HW4

Jie Ren

April 4, 2019

Exercise 1

```
rm(list=ls())
setwd("C:/Users/jiere/Dropbox/Spring 2019/ECON 613/ECON613_HW/HW4_output")
# install.packages("lme4")
# install.packages("Matrix")
# install.packages("ggplot2")
# install.packages("reshape2")
library("reshape2")
library("ggplot2")
library("lme4")
kt <- read.csv("Koop-Tobias.csv")
```

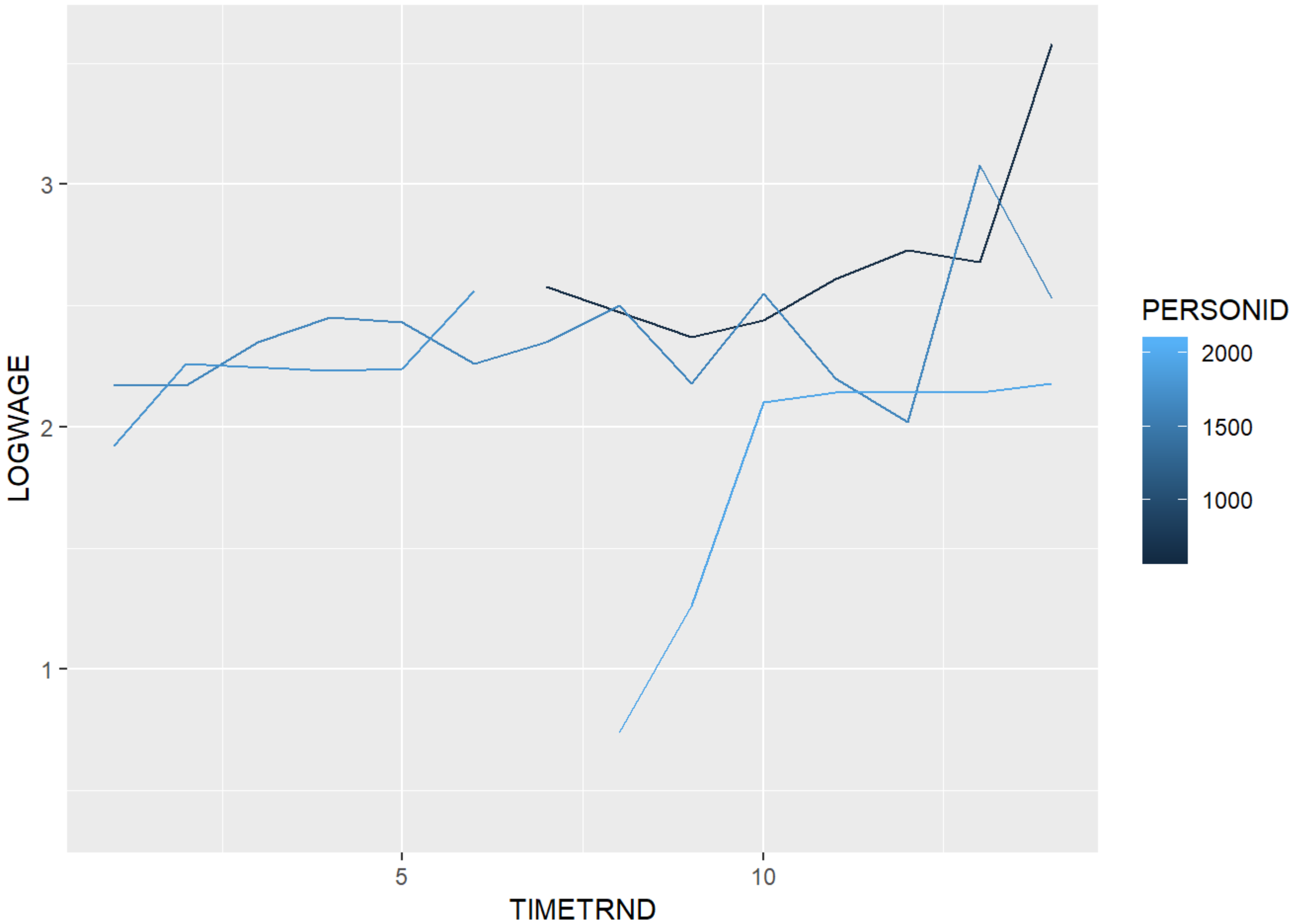
Find the sample dimension for 5 randomly selected individual

```
rd <- sample(1:2178,5)
dimension <- aggregate(list(Dimension = kt$PERSONID), list(PERSONID = kt$PERSONID), length)[rd,]
rownames(dimension) <- NULL
dimension
```

```
##   PERSONID Dimension
## 1      912         4
## 2     1738        11
## 3     1591         3
## 4     1426         2
## 5     1763         4
```

```
rnd <- sample(1:2178,5)
kt.sub <- kt[kt$PERSONID %in% rnd,c("PERSONID","LOGWAGE","TIMETRND")]

ggplot(kt.sub, aes( x=TIMETRND, y=LOGWAGE, group=PERSONID, col=PERSONID)) +
  geom_line()
```



