Homework 6 Solutions

Hansen 17.2

$\mathbb{E}[e_{it}|X_{it}] = 0$ is not a strong enough condition for $\hat{\beta}$ from a fixed effects regression to be unbiased. This condition says that e_{it} is (mean) independent of X_{it}, but it does not rule out that e_{it} could be related to, say, X_{it+1}. This is not an entirely strange case either, particularly if a good "shock" in the current period leads to the covariate changing in the next time period.

More specifically, recall that

$$\begin{aligned}\mathbb{E}[\hat{\beta} - \beta|\mathbf{X}] &= \left(\sum_{i=1}^n \dot{\mathbf{X}}_i'\dot{\mathbf{X}}_i\right)^{-1} \sum_{i=1}^n \dot{\mathbf{X}}_i'\mathbb{E}[\mathbf{e}_i|\mathbf{X}] \\ &= \left(\sum_{i=1}^n \dot{\mathbf{X}}_i'\dot{\mathbf{X}}_i\right)^{-1} \sum_{i=1}^n \dot{\mathbf{X}}_i'\mathbb{E}[\mathbf{e}_i|\mathbf{X}_i]\end{aligned}$$

where $\mathbf{X}$ is the $nT \times k$ "data matrix" and the other notation is from class, and the second equality uses that the observations are independent of each other. Notice that,

$$\mathbb{E}[\mathbf{e}_i|\mathbf{X}_i] = \begin{bmatrix}\mathbb{E}[e_{i1}|X_{i1}, X_{i2}, \ldots, X_{iT}] \\ \mathbb{E}[e_{i2}|X_{i1}, X_{i2}, \ldots, X_{iT}] \\ \vdots \\ \mathbb{E}[e_{iT}|X_{i1}, X_{i2}, \ldots, X_{iT}]\end{bmatrix}$$

The condition in the problem is not strong enough that this term is equal to 0. And, if it is some function of X, then $\hat{\beta}$ would not, in general, be unbiased for β.

Additional Question 1

Part (a)

```r
library(Matrix)
load("job_displacement_clean2.RData")
# drop already treated
data <- subset(data, first.displaced != 2001)
data <- droplevels(data)
Y <- data$learn
data$D <- 1*( (data$year >= data$first.displaced) & data$first.displaced != 0)
X <- model.matrix(~ as.factor(year) + D, data=data)
n <- length(unique(data$id))
tp <- length(unique(data$year))
iT <- matrix(rep(1,tp))
Dmat <- bdiag(replicate(n,iT,simplify=FALSE))
M <- Matrix::Diagonal(n*tp) - Dmat%*%solve(t(Dmat)%*%Dmat)%*%t(Dmat)
bet <- solve(t(X)%*%M%*%X) %*% t(X)%*%M%*%Y
bet
```

```
## 8 x 1 Matrix of class "dgeMatrix"
##              [,1]
## [1,]  5.29358972
## [2,]  0.09436799
## [3,]  0.18195412
## [4,]  0.26904810
## [5,]  0.28752717
## [6,]  0.35242722
## [7,]  0.38427488
## [8,] -0.23557976
```

Next, let's calculate the standard errors where we use that

$$\begin{aligned}\sqrt{n}(\hat{\beta} - \beta) &= \mathbb{E}[\mathbf{X}_i'\mathbf{M}_i\mathbf{X}_i]^{-1}\frac{1}{\sqrt{n}}\sum_{i=1}^{n}\mathbf{X}_i'\mathbf{M}_i\mathbf{e}_i + o_p(1)\\ &\xrightarrow{d} N(0, \mathbf{V})\end{aligned}$$

where

$$\begin{aligned}\mathbf{V} &= \mathbb{E}[\mathbf{X}_i'\mathbf{M}_i\mathbf{X}_i]^{-1}\boldsymbol{\Omega}\mathbb{E}[\mathbf{X}_i'\mathbf{M}_i\mathbf{X}_i]^{-1}\\ &= \mathbb{E}[\dot{\mathbf{X}}_i'\dot{\mathbf{X}}_i]^{-1}\boldsymbol{\Omega}\mathbb{E}[\dot{\mathbf{X}}_i'\dot{\mathbf{X}}_i]^{-1}\end{aligned}$$

and

$$\begin{aligned}\boldsymbol{\Omega} &= \mathbb{E}[\mathbf{X}_i'\mathbf{M}_i\mathbf{e}_i\mathbf{e}_i'\mathbf{M}_i\mathbf{X}_i]\\ &= \mathbb{E}[\dot{\mathbf{X}}_i'\mathbf{e}_i\mathbf{e}_i'\dot{\mathbf{X}}_i]\end{aligned}$$

