A3 Hung-Wei Chang

```r
library('tidyverse')
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.
3.0 --
```

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.6     v dplyr   1.0.4
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflict
s() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library('dplyr')
library('stringr')
library('tidyr')
library("readr")
library('ggplot2')
library('lubridate')
```

```
## Warning: package 'lubridate' was built under R version 4.0.5
```

```
## 
## Attaching package: 'lubridate'
```

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

```r
library('AER')
```

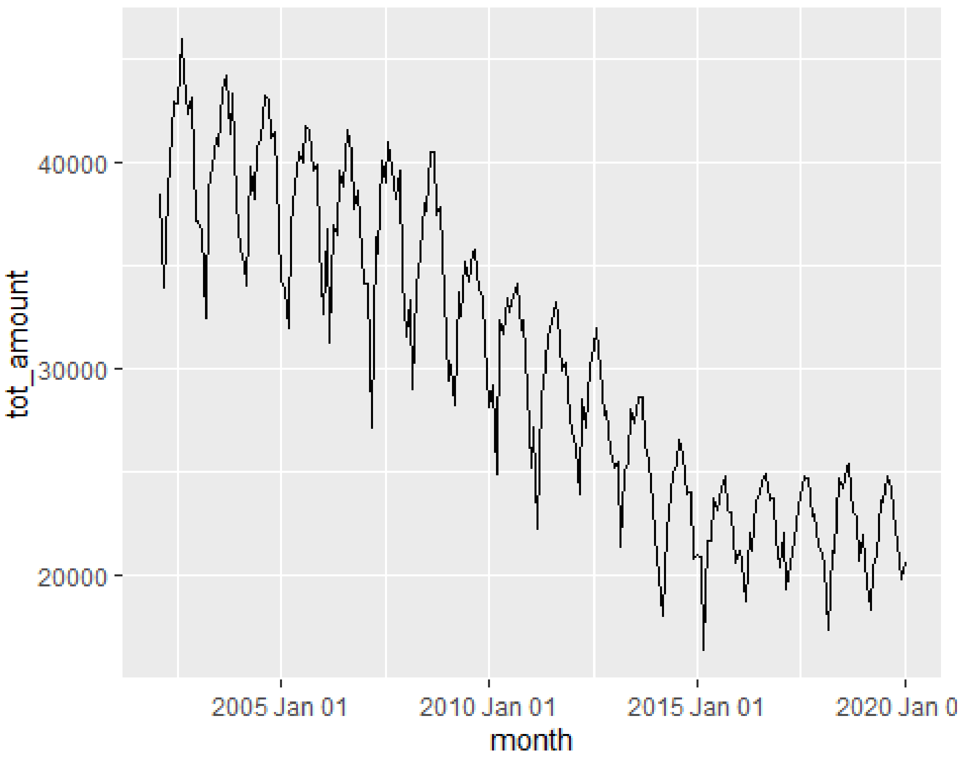
```
## Loading required package: car
```

```
## Loading required package: carData
```

```
## 
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
## 
##     recode
```

```
## The following object is masked from 'package:purrr':
## 
##     some
## Loading required package: lmtest

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich

## Loading required package: survival

library('moments')

crimelong <- read.csv('crime_long.csv')
officers <- read.csv('officers.csv')
population <- read.csv('population.csv')

#-----
# q1
#-----
# change the time index to lubridate object, so it will be easier to manipula
te and plot the time series
crimelong$crime_month <- ymd(crimelong$crime_month)
population$month <- ymd(population$month)
officers$month <- ymd(officers$month)

# group all the month together by the group_by function and create a new mont
h variable, then
# use summarize function to sum all the crimes in a given month
dat_totalcrime <- crimelong %>%
  group_by(month=ceiling_date(crime_month, "month")) %>%
  summarize(tot_amount=sum(crimes))

# graph the total crime in a month by ggplot, adding the x-labels
dat_totalcrime %>% ggplot(aes(x=month, y=tot_amount ))  + geom_line()  +  sca
le_x_date(date_labels = "%Y %b %d")
```