Noticed that this is an unbalanced panel and the time trend variable is not consecutive

Exercise 2

Check with lme4 package

```
re.lm <- lmer(LOGWAGE ~ EDUC + POTEXPER + (1|PERSONID), data = kt)
summary(re.lm)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: LOGWAGE ~ EDUC + POTEXPER + (1 | PERSONID)
##    Data: kt
##
## REML criterion at convergence: 16700.7
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -7.1639 -0.4559  0.0635  0.5351  7.3176
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  PERSONID (Intercept) 0.1330   0.3647
##  Residual             0.1129   0.3360
## Number of obs: 17919, groups:  PERSONID, 2178
##
## Fixed effects:
##              Estimate Std. Error t value
## (Intercept) 0.5667942  0.0434683   13.04
## EDUC        0.1077081  0.0033500   32.15
## POTEXPER    0.0387584  0.0007186   53.93
##
## Correlation of Fixed Effects:
##          (Intr) EDUC
## EDUC     -0.972
## POTEXPER -0.066 -0.070
```

Exercise 2&3

Calculate the random effect using the transformed model (Mannually)

(Method from Principle of Econometrics) Fisrt let's define a function for between estimator

```
tmean <- function(x,id,rep = T){# calculate the mean for each individual id, and and repeat thes
e mean for the same id
  # or alternatively use ave
  dim     <- aggregate(list(Tp = id), list(id = id), length)
  mean    <- aggregate(list(idmean = x),list(id = id),mean)
  gpmean  <- rep(mean$idmean,dim$Tp)
  ifelse(rep == T, return(gpmean),return(mean$idmean))
}


fix.bt <- function(y,X,id){
  dep     <- tmean(y,id,rep = F)
  indep   <- apply(X,2,tmean, id = id, rep = F)
  result  <- lm(dep~indep)
  return(result)
}
```

The let's define a funciton for with-in estimator

```
fix.wi <- function(y,X,id){
  dep     <- y - tmean(y,id)
  indep   <- as.matrix(X - apply(X,2,tmean, id = id))
  result  <- lm(dep~0+indep)
  return(result)
}
```

Using the variance from these two estimator, we are able the caculate the vairance of residual in random effect estimator. Then we can get get the transformed model and simply do OLS!

```
fix.re.ib <- function(y,X,id){
  N <- length(id)
  n <- length(unique(id))
  k <- ncol(X)
  sigma2_u <- sum((summary(fix.bt(y,X,id))$residual)^2)/(n-k)
  sigma2_e <- sum((summary(fix.wi(y,X,id))$residual)^2)/((N-n)-k)

  # for unbalanced panel use harmonized mean of time period of each id has
  dim       <- aggregate(list(Tp = id), list(id = id), length)
  Th        <- length(unique(id))/sum(1/dim$Tp)
  sigma2_v  <- sigma2_u - sigma2_e/Th

  # Calculate lumbda for the tranformed model
  Tp_i    <- rep(dim$Tp,dim$Tp)
  lambda  <- 1-sqrt(sigma2_e/(Tp_i*sigma2_v + sigma2_e)) # special case of unbalanced panel

  # Transformed model
  X       <- cbind(Intercept = 1,X)
  dep     <- y - lambda*tmean(y,id)
  indep   <- as.matrix(X - rep(lambda,ncol(X))*apply(X,2,tmean, id = id))
  result  <- lm(dep~0+indep)
  return(result)
}
```