It's worth thinking about how to actually estimate these because $\dot{\mathbf{X}}_i$ is a matrix rather than our usual case of it being a vector. First, notice that

$$\dot{\mathbf{X}}_i'\dot{\mathbf{X}}_i = \sum_{t=1}^{T}\dot{X}_{it}\dot{X}_{it}'$$

which is a $k \times k$ matrix. Thus, the natural estimate of $\mathbb{E}[\dot{\mathbf{X}}_i'\dot{\mathbf{X}}_i]$ is

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{t=1}^{T}\dot{X}_{it}\dot{X}_{it}' = \dot{\mathbf{X}}'\dot{\mathbf{X}}/n$$

which corresponds to exactly the same way that we estimated this type of term throughout the semester. Next,

$$\dot{\mathbf{X}}_i'\mathbf{e}_i = \sum_{t=1}^{T}X_{it}e_{it}$$

which is a $k \times 1$ vector and so that

$$\dot{\mathbf{X}}_i'\mathbf{e}_i\mathbf{e}_i'\dot{\mathbf{X}}_i = \left(\sum_{t=1}^{T}\dot{X}_{it}e_{it}\right)\left(\sum_{t=1}^{T}\dot{X}_{it}e_{it}\right)'$$

and implies that we would estimate $\boldsymbol{\Omega}$ by

$$\hat{\boldsymbol{\Omega}} = \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{t=1}^{T}\dot{X}_{it}\hat{e}_{it}\right)\left(\sum_{t=1}^{T}\dot{X}_{it}\hat{e}_{it}\right)'$$

As far as I know, you can't play the same matrix algebra "trick" that we usually use here (in particular, recall that in the cross sectional case we could estimate $\hat{\mathbf{\Omega}} = \frac{1}{n}\sum_{i=1}^n X_i X_i' \hat{e}_i^2$, but that, for programming, it was often convenient to re-express this $\hat{\mathbf{\Omega}} = \tilde{\mathbf{X}}'\tilde{\mathbf{X}}/n$ where a typical element of $\tilde{\mathbf{X}}$ is given by $X_i \hat{e}_i$.) Anyway, the line below that uses the `rowsum` function is essentially just manually calculating $\sum_{t=1}^T X_{it}\hat{e}_{it}$ and then using matrix algebra below it.

```
ehat <- as.numeric(Y - X%*%bet)
n <- length(unique(data$id))
dotX <- M%*%X
Q <- t(X) %*% dotX / n
dotXe <- rowsum(as.matrix(dotX*ehat), group=data$id)
Omeg <- t(dotXe)%*%dotXe/n

V <- solve(Q)%*%Omeg%*%solve(Q)
se <- sqrt(diag(V))/sqrt(n)
round(cbind.data.frame(bet=as.numeric(bet), se=se), 4)
```

```
##       bet     se
## 1  5.2936 0.1003
## 2  0.0944 0.0085
## 3  0.1820 0.0098
## 4  0.2690 0.0108
## 5  0.2875 0.0115
## 6  0.3524 0.0116
## 7  0.3843 0.0124
## 8 -0.2356 0.0257
```

Thus, we estimate that job displacement reduces earnings by about 23%. As a check, let's compare this to what we get from `fixest`.

```
library(fixest)
fe_reg <- feols(learn ~ as.factor(year) + D | id, data=data)
summary(fe_reg)
```

```
## OLS estimation, Dep. Var.: learn
## Observations: 18,928
## Fixed-effects: id: 2,704
## Standard-errors: Clustered (id)
##                      Estimate Std. Error  t value  Pr(>|t|)
## as.factor(year)2003  0.094368   0.008533 11.05873 < 2.2e-16 ***
## as.factor(year)2005  0.181954   0.009756 18.65073 < 2.2e-16 ***
## as.factor(year)2007  0.269048   0.010758 25.00982 < 2.2e-16 ***
## as.factor(year)2009  0.287527   0.011507 24.98624 < 2.2e-16 ***
## as.factor(year)2011  0.352427   0.011598 30.38799 < 2.2e-16 ***
## as.factor(year)2013  0.384275   0.012396 30.99875 < 2.2e-16 ***
## D                   -0.235580   0.025677 -9.17486 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 0.352029     Adj. R2: 0.75681
##                    Within R2: 0.107896
```

These appear to be the same (or the same up to possibly a degree of freedom adjustment).