```
#-----
#q2
#-----
# merge the two data sets by two keys (crime_month, district) because we need
 units in the following exercise
dat_merge <- left_join(crimelong, population, by = c("crime_month" = "month",
 "district" = "district"))
print(dat_merge[1:10, 1:7])

##    crime_month district crime_type crimes period tot_pop tot_white
## 1   2002-01-01        1       drug    104     NA      NA        NA
## 2   2002-01-01        1      other     97     NA      NA        NA
## 3   2002-01-01        1      other    174     NA      NA        NA
## 4   2002-01-01        1   property    658     NA      NA        NA
## 5   2002-01-01        1   property    201     NA      NA        NA
## 6   2002-01-01        1    violent    182     NA      NA        NA
## 7   2002-01-01        1    violent     60     NA      NA        NA
## 8   2002-01-01        2       drug    161     NA      NA        NA
## 9   2002-01-01        2      other    112     NA      NA        NA
## 10  2002-01-01        2      other    158     NA      NA        NA

#-----
#q3
#-----

# change the categorical variable to 'factor' type
dat_merge$crime_type = as.factor(dat_merge$crime_type)
```

```
dat_merge$district = as.factor(dat_merge$district)

# find the total crimes per resident
# also, I find the median income, share of black, hispanic, and white residen
ts here because I only have
#   to group by the month, district
panel_total <- dat_merge %>%
  group_by(month=ceiling_date(crime_month, "month"), district) %>%
  summarize(crime_total_per_resident = sum(crimes)/ sum(tot_pop),
            median_income = median(p50_inc),
            share_black = sum(tot_black) / sum(tot_pop),
            share_hispanic = sum(tot_hisp) / sum(tot_pop),
            share_white = sum(tot_white) / sum(tot_pop)
  )
```

```
## `summarise()` has grouped output by 'month'. You can override using the `.
groups` argument.
```

```
print(panel_total[1:10, 1:5])
```

```
## # A tibble: 10 x 5
## # Groups:   month [1]
##    month      district crime_total_per_resident median_income share_black
##    <date>     <fct>                       <dbl>         <dbl>       <dbl>
##  1 2002-02-01 1                              NA            NA          NA
##  2 2002-02-01 2                              NA            NA          NA
##  3 2002-02-01 3                              NA            NA          NA
##  4 2002-02-01 4                              NA            NA          NA
##  5 2002-02-01 5                              NA            NA          NA
##  6 2002-02-01 6                              NA            NA          NA
##  7 2002-02-01 7                              NA            NA          NA
##  8 2002-02-01 8                              NA            NA          NA
##  9 2002-02-01 9                              NA            NA          NA
## 10 2002-02-01 10                             NA            NA          NA
```

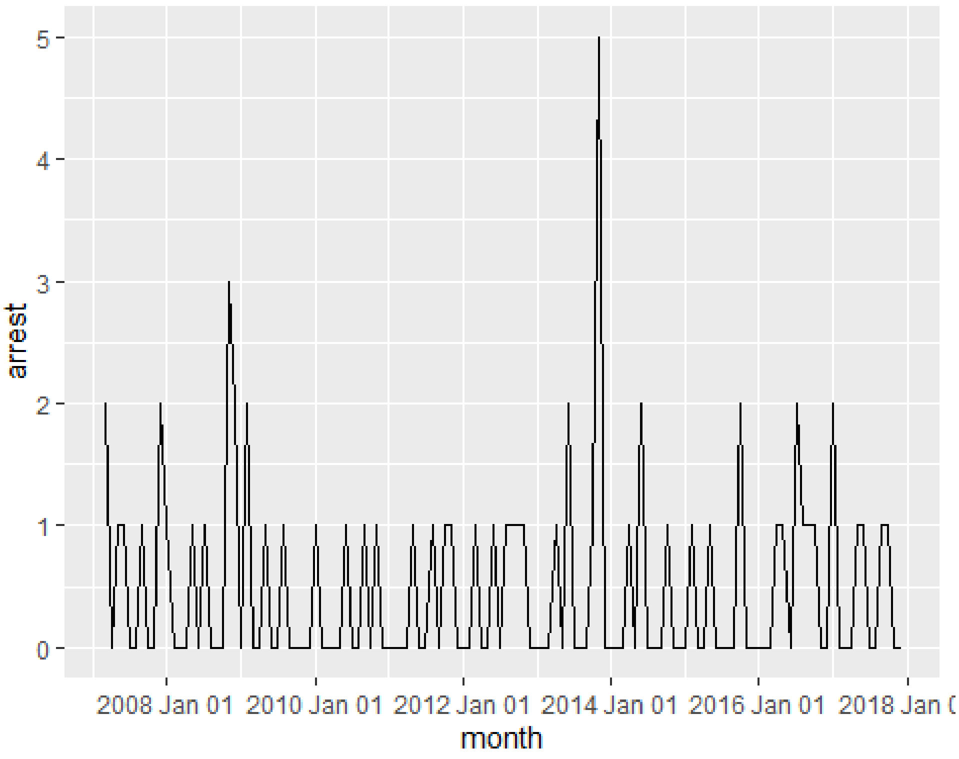
```
# find the violent crimes and property crimes per resident
# I group by both (month, district, crime_type) to find the crime rate for ea
ch crime category
#   during each month
g_dat_merge_crime_type <- dat_merge %>%
  group_by(month=ceiling_date(crime_month, "month"), district, crime_type) %>
%
  summarize(crime_type_per_resident = sum(crimes)/ sum(tot_pop)  )
```

