ME\_Assignment\_4

Jia Ru

2016-03-22

(I use R instead of STATA for this homework)

# Question 1

Consider the following wage rate equation specifications:

where stand for years of schooling, experience, industry dummies and occupational dumming, and is the union status dummy.

estimate (1) and (2) by:  
1. covariance method (Least Squares dummy variable)  
2. generalized method of moments estimator  
3. Random Effects Estimator

Use your results to answer the following questions:  
(a) Your preferred specification.  
(b) Does union membership raise wage rate?

#########################################################  
# Q1   
########################################################  
  
# Import data  
rm(list = ls())  
library(readxl)  
data <- read\_excel(path = "data\_assignment4.xls", col\_names = TRUE)  
  
# generate lag\_LWAGE  
data <- data[order(data$id,data$time),] # sort  
data$lag\_LWAGE <- rep(NA,nrow(data))  
for (i in 1:nrow(data)){  
 if (i%%7==1) next  
 data[i,"lag\_LWAGE"] <- data[i-1,"LWAGE"]  
}  
# set panel data structure  
library(plm)  
pdata <- plm.data(x = data,indexes = c("id","time"))

### (1) covariance method (Least Squares dummy variable)

# (1) LSDV  
  
lsdv <- lm(LWAGE~ED+EXP+IND+OCC+UNION+factor(id),data = data)  
lsdv\_lag <- lm(LWAGE~ED+EXP+IND+OCC+UNION+factor(id)+lag\_LWAGE,data = data)  
  
summary(lsdv\_lag)$coefficients[1:6,]  
summary(lsdv)$coefficients[1:6,]

## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 3.74552012 0.315090685 11.8871179 7.207466e-32  
## ED 0.07857988 0.028731307 2.7349913 6.275242e-03  
## EXP 0.07651207 0.002318759 32.9969982 1.055277e-203  
## IND 0.01259265 0.016705404 0.7538067 4.510250e-01  
## OCC -0.02704422 0.015185098 -1.7809708 7.501940e-02  
## UNION 0.01820191 0.016487117 1.1040081 2.696790e-01  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 4.47462629 0.291439115 15.353554 1.446003e-51  
## ED 0.10008975 0.027430646 3.648830 2.672098e-04  
## EXP 0.09677881 0.001189885 81.334562 0.000000e+00  
## IND 0.02018713 0.015580279 1.295685 1.951679e-01  
## OCC -0.02384978 0.013846470 -1.722445 8.507572e-02  
## UNION 0.03420595 0.015047170 2.273248 2.307046e-02

### (2) generalized method of moments estimator

library(gmm)  
  
gmm <- gmm(g = LWAGE~ED+EXP+IND+OCC+UNION,   
 x = ~ED+EXP+IND+OCC+UNION,  
 data = data)  
  
gmm\_lag <- gmm(g = LWAGE~ED+EXP+IND+OCC+UNION+lag\_LWAGE,   
 x = ~ED+EXP+IND+OCC+UNION+lag\_LWAGE,  
 data = data)  
  
gmm$coefficients  
gmm\_lag$coefficients

## (Intercept) ED EXP IND OCC UNION   
## 5.51357587 0.07028259 0.01212397 0.10838322 -0.14879002 0.14411194   
## (Intercept) ED EXP IND OCC   
## 0.6669967774 0.0087994828 0.0002018259 0.0235619691 -0.0226658954   
## UNION lag\_LWAGE   
## 0.0145896230 0.8957378543

### (3) Random Effects Estimator

# (3) Random Effects   
  
re <- plm(LWAGE~ED+EXP+IND+OCC+UNION, model = "random", data = pdata)  
re\_lag <- plm(LWAGE~ED+EXP+IND+OCC+UNION+lag\_LWAGE, model = "random", data = pdata)  
  
summary(re\_lag)$coefficients  
re\_lag$ercomp # estimation of the components of the errors of RE model  
  
summary(re)$coefficients  
re$ercomp

## Estimate Std. Error t-value Pr(>|t|)  
## (Intercept) 0.368600977 0.0379089967 9.723311 4.500404e-22  
## ED 0.005446663 0.0011052008 4.928211 8.676635e-07  
## EXP -0.000419706 0.0002141759 -1.959632 5.011668e-02  
## IND 0.017568651 0.0047506165 3.698183 2.204215e-04  
## OCC -0.011465674 0.0061317760 -1.869878 6.158270e-02  
## UNION 0.006007399 0.0050617266 1.186828 2.353746e-01  
## lag\_LWAGE 0.949113832 0.0061341387 154.726503 0.000000e+00  
## var std.dev share  
## idiosyncratic 0.021979 0.148254 1.095  
## individual -0.001915 NA -0.095  
## theta: -0.4476   
## Estimate Std. Error t-value Pr(>|t|)  
## (Intercept) 4.13875825 0.093716241 44.1626573 0.000000e+00  
## ED 0.11137433 0.006400367 17.4012413 1.605957e-65  
## EXP 0.05542692 0.001097691 50.4941092 0.000000e+00  
## IND 0.01571681 0.017506697 0.8977599 3.693655e-01  
## OCC -0.04569672 0.016588057 -2.7547962 5.898424e-03  
## UNION 0.06498127 0.017227059 3.7720468 1.641656e-04  
## var std.dev share  
## idiosyncratic 0.02353 0.15341 0.191  
## individual 0.09939 0.31527 0.809  
## theta: 0.8191