Part (b)

```r
# list of time periods
# for simplicity I'm going to convert this to 1,2,3,4,5,6,7...
tlist <- (sort(unique(data$year)) - 1999)/2
# list of groups (excluding never-treated)
glist <- (sort(unique(data$first.displaced))[-1] - 1999)/2

# create new variables in updated time scale
data$G <- ifelse(data$first.displaced==0, 0, (data$first.displaced - 1999)/2)
data$tp <- (data$year-1999)/2


# write a function to compute att(g,t)
# I compute these as averages using weights, but it
# is fine to use subsets of data here too.
# @param w weights, used for bootstrap
# @param base_period, allows base period to optionally
#  be fixed at one
compute.attgt <- function(data, w=rep(1,nrow(data)),
                          use_base_period_1=FALSE) {
  # data frame to store results
  results <- list()
  counter <- 1

  for (this_t in tlist[-1]) {
    for (this_g in glist) {
      base_period <- min(this_t-1, this_g-1)
      if (use_base_period_1) base_period <- 1
      G <- 1*(data$G==this_g)
      U <- 1*(data$G==0)
      pre <- 1*(data$tp == base_period)
      post <- 1*(data$tp == this_t)
      pg <- weighted.mean(data$G == this_g, w=w)
      pu <- weighted.mean(data$G == 0, w=w)
      ppre <- mean(pre)
      ppost <- mean(post)

      this_attgt <- weighted.mean(data$learn*G*post/pg/ppost, w=w) -
                          weighted.mean(data$learn*G*pre/pg/ppre, w=w) -
        (weighted.mean(data$learn*U*post/pu/ppost, w=w) -
           weighted.mean(data$learn*U*pre/pu/ppre, w=w))
      results[[counter]] <- c(attgt=this_attgt, g=this_g, t=this_t)
      counter <- counter+1
    }
  }

  # convert to data frame
  results <- as.data.frame(do.call("rbind", results))

  results
}

results <- compute.attgt(data)
# print results
round(results[order(results$g, results$t),],4)
```

```
##      attgt g t
## 1  -0.2091 2 2
## 7  -0.1562 2 3
## 13 -0.1775 2 4
## 19 -0.2375 2 5
## 25 -0.2347 2 6
## 31 -0.2781 2 7
## 2   0.0117 3 2
## 8  -0.1138 3 3
## 14 -0.1124 3 4
## 20 -0.1677 3 5
## 26 -0.1698 3 6
## 32 -0.0313 3 7
## 3   0.0726 4 2
## 9  -0.0641 4 3
## 15 -0.1989 4 4
## 21 -0.3070 4 5
## 27 -0.2000 4 6
## 33 -0.2506 4 7
## 4  -0.0128 5 2
## 10  0.0036 5 3
## 16 -0.0603 5 4
## 22 -0.3184 5 5
## 28 -0.3115 5 6
## 34 -0.2210 5 7
## 5   0.0195 6 2
## 11 -0.1013 6 3
## 17 -0.0129 6 4
## 23  0.0565 6 5
## 29 -0.2505 6 6
## 35 -0.1633 6 7
## 6   0.1183 7 2
## 12 -0.0379 7 3
## 18 -0.0295 7 4
## 24  0.0205 7 5
## 30 -0.1143 7 6
## 36 -0.2056 7 7
```

Part (c)

```
# function to compute att0
# ret_weights argument optionally returns the underlying
# weights rather than att0
compute.att0 <- function(attgt_results, w=rep(1,nrow(data)),
                         ret_weights=FALSE) {
  # overall attgt weights
  ever_treated <- which(data$G != 0)
  w <- w[ever_treated]
  pg <- sapply(glist, function(g) weighted.mean(data[ever_treated,]$G==g, w=w))
  maxT <- max(tlist)
  w0 <- function(g,t) {
    1*(t >= g)*pg[glist==g] / (maxT - g + 1)
  }
  # add weights to results
  w0gt <- sapply(1:nrow(attgt_results),
                 function(i) w0(attgt_results$g[i], attgt_results$t[i]))
  attgt_results$w0 <- w0gt
  # optionally return computed weights
  if(ret_weights) return(attgt_results)
  att0 <- sum(attgt_results$attgt*attgt_results$w0)
  att0
}

att0 <- compute.att0(results)

# bootstrap standard errors
B <- 100
id_list <- unique(data$id)
boot_att0 <- list()
for (b in 1:B) {
  # draw weights from multinomial distribution (this is exactly the same
  # as empirical bootstrap)
  boot_weights <- as.numeric(rmultinom(n=1, size=n, prob=rep(1/n,n)))
  this_boot_weights_id <- cbind.data.frame(id=id_list, boot_weights=boot_weights)
  boot_data <- merge(data, this_boot_weights_id, by="id")
  boot_attgt <- compute.attgt(data, w=boot_data$boot_weights)
  boot_att0[[b]] <- compute.att0(boot_attgt, w=boot_data$boot_weights)
}

boot_att0 <- do.call("rbind", boot_att0)
```

```
se <- sd(boot_att0)

round(cbind.data.frame(att0=att0, se=se), 4)
```

```
##       att0     se
## 1 -0.2111 0.0257
```

Thus, we are estimating a large, negative and statistically significant effect of job displacement.