First time difference estimator

Here we regard the discouninous timetime trend as continous, per Professor Sidibe.

```r
fix.fd <- function(y,X,id){
  df <- data.frame(y,X,id)
  for (i in 1:ncol(X)){
    df <- transform(df, col=ave(df[,i+1], df$id, FUN = function(x) c(NA, diff(x)))) # MUST inclu
de FUN, or cause error
    names(df)[ncol(df)]<-paste("indep",i,sep=" ")
  }
  df <- transform(df, dep=ave(y, id, FUN = function(x) c(NA, diff(x))))

  indep   <- as.matrix(na.omit(df[,grepl("indep",colnames(df))]))
  dep     <- na.omit(df[,"dep"])
  result  <- lm(dep~0+indep)
  return(result)
}

coef(fix.re.ib(kt$LOGWAGE,kt[,c("EDUC","POTEXPER")],kt$PERSONID)) # random
```

```
## indepIntercept      indepEDUC  indepPOTEXPER
##     0.56356104     0.10793517     0.03876441
```

```r
coef(fix.wi(kt$LOGWAGE,kt[,c("EDUC","POTEXPER")],kt$PERSONID)) # with-in
```

```
##     indepEDUC indepPOTEXPER
##    0.12366202    0.03856107
```

```r
coef(fix.bt(kt$LOGWAGE,kt[,c("EDUC","POTEXPER")],kt$PERSONID)) # between
```

```
##   (Intercept)     indepEDUC indepPOTEXPER
##    0.84556883    0.09309987    0.02599874
```

```r
coef(fix.fd(kt$LOGWAGE,kt[,c("EDUC","POTEXPER")],kt$PERSONID)) # first difference
```

```
## indepindep.1 indepindep.2
##   0.04793559   0.03286052
```

Their result are close for random effect and with-in estimator, but others are quite different