```
## `summarise()` has grouped output by 'month', 'district'. You can override
using the `.groups` argument.
```

```
# after grouping out the data, select the desired crime type, violent and pro
perty
panel_violent <- filter(g_dat_merge_crime_type, crime_type == 'violent')[c(1,
 2, 4)]
```

```
panel_property <- filter(g_dat_merge_crime_type, crime_type == 'property')[c
(1,2, 4)]

colnames(panel_violent) = c('month', 'district', 'crime_vio_per_resident')
colnames(panel_property) = c('month', 'district','crime_pro_per_resident')


# merge all the data (panel_total, panel_violent, panel_property) all togethe
r to get my final
#   panel data (each of the three has 5128 obs.)
panel_data <- left_join(panel_total, panel_violent, by = c('month' = 'month',
 'district'= 'district'))
panel_data <- left_join(panel_data, panel_property,  by = c('month' = 'month
', 'district'= 'district'))

# re-order the column as the sequence of the assignment questions
panel_data <- panel_data[c("month", 'district',"crime_total_per_resident", "c
rime_vio_per_resident",
                           'crime_pro_per_resident', 'median_income', 'share_
black',
                           'share_hispanic', 'share_white')]

# sort the panel_data by district (unit)
panel_data = arrange(panel_data, district)

# In the officers.csv data set, district name was called unit, so I changed t
he colname of 'district' to 'unit'
#     for consistency
colnames(panel_data)[2] = c('unit')

# I did not drop the na values
print(panel_data[1:10, 1:5])

## # A tibble: 10 x 5
## # Groups:   month [10]
##    month      unit  crime_total_per_res~ crime_vio_per_resi~ crime_pro_per
_resi~
##    <date>     <fct>                <dbl>               <dbl>
 <dbl>
##  1 2002-02-01 1                       NA                  NA
    NA
##  2 2002-03-01 1                       NA                  NA
    NA
##  3 2002-04-01 1                       NA                  NA
    NA
##  4 2002-05-01 1                       NA                  NA
    NA
##  5 2002-06-01 1                       NA                  NA
    NA
```

```
##  6 2002-07-01 1                       NA                  NA
    NA
##  7 2002-08-01 1                       NA                  NA
    NA
##  8 2002-09-01 1                       NA                  NA
    NA
##  9 2002-10-01 1                       NA                  NA
    NA
## 10 2002-11-01 1                       NA                  NA
    NA

#==========================
#Exercise 3
#==========================

# plot the arrest number over time with 1 particular officer, just to briefly
 understand the data
p <- ggplot(filter(officers, NUID == 1), aes(x=month, y= arrest)) + geom_line
() +  scale_x_date(date_labels = "%Y %b %d")
print(p)
```