# estimation result  
  
library(texreg)  
  
screenreg(  
 l = list(lsdv,lsdv\_lag,gmm,gmm\_lag,re,re\_lag),  
 omit.coef = "id",  
 custom.model.names = c("lsdv","lsdv\_lag","gmm","gmm\_lag","re","re\_lag")  
 )

##   
## ================================================================================================  
## lsdv lsdv\_lag gmm gmm\_lag re re\_lag   
## ------------------------------------------------------------------------------------------------  
## (Intercept) 4.47 \*\*\* 3.75 \*\*\* 5.51 \*\*\* 0.67 \*\*\* 4.14 \*\*\* 0.37 \*\*\*  
## (0.29) (0.32) (0.10) (0.09) (0.09) (0.04)   
## ED 0.10 \*\*\* 0.08 \*\* 0.07 \*\*\* 0.01 \*\*\* 0.11 \*\*\* 0.01 \*\*\*  
## (0.03) (0.03) (0.01) (0.00) (0.01) (0.00)   
## EXP 0.10 \*\*\* 0.08 \*\*\* 0.01 \*\*\* 0.00 0.06 \*\*\* -0.00   
## (0.00) (0.00) (0.00) (0.00) (0.00) (0.00)   
## IND 0.02 0.01 0.11 \*\*\* 0.02 \*\*\* 0.02 0.02 \*\*\*  
## (0.02) (0.02) (0.03) (0.01) (0.02) (0.00)   
## OCC -0.02 -0.03 -0.15 \*\*\* -0.02 \*\*\* -0.05 \*\* -0.01   
## (0.01) (0.02) (0.03) (0.01) (0.02) (0.01)   
## UNION 0.03 \* 0.02 0.14 \*\*\* 0.01 \*\* 0.06 \*\*\* 0.01   
## (0.02) (0.02) (0.03) (0.01) (0.02) (0.01)   
## lag\_LWAGE 0.18 \*\*\* 0.90 \*\*\* 0.95 \*\*\*  
## (0.02) (0.02) (0.01)   
## ------------------------------------------------------------------------------------------------  
## R^2 0.91 0.91 0.40 0.91   
## Adj. R^2 0.89 0.89 0.40 0.91   
## Num. obs. 4165 3570 4165 3570 4165 3570   
## RMSE 0.15 0.15   
## Criterion function 0.00 0.00   
## ================================================================================================  
## \*\*\* p < 0.001, \*\* p < 0.01, \* p < 0.05

### Answer the question:

#### (a) Your preferred specification.

I prefer the LSDV specification. As I argued in Assignment-3, there is endogeneity problem, i.e, the heterogeneity term is correlated with covariates , so both GMM and random effect model is not appropreate.

Also I think the equation (1) (with lag term of LWAGE) is more appropriate, the reason is same as above: FD(first-order difference) model eliminates the individual effect .

#### (b) Does union membership raise wage rate?

According to the reasoning above, I consider ladv\_lag model. The coefficent of UNION in the model is 0.01, not significant. So I think union membership does not necessarily raise wage rate.

Moreover, although in other models the coefficient is significant, it only stands for correlation relationship, not causal relationship.

# Question 2

Consider the model

Generate observations of and throw away the first observations. Consider the case of and .

Estimate by the  
1. simple instrumental variable method  
2. GMM  
3. MLE  
Construct the t-statitic for the null:  
Replicate the experiment 1000 times. Find the actual size based on different estimators using the critical value of 1.96. (for the nominal significance level of 5%).

ANS:

First I define a funciton to do **one time** simulation, then when I do 1000 times simulation I will just invoke this function.

###################################################################  
# This chunk defines the one time simulation function :  
# arguments: N(number of individuals) and TT(number of time)  
# STEP\_1 simulate DGP  
# STEP\_2 do IV, GMM, MLE regression and hypothesis test  
# return: a list of 3, which indicates whether to reject H\_0: gamma==0.05 in the three model specifications.   
###################################################################  
  
rm(list=ls())  
  
SM <- function(N,TT) {  
   
################# DGP ##################################  
  
#N <- 200  
#TT <- 5 or 54  
# in R, "T"" stands for boolean "True" , so use "TT" to escape.  
gamma <- 0.5  
  