Exercise 4

```
# randomly selection 100 individual
rnd <- sample(1:2178,100)
kt.rand <- kt[kt$PERSONID %in% rnd,]
fix.dvls <- function(df, y_name, X_name,id_name,dmatrix = F){
  idcol <- which( colnames(df)== id_name )
  idx   <- unique(df[,idcol])
  for (i in 1:100){
    df$D <- 0
    df[df[,idcol] == idx[i],ncol(df)] <- 1
    names(df)[ncol(df)] <- paste("Dummy_",idx[i],sep = "")
  }
  dummy <- df[,grepl("Dummy_",colnames(df))]
  dummy <- dummy[,-1] # drop the first person to avoid dummy variable trap
  print(paste("use the first selected individual as reference, which id =",idx[1],sep = " "))

  dep       <- df[,y_name]
  indep     <- as.matrix(cbind(df[,X_name],dummy))
  result.d  <- lm(dep~0+indep)
  ifelse(dmatrix == F,return(result.d),return(dummy))
}
coef(fix.dvls(kt.rand,"LOGWAGE",c("EDUC","POTEXPER"),"PERSONID"))
```

```
## [1] "use the first selected individual as reference, which id = 2"
```

```
##       indepEDUC   indepPOTEXPER   indepDummy_33   indepDummy_43 
##     0.138715533     0.043396522     0.394005875     0.232749064 
##   indepDummy_61   indepDummy_77  indepDummy_109  indepDummy_147 
##     0.359771101     0.185381188     0.235062434     0.044578925 
##  indepDummy_175  indepDummy_184  indepDummy_233  indepDummy_263 
##     0.765822237     0.471089779     0.265709837     0.320217630 
##  indepDummy_270  indepDummy_276  indepDummy_285  indepDummy_296 
##    -0.459377515     0.560430997    -0.021829341     0.354540450 
##  indepDummy_306  indepDummy_337  indepDummy_344  indepDummy_350 
##     0.638430997     0.525840426    -0.443160059     0.535034475 
##  indepDummy_369  indepDummy_379  indepDummy_467  indepDummy_567 
##     0.107315123     0.205620540     0.435362773     0.430267127 
##  indepDummy_622  indepDummy_623  indepDummy_667  indepDummy_682 
##    -0.102551613     0.468227109    -0.368362047     0.406744228 
##  indepDummy_683  indepDummy_700  indepDummy_711  indepDummy_747 
##    -0.230379318     0.490781231    -0.212071659    -0.107775134 
##  indepDummy_755  indepDummy_778  indepDummy_799  indepDummy_801 
##    -0.201246270     0.138166259     0.451146767     0.281747930 
##  indepDummy_828  indepDummy_841  indepDummy_861  indepDummy_868 
##    -0.105853428     0.238183989     0.305442172     0.352436692 
##  indepDummy_893  indepDummy_932  indepDummy_938  indepDummy_957 
##     0.419210822     0.183241431     0.470151703    -0.357140441 
##  indepDummy_961 indepDummy_1046 indepDummy_1052 indepDummy_1071 
##     1.202174029    -0.034224296     0.330720423     0.798781602 
## indepDummy_1075 indepDummy_1089 indepDummy_1131 indepDummy_1150 
##    -0.098007562     0.332203518    -1.081723907     0.100034475 
## indepDummy_1151 indepDummy_1214 indepDummy_1260 indepDummy_1322 
##    -0.648592895     0.500711723     0.143860176    -0.355607758 
## indepDummy_1339 indepDummy_1354 indepDummy_1377 indepDummy_1380 
##     0.341778890     0.304072086     0.040048496     0.229494304 
## indepDummy_1402 indepDummy_1404 indepDummy_1446 indepDummy_1466 
##    -0.409107581     0.427982736    -0.329282732    -0.255118454 
## indepDummy_1484 indepDummy_1487 indepDummy_1509 indepDummy_1518 
##     0.067038823     0.854760273     0.401762113    -0.369700545 
## indepDummy_1529 indepDummy_1538 indepDummy_1540 indepDummy_1558 
##     0.310945489     0.006061654    -0.254982949     0.294676406 
## indepDummy_1566 indepDummy_1574 indepDummy_1599 indepDummy_1602 
##     0.490337276     0.505318824     0.101751120    -0.078077698 
## indepDummy_1628 indepDummy_1655 indepDummy_1678 indepDummy_1705 
##    -0.107103495     0.077162673     0.377133800    -0.139777917 
## indepDummy_1719 indepDummy_1740 indepDummy_1752 indepDummy_1767 
##    -0.165082011     0.091078913     0.001667613     0.585592253 
## indepDummy_1779 indepDummy_1788 indepDummy_1806 indepDummy_1837 
##    -0.255991102    -0.436198068    -0.561462728     0.144336214 
## indepDummy_1896 indepDummy_1901 indepDummy_1922 indepDummy_1924 
##    -0.543698329    -0.026462395     0.345230176    -1.003293112 
## indepDummy_1940 indepDummy_1956 indepDummy_1960 indepDummy_1981 
##    -0.336629124    -0.414205666    -0.146421177     0.214942760 
## indepDummy_2005 indepDummy_2012 indepDummy_2038 indepDummy_2057 
##     0.221717992     0.210975692     1.000219105     0.094331861 
## indepDummy_2148 
##     0.172186938
```

```
# do with MLE
dummy <- fix.dvls(kt.rand,"LOGWAGE",c("EDUC","POTEXPER"),"PERSONID",dmatrix = T)
```

```
## [1] "use the first selected individual as reference, which id = 2"
```

```
# Define likelihood function
ll.DVLS <- function(b){
  n    <- length(y)
  b    <- b[2:length(b)]
  sig2 <- b[1]
  ll   <- -n/2*log(2*pi)-n/2*log(sig2)-(y-X%*%b)^2/(2*sig2)
  ll.s <- -sum(ll)
  ll.s
  return(ll.s)
}

y <- kt.rand$LOGWAGE
X <- cbind(as.matrix(kt.rand[,c("EDUC","POTEXPER")]),as.matrix(dummy))

for (i in 1:10){
  tryCatch({
  set.seed(i)
  b <- rnorm(102)
  result <- optim(b, ll.DVLS)
  print(paste("Succeed but with convergence equal",result$convergence,sep = " "))
  }, error=function(e){print(paste("Failed with convergence equal",result$convergence,sep = " "
))})
}
```

```
## [1] "Succeed but with convergence equal 1"
## [1] "Succeed but with convergence equal 1"
## [1] "Failed with convergence equal 1"
## [1] "Failed with convergence equal 1"
## [1] "Succeed but with convergence equal 1"
## [1] "Failed with convergence equal 1"
## [1] "Failed with convergence equal 1"
## [1] "Succeed but with convergence equal 1"
## [1] "Failed with convergence equal 1"
## [1] "Failed with convergence equal 1"
```

```
set.seed(NULL)
```