```
# merge the officers data and the panel_data from exercise 2
officers$unit = as.factor(officers$unit)
mer_ex3 <- left_join(officers, panel_data, by = c('month' = 'month', 'unit' =
 'unit'))
```

```
# estimate the ols model
eg3_1 <- lm(arrest ~ tenure +  crime_total_per_resident +  median_income +  s
hare_black +
                     share_hispanic +   share_white , data = mer_ex3)
print(summary(eg3_1))
```

```
##
## Call:
## lm(formula = arrest ~ tenure + crime_total_per_resident + median_income +
##     share_black + share_hispanic + share_white, data = mer_ex3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5017 -0.4993 -0.4982  0.5009  5.5027
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                5.062e-01  1.275e-02  39.714   <2e-16 ***
## tenure                    -4.375e-06  8.335e-06  -0.525    0.600
## crime_total_per_resident -2.341e-01  9.561e-01  -0.245    0.807
## median_income              3.035e-08  9.585e-08   0.317    0.752
## share_black               -6.936e-03  1.263e-02  -0.549    0.583
## share_hispanic            -4.651e-03  1.327e-02  -0.350    0.726
## share_white               -1.244e-02  1.701e-02  -0.731    0.465
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7068 on 1077902 degrees of freedom
##   (27 observations deleted due to missingness)
## Multiple R-squared:  2.066e-06,  Adjusted R-squared:  -3.5e-06
## F-statistic: 0.3711 on 6 and 1077902 DF,  p-value: 0.8977
```

```
# printCoefmat(coeftest(eg3_1, vcov = sandwich))

#==========================
#Exercise 4
#==========================

# estimate the ols model with unit and time fixed effect
# this code will take some time (not over 1 minute) to run
eg4 <- lm(arrest ~ tenure +  crime_total_per_resident +  median_income +  sha
re_black +
              share_hispanic +  share_white + as.factor(unit) + as.factor(mon
th) , data = mer_ex3)

# don't want to report the coefficients of fixed effect
printCoefmat(coeftest(eg4, vcov = sandwich)[1:7,])
```

```
##                              Estimate  Std. Error t value  Pr(>|t|)
## (Intercept)                6.3524e-01  1.0562e-01  6.0144 1.806e-09 ***
## tenure                    -3.7244e-06  8.5067e-06 -0.4378    0.6615
## crime_total_per_resident -4.2838e+00  2.8547e+00 -1.5006    0.1335
## median_income             2.4308e-08  6.4636e-07  0.0376    0.9700
## share_black              -6.7213e-02  1.0411e-01 -0.6456    0.5186
## share_hispanic           -1.2431e-01  2.0151e-01 -0.6169    0.5373
## share_white              -1.7357e-01  1.8316e-01 -0.9477    0.3433
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

#=========================
#Exercise 5
#=========================

#-----
#q1
#-----

# using the data from exercise 3, I calculate the mean for every column for t
he within and between estimator
avg_overtime <-  mer_ex3 %>%
  group_by(NUID, unit) %>%
  summarize(num_period = n(),
            avg_arrest = mean(arrest),
            avg_tenure = mean(tenure),
            avg_totalcrime = mean(crime_total_per_resident),
            avg_median_inc = mean(median_income),
            avg_black = mean(share_black),
            avg_hispanic = mean(share_hispanic),
            avg_white = mean(share_white)
            )

## `summarise()` has grouped output by 'NUID'. You can override using the `.g
roups` argument.

# after creating the mean, I left-merge this avg dataframe to the original da
taframe, so I can calculate the demean estimator
# I later figure it out that I have a much smarter way to do so. I can just c
alculat the demean estimator in the last step all at once
mer_ex5 <- left_join(mer_ex3, avg_overtime, by = c('NUID' = "NUID", 'unit' =
'unit'))



dat_within_all <- mer_ex5 %>% mutate(wi_arrest = arrest - avg_arrest,
                                     wi_tenure = tenure - avg_tenure,
                                     wi_totalcri = crime_total_per_resident - avg
_totalcrime,
                                     wi_median_inc = median_income - avg_median_i
nc,
                                     wi_black = share_black - avg_black,
                                     wi_hispanic = share_hispanic - avg_hispanic,
                                 wi_white = share_white - avg_white
                                 )


dat_within <- dat_within_all[-4:-20]
dat_between <- dat_within_all[-4:-13]

print(dat_within[1:10, 1:5])

