames(u)
  u$idmen <- as.numeric(u$idmen)
  p <- Mhh3 %>% #Mhh3 contains migration data I created
    filter(year == x)

  K <- inner_join(p, u, by = "idmen")

  n.times <- K$members
  N <- K[rep(seq_len(nrow(K)), n.times),] #by this time I have individuals' household
data
  z <- table(N$year, N$mig_survey)
  y <- as.data.frame.matrix(z)

  y$ratio <- y$"TRUE"/(y$"FALSE"+y$"TRUE")
  return(y)
}
migrate(2004) #check if my function works
```

```
##      FALSE TRUE      ratio
## 2004 21091  612 0.02819887
```

```
migrate_year <- sapply(year, migrate)
migrate_year <-  as.data.frame(t(migrate_year))
migrate_year$ratio <- as.numeric(migrate_year$ratio)
plot(migrate_year$ratio, xlab = "year", ylab = "Just Migrated Ratio")
```




```
#3.4
plot(migrate_year$ratio, type = "b", pch = 19, col = "red", xlab = "year", ylab = "ra
tio")
lines(dwelling_year$ratio, type = "b", pch = 17, col = "blue") #lines() is used to ad
d a line on a plot
legend("topleft", c("3.3_migration", "3.2_dwelling"), lty = c(1,1), col = c("red", "b
lue"))
```



3.3_migration
3.2_dwelling
ratio
0.03
0.04
0.05
0.06
year
5
10
15

```
# I prefer method in 3.2 because there is a unnatural spike for the method in 3.3 dur
ing 2015.
# One reason for this is the change of data collection method.
# The newly introduced "move" data captures who have moved since the last survey
# while "datent" captures whether they move the same year as they filling the survey
# Suppose John responded in 2013 and 2018, and he moved in 2017. In 2018, we would sa
y he did not move for the 3.2 method
# However, we would code he have moved for the variable "move". If re-entries are ple
nty, the spike for method in 3.3 can be caused by this situation.
# Consequently, if we want to know whether the respondent has just migrated, method i
n 3.2 would be better.

#3.5
x <- Mhh3%>%
  select(idmen, move_in)
Mig <- M1 %>%
  left_join(x, by = "idmen") %>%
  filter(move_in == TRUE)



Mig2 <- Mig[ order(Mig$idmen, Mig$idind, Mig$empstat), ] #sort by idmen, idind, and e
mpstat
Mig2 <- Mig2 %>%
  select(idmen, idind, year, empstat, profession)

try <- function (x) {
  Mig2 %>%
    filter(idind == x) %>%
    mutate(change = length(unique(empstat)) >= 2) # because 2 represents change in co
ndition (1 is no change)
}

tryls <- sapply(unique(Mig2$idind), try)
tryMatrix <- as.data.frame.matrix(t(tryls))
tryMatrix$change <- as.vector(tryMatrix$change)

for (i in 1: nrow(tryMatrix)) {
tryMatrix$change2[i] = sample(unlist(tryMatrix$change[i]), 1)
}
#I use sample because "change" is either all true or all false
#right now, I have compiled "whether an individual changes their profession" in varia
ble change 2

full <- function (x) {
  Mig2 %>%
    filter(idind == x) %>%
    mutate(change = length(unique(empstat)) >= 2 | length(unique(profession)) >= 2)
}

fullls <- sapply(unique(Mig2$idind), full)
fullMx <- as.data.frame.matrix(t(fullls))
fullMx$change <- as.vector(fullMx$change)

for (i in 1: nrow(fullMx)) {
  fullMx$change2[i] = sample(unlist(fullMx$change[i]), 1) #I use sample because "chan
```

```
ge" is either all true or all false
}
# I have complied "whether an individual changes profession or changes employment sta
tus in change2
# Note, in this way if one couple where the husband fills in their household record u
nder one id,
# and the couple's professions are different, I may have coded it as change in profes
sion.
# Overall, I may have overestimated the number.

for (i in 1: nrow(fullMx)) {
  fullMx$idmen[i] = sample(unlist(fullMx$idmen[i]), 1) #turn the list idmen to a vect
or
}

answer <- fullMx %>%
  filter(change2 == TRUE)
length(unique(answer$idmen)) #find the unique households that experience this situati
on
```

```
## [1] 3090
```

```
#Exercise 4

entry_exit <- function (x) {
  a <- M1 %>%
    filter(idind == x) %>%
    arrange(year)%>%
    mutate(entry = year[1])
  a$exit  = year[nrow(a)]
  a <- a%>%
    select(idind, entry, exit)
  return(a)
} #this function creates a data.frame for every individual.
#Since the data.frame is already sorted by year, the year of the first row must be en
try, and the last must be exit
#By this I rule out re-entry and multiple exits.

entry_exit(M1$idind[678]) #just testing my function
```

```
##          idind entry exit
## 1 1.120274e+18  2004 2005
## 2 1.120274e+18  2004 2005
```

```r
a <- lapply(unique(M1$idind), entry_exit)
b <- bind_rows(a)
y <- b[!duplicated(b),] #clean out duplicated individual observation
z <- table(y$exit) #this lists exits across years

d <- as.data.frame(z)

active <- function(x) {
  length( which ( M1$year == x))
}
d$active <- sapply(year, active)

d$attrition <- d$Freq/d$active
#Report your final result as a table in proportions.
d
```

```
##    Var1 Freq active  attrition
## 1  2004 3631  22144 0.16397218
## 2  2005 4758  24241 0.19627903
## 3  2006 2888  24940 0.11579791
## 4  2007 4530  25907 0.17485622
## 5  2008 1960  25510 0.07683261
## 6  2009 3136  25611 0.12244739
## 7  2010 1138  26528 0.04289807
## 8  2011 3528  27071 0.13032396
## 9  2012 1826  28534 0.06399383
## 10 2013 1822  26353 0.06913824
## 11 2014  380  26787 0.01418599
## 12 2015 2176  26644 0.08166942
## 13 2016  310  26647 0.01163358
## 14 2017  908  25402 0.03574522
## 15 2018  882  24698 0.03571139
## 16 2019 1594  26484 0.06018728
```