Result: this likelihood funcion is not converging.

Part2: Regressing using the time invarient variables

```r
tiv.reg <- function (kt.bt){
    # calculate the result based on input function
    individual <- coef(lm(LOGWAGE~0+EDUC+POTEXPER+as.factor(PERSONID),kt.bt))
    individual <- individual[3:length(individual)]

    kt.bt.tiv <- kt.bt[!duplicated(kt.bt$PERSONID),] # keep one obs for each person

    # regress fixed effection on time in-varients
    kt.bt.tiv$individual <- individual
    result.3 <- lm(individual~ABILITY + MOTHERED + FATHERED + BRKNHOME + SIBLINGS, kt.bt.tiv)
    return(coef(result.3))
}


# Do ols to get individual fixed effect
tiv.reg(kt.rand)
```

```
##   (Intercept)       ABILITY      MOTHERED      FATHERED      BRKNHOME
##  0.2528095301  0.0492612922 -0.0004590623  0.0117547464 -0.1935114045
##      SIBLINGS
## -0.0216516649
```

```r
tiv.res <- data.frame(tiv.reg(kt.rand))

boot.result <- tiv.res[-1] # creating empty data frame
id100 <- unique(kt.rand$PERSONID)

for (i in 1:49){
    # sampling from existing 100 person
    smp <- sample(id100,100, replace = T)
    kt.bt <- kt.rand[0,]
    for (j in 1:100){
      kt.bt <- rbind(kt.bt, kt.rand[kt.rand$PERSONID %in% smp[j],])
    }
    # apply tiv.reg, collect result
    boot.result <- cbind(boot.result,tiv.reg(kt.bt))
}
data.frame(coefficient = tiv.res, se = apply(boot.result,1,sd)) # corrected SE
```

```
##             tiv.reg.kt.rand.         se
## (Intercept)     0.2528095301 0.65335784
## ABILITY         0.0492612922 0.05583900
## MOTHERED       -0.0004590623 0.02528241
## FATHERED        0.0117547464 0.01725471
## BRKNHOME       -0.1935114045 0.25973178
## SIBLINGS       -0.0216516649 0.02894438
```

None of these are significant

Part 3

For with-in estimator: Since there may be the issue of heteroskedasticity causing by the correlation of error term between each individual, a sandwich form (huber white) standard error is needed rathe than standard ols se.

```
X <- as.matrix(kt[,c("EDUC","POTEXPER")])
inv_XX <- solve(t(X) %*% X)
residual <- fix.wi(kt$LOGWAGE,kt[,c("EDUC","POTEXPER")],kt$PERSONID)$residuals

D <- t(X) %*% diag(residual)^2 %*% X

EHW <- inv_XX %*% D %*% inv_XX

diag(sqrt(EHW))
```

```
##         EDUC     POTEXPER
## 0.0004080221 0.0005525334
```