##    NUID      month unit  wi_arrest wi_tenure
## 1      1 2007-03-01   14  1.5315315 -55.20721
## 2      1 2007-04-01   14 -0.4684685 -54.20721
## 3      1 2007-05-01   14  0.5315315 -53.20721
## 4      1 2007-06-01   14  0.5315315 -52.20721
## 5      1 2007-07-01   14 -0.4684685 -51.20721
## 6      1 2007-08-01   14 -0.4684685 -50.20721
## 7      1 2007-09-01   14  0.5315315 -49.20721
## 8      1 2007-10-01   14 -0.4684685 -48.20721
## 9      1 2007-11-01   14 -0.4684685 -47.20721
## 10     1 2007-12-01   14  1.5315315 -46.20721

print(dat_between[1:10, 1:5])

##    NUID      month unit avg_arrest avg_tenure
## 1      1 2007-03-01   14  0.4684685   73.20721
## 2      1 2007-04-01   14  0.4684685   73.20721
## 3      1 2007-05-01   14  0.4684685   73.20721
## 4      1 2007-06-01   14  0.4684685   73.20721
## 5      1 2007-07-01   14  0.4684685   73.20721
## 6      1 2007-08-01   14  0.4684685   73.20721
## 7      1 2007-09-01   14  0.4684685   73.20721
## 8      1 2007-10-01   14  0.4684685   73.20721
## 9      1 2007-11-01   14  0.4684685   73.20721
## 10     1 2007-12-01   14  0.4684685   73.20721

# first difference

# to find the first difference column, I first sort the time index in  descen
ding order
dat_firstdiff <- mer_ex3 %>% arrange(NUID, unit, desc(month))

# calculate Yt- Tt-1 in the individual, unit level
dat_firstdiff <- dat_firstdiff %>%
  group_by(NUID, unit) %>%
  mutate(fd_tenure =  tenure - lag(tenure),
         fd_arrest = arrest - lag(arrest),
         fd_crimetot = crime_total_per_resident - lag(crime_total_per_residen
t),
         fd_medinc = median_income - lag(median_income),
         fd_black = share_black - lag(share_black),
         fd_hispanic = share_hispanic - lag(share_hispanic),
         fd_white = share_white - lag(share_white)
         )
dat_firstdiff <- dat_firstdiff[-4:-12]

# first difference will generate NA values(the first one in the window), so I
 dropped the na
dat_firstdiff_noNA <- na.omit(dat_firstdiff)

print(dat_firstdiff_noNA[1:10, 1:5])

## # A tibble: 10 x 5
## # Groups:   NUID, unit [2]
##     NUID month      unit  fd_tenure fd_arrest
##    <int> <date>     <fct>     <int>     <int>
##  1     1 2017-02-01 3             0         0
##  2     1 2017-01-01 3            -1         2
##  3     1 2016-12-01 3            -1        -2
##  4     1 2016-11-01 3            -1         0
##  5     1 2016-10-01 3            -2         1
##  6     1 2016-09-01 3             0         0
##  7     1 2016-08-01 3            -2         0
##  8     1 2016-07-01 3            -1         1
##  9     1 2016-06-01 3            -1        -2
## 10     1 2016-04-01 14           -1         0