# initialize data  
data2 <- as.data.frame(matrix(NA, nrow = N\*TT, ncol = 6), colnames)  
names(data2) <- c("y","lag\_y","a","u","id","time")   
  
a <- rnorm(N)  
for (i in 1:N ){  
   
 data2[((i-1)\*TT+1):(i\*TT),"id"] <- i # id  
 data2[((i-1)\*TT+1):(i\*TT),"time"] <- 1:TT # time  
 data2[((i-1)\*TT+1):(i\*TT),"a"] <- a[i] # for the same individual (i) alpha is same  
   
 y <- rep(0,200+TT) # initialize y\_i # vector  
 u <- rnorm(200+TT) # generate u\_i # vector  
 for (t in 2:(200+TT)) {  
 y[t] <- gamma\*y[t-1] + a[i] + u[t]  
 }  
   
 data2[((i-1)\*TT+1):(i\*TT),"y"] <- y[201:(200+TT)]  
 data2[((i-1)\*TT+1):(i\*TT),"lag\_y"] <- y[200:(200+TT-1)]  
 data2[((i-1)\*TT+1):(i\*TT),"u"] <- u[201:(200+TT)]  
   
}  
data2$id <- factor(data2$id) # in order to treat id as dummy in regression  
  
  
######################## simple IV ########################  
library(magrittr)   
library(AER)  
  
iv <- ivreg(y~lag\_y|lag\_y, data = data2)  
# equivalent to  
lm <- lm(y~lag\_y,data=data2)  
  
summary(iv)  
# hypothesis test  
hyp <- linearHypothesis(iv,"lag\_y=0.5",test = "F")   
reject <- hyp$`Pr(>F)`[2]>0.05  
iv\_test <- as.numeric(reject) # whether reject H\_0  
  
  
  
######################## GMM ########################  
gmm <- gmm(g = y~lag\_y,   
 x = ~lag\_y,  
 data = data2)  
gmm  
# hypothesis test  
hyp <- linearHypothesis(gmm,"lag\_y=0.5",test = "F")   
reject <- hyp$`Pr(>F)`[2]>0.05  
gmm\_test <- as.numeric(reject) # whether reject H\_0  
  
  
  
######################## MLE ########################  
library(stats4)  
y <- data2$y  
x <- data2$lag\_y  
  
LL <- function(gamma, mu, sigma) {  
 R = y - x \* gamma  
 R = suppressWarnings(dnorm(R, mu, sigma, log = TRUE))  
 -sum(R)  
}  
mle <- mle(LL, start = list(gamma = 0.5, mu = 0, sigma=1))  
  
mle  
  
# construct t test  
gamma\_hat <- summary(mle)@coef["gamma","Estimate"]  
gamma\_se <- summary(mle)@coef["gamma","Std. Error"]   
t <- (gamma\_hat-gamma)/gamma\_se   
reject <- abs(t)>1.96   
mle\_test <- as.numeric(reject)  
  
return(list("iv"=iv\_test,"gmm"=gmm\_test,"mle"=mle\_test))  
  
}

Next, define a function of 1000 times simulation procedure. This function returns the simulated size of the hypothesis testing of the three regression methods

###################################################################  
# This chunk defines the 1000 times simulation function :  
# arguments: N and TT, and n\_sim=1000 is fixed.  
# STEP\_1 invoke function SM, replicated it 1000 times  
# STEP\_2 calculate the proportion that reject/accepts H\_0, this is the simulated size.  
# return: a list of 3, which is the simulated size.  
###################################################################  
  
n\_sim <- 1000  
  
SMsize <- function(N,TT) {  
 N\_v <- rep(N, times = n\_sim)  
 TT\_v <- rep(TT, times = n\_sim)  
   
 m <- mapply(FUN=SM, N=N\_v,TT=TT\_v)  
 mat <- matrix(  
 unlist(m),  
 ncol=n\_sim, nrow=3, byrow=F,   
 row.names(c("iv","gmm","mle"))  
 )  
 size <- rowSums(mat)/n\_sim  
 size3 <- list("size\_iv"=size[[1]],  
 "size\_gmm"=size[[2]],  
 "size\_mle"=size[[3]]  
 )  
 return(size3)  
}

Now invoke the function SMsize() to do two type of simulation and see the sizes:

print("(1) N=200, T=5")  
SMsize(N=200,TT=5)  
  
print("(1) N=200, T=54")  
SMsize(N=200,TT=54)

## [1] "(1) N=200, T=5"  
## $size\_iv  
## [1] 0.7272727  
##   
## $size\_gmm  
## [1] 0.6363636  
##   
## $size\_mle  
## [1] 0.3636364  
##   
## [1] "(1) N=200, T=54"  
## $size\_iv  
## [1] 0.1818182  
##   
## $size\_gmm  
## [1] 0.1818182  
##   
## $size\_mle  
## [1] 0.8181818

from the results we can see:  
(1) when T is small, iv and gmm is closer to the "real world", while mle is not.  
(2) when T is large, however, mle is performs better.