# implement within, between, firstdiff estimators
eg5_within <- lm(wi_arrest ~ wi_tenure +  wi_totalcri +  wi_median_inc +  wi_
black +
                   wi_hispanic +  wi_white + as.factor(unit) + as.factor(mont
h) , data = dat_within)

eg5_between <- lm(avg_arrest ~ avg_tenure +  avg_totalcrime +  avg_median_inc
 +  avg_black +
                    avg_hispanic +  avg_white + as.factor(unit) + as.factor(m
onth) , data = dat_between)

eg5_firstdiff <- lm(fd_arrest ~ fd_tenure +  fd_crimetot +  fd_medinc +  fd_b
lack +
                      fd_hispanic +  fd_white + as.factor(unit) + as.factor(m
onth) , data =
                      dat_firstdiff_noNA)

# model summary for the three models
# don't want to print out fixed effect

printCoefmat(coeftest(eg5_within, vcov = sandwich)[1:7,])

##                  Estimate  Std. Error t value Pr(>|t|)
## (Intercept)   -5.8573e-04  8.4726e-03 -0.0691   0.9449
## wi_tenure     -2.5938e-05  4.3432e-05 -0.5972   0.5504
## wi_totalcri   -5.0064e+00  3.0864e+00 -1.6221   0.1048
## wi_median_inc -3.4098e-07  8.2356e-07 -0.4140   0.6788
## wi_black      -5.6940e-02  1.2148e-01 -0.4687   0.6393
## wi_hispanic   -4.2864e-02  2.3204e-01 -0.1847   0.8534
## wi_white      -1.1691e-01  2.1264e-01 -0.5498   0.5825
```

```
printCoefmat(coeftest(eg5_between, vcov = sandwich)[1:7,])
```

```
##                    Estimate  Std. Error t value  Pr(>|t|)
## (Intercept)     6.5289e-01  3.7656e-02 17.3385 < 2.2e-16 ***
## avg_tenure     -6.6645e-06  1.4172e-06 -4.7025  2.57e-06 ***
## avg_totalcrime  1.2339e+00  7.8998e-01  1.5619 0.1183096
## avg_median_inc  3.4658e-07  1.9630e-07  1.7655 0.0774768 .
## avg_black      -1.2488e-01  3.9049e-02 -3.1981 0.0013834 **
## avg_hispanic   -2.9849e-01  7.7058e-02 -3.8735 0.0001073 ***
## avg_white      -2.5700e-01  6.7894e-02 -3.7854 0.0001535 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
printCoefmat(coeftest(eg5_firstdiff, vcov = sandwich)[1:7,])
```

```
##                 Estimate  Std. Error t value Pr(>|t|)
## (Intercept) -3.2939e-03  1.2100e-02 -0.2722   0.7854
## fd_tenure    6.1832e-05  7.8172e-04  0.0791   0.9370
## fd_crimetot -1.7889e+00  5.0981e+00 -0.3509   0.7257
## fd_medinc    4.9655e-06  4.1466e-06  1.1975   0.2311
## fd_black     7.9884e-02  7.7457e-01  0.1031   0.9179
## fd_hispanic  3.7644e-01  1.4991e+00  0.2511   0.8017
## fd_white    -8.5363e-01  1.3304e+00 -0.6416   0.5211
```

```
#-----
#q2
#-----
#GMM approach

# after trying really hard, I did not solve this question successfully, but p
lease see my code and thinking process, thank you!

# create a data frame w/o na value
mer_ex5_no_na <- mer_ex3 %>% drop_na()

# reorder the column because I want to slice the data frame and create the in
dependent variables matrix
mer_ex5_no_na <- mer_ex5_no_na[c("NUID", "arrest", "unit", "month", "tenure",
 "crime_total_per_resident",
                                "median_income", "share_black" ,  "share_his
panic", "share_white" )]

# create factor variables and drop the month fixed effect, because I have a h
ard time trying to estimate #   all the beta parameters and unit, month fixed
```

```
 effect
# I focus on unit fixed effect here
mer_ex5_no_na <- subset(mer_ex5_no_na, select = -c(month) )
# mer_ex5_no_na$month <-  as.factor(mer_ex5_no_na$month)

# create a factor unit variable
mer_ex5_no_na$unit <- as.factor(mer_ex5_no_na$unit)

# because I want to estimate the unit fixed effect, I use this library to cre
ate 0, 1  encoded dummy variable
mer_ex5_final <- fastDummies::dummy_cols(mer_ex5_no_na)

# change the median_income to log scale
mer_ex5_final$median_income = log(mer_ex5_final$median_income)

# create independent variables matrix and dependent variable vector
ex5_ind <- as.matrix(mer_ex5_final[, 4:34])
ex5_dep <- as.matrix(mer_ex5_final[, 2])

# calculate the numbers of parameters
n_individual = nrow(ex5_ind)
n_estimators = ncol(ex5_ind)  # including unit fixed effect and beta, gamma s
pecified in the assignment
n_par =  n_estimators

print(n_individual)

## [1] 1077909

print(n_estimators)

## [1] 31

print(n_par)

## [1] 31

# method of moments starts here
# calculate the variance of the moments by boot strap.
nboot   = 9
mom_mat = mat.or.vec(n_par,nboot)
for (iN in 1:nboot)
{
  xs           = sample(ex5_dep, n_individual,replace=T)  # ex5_dep is the de
pendent (observed) variable
  mom          <- all.moments(xs, order.max= n_par )
  # If I understand  correctly, we need at least as many moments as the numbe
rs of our estimated
  #    parameters. Hence, I calculate until the n_par(th) moments
  # However, here comes the bug I can't solve. In my model, I need to estimat
e n_par = 31 estimators.
```

```
  # Calculating until the 31th moment makes my computer really slow, and I am
 still trying to solve this   # problem

   mom_mat[,iN] = mom[-1]
}

vs = apply(mom_mat,1,var) # the bootstrap variance for the sandwich formula
print(vs)
```

```
##  [1] 5.029560e-07 2.136206e-06 2.115732e-05 2.797419e-04 4.062087e-03
##  [6] 6.247493e-02 1.023400e+00 1.826826e+01 3.677823e+02 8.623934e+03
## [11] 2.368649e+05 7.398772e+06 2.513519e+08 8.944538e+09 3.255576e+11
## [16] 1.196186e+13 4.407341e+14 1.623211e+16 5.967334e+17 2.188617e+19
## [21] 8.007756e+20 2.923221e+22 1.064922e+24 3.872455e+25 1.405960e+27
## [26] 5.097706e+28 1.846202e+30 6.679786e+31 2.414853e+33 8.724088e+34
## [31] 3.149913e+36
```

```
# I define this function to take the parameters of the model, variance for th
e sandwich formula, and the independent variables matrix, and the dependent v
ariable vector

mm_data = function(param,vs, ex5_dep, ex5_ind)
{
  data_mom = mat.or.vec(n_par,1)

   # I try to implement a linear model and calculate the error term
  error_term  =   ex5_dep - ex5_ind %*% param

  # calculate the moments of the error term and store it as a vector
  data_mom = all.moments(error_term, order.max = n_par )[-1]

  # calculate and return the criterion function
  crit    = (t(t(data_mom)))*(1/vs)*(data_mom )
  return(sum(crit));
}

# using the random starting point to optimize
# start = runif(n_par, -10, 10)

# The following optim function takes a long time to run, and I am sure someth
ing goes wrong
# after a while, it shows that the initial  value is 313347043802720695466248
6208042262606446262.000000, which is not reasonable, I guess

# res  = optim(start,fn= mm_data,method="BFGS",control=list(trace=6,REPORT=1,
maxit=1000),vs=vs,ex5_dep = # ex5_dep, ex5_ind = ex5_ind)

# the following code res$par will yield an error, because I successfully opti
mize and obtain res
# param    = res$par
```

using the random starting point to optimize

```
start = runif(n_par, -10, 10)
```

The following optim function takes a long time to run, and I am sure something goes wrong

after a while, it shows that the initial value is 3133470438027206954662486208042262606446262.000000, which is unreasonably large.

```
res = optim(start,fn= mm_data,method=“BFGS”,control=list(trace=6,REPORT=1,maxit=1000),vs=vs,ex5_dep = ex5_dep, ex5_ind = ex5_ind)
```

the following code res$par will yield an error, because I did not successfully optimize and obtain res # param = res$par