Part (d)

```
# twfe weights
Edt <- function(t) {
  mean( (t >= data$G) & (data$G !=0) )
}
mEdt <- mean(sapply(tlist, Edt))
pg2 <- sapply(glist, function(g) mean(data$G==g))
maxT <- max(tlist)
wTWFE_num <- function(g,t,post0=FALSE) {
  if ((t < g) & post0) return(0)
  hgt <- 1*(t>=g) - (maxT-g+1)/maxT - Edt(t) + mEdt
  hgt*pg2[glist==g]
}
# add weights to results
wTWFEgt_num <- sapply(1:nrow(results),
                      function(i) wTWFE_num(results$g[i],
                                            results$t[i],
                                            post0=TRUE))
wTWFEgt_den <- sapply(1:nrow(results),
                      function(i) wTWFE_num(results$g[i],
                                            results$t[i],
                                            post0=TRUE))
wTWFEgt <- wTWFEgt_num/sum(wTWFEgt_den)
results$wTWFE <- wTWFEgt

# get results from att0
results$watt0 <- compute.att0(results, ret_weights=TRUE)$w0

# print results
round(results[order(results$g, results$t),], 4)
```

```
##       attgt g t  wTWFE  watt0
## 1  -0.2091 2 2 0.0394 0.0378
## 7  -0.1562 2 3 0.0330 0.0378
## 13 -0.1775 2 4 0.0284 0.0378
## 19 -0.2375 2 5 0.0218 0.0378
## 25 -0.2347 2 6 0.0171 0.0378
## 31 -0.2781 2 7 0.0105 0.0378
## 2   0.0117 3 2 0.0000 0.0000
## 8  -0.1138 3 3 0.0465 0.0344
## 14 -0.1124 3 4 0.0430 0.0344
## 20 -0.1677 3 5 0.0380 0.0344
## 26 -0.1698 3 6 0.0344 0.0344
## 32 -0.0313 3 7 0.0294 0.0344
## 3   0.0726 4 2 0.0000 0.0000
## 9  -0.0641 4 3 0.0000 0.0000
## 15 -0.1989 4 4 0.0454 0.0303
## 21 -0.3070 4 5 0.0419 0.0303
## 27 -0.2000 4 6 0.0394 0.0303
## 33 -0.2506 4 7 0.0358 0.0303
## 4  -0.0128 5 2 0.0000 0.0000
## 10  0.0036 5 3 0.0000 0.0000
## 16 -0.0603 5 4 0.0000 0.0000
## 22 -0.3184 5 5 0.0836 0.0594
## 28 -0.3115 5 6 0.0799 0.0594
## 34 -0.2210 5 7 0.0748 0.0594
## 5   0.0195 6 2 0.0000 0.0000
## 11 -0.1013 6 3 0.0000 0.0000
## 17 -0.0129 6 4 0.0000 0.0000
## 23  0.0565 6 5 0.0000 0.0000
## 29 -0.2505 6 6 0.0718 0.0626
## 35 -0.1633 6 7 0.0682 0.0626
## 6   0.1183 7 2 0.0000 0.0000
## 12 -0.0379 7 3 0.0000 0.0000
## 18 -0.0295 7 4 0.0000 0.0000
## 24  0.0205 7 5 0.0000 0.0000
## 30 -0.1143 7 6 0.0000 0.0000
## 36 -0.2056 7 7 0.1178 0.1761
```

Notice that none of the TWFE weights are negative here though some of them do seem fairly different from the weights on ATT^O.

As a final side-comment, you might notice that

```
attTWFE <- sum(results$attgt*results$wTWFE)
attTWFE
```

```
## [1] -0.2139115
```

is not exactly equal to α that we calculated earlier. There are two reasons for this. First, our expression for α in terms of underlying $ATT(g,t)$'s relied on parallel trends actually holding; so if it does not, then we will not get exactly the same thing. Second, there is estimation error in $ATT(g,t)$; that is, we have $\widehat{ATT}(g,t)$ rather than $ATT(g,t)$, and the way to estimate this is not unique. Let me very quickly give show how you can recover α. In the notes, the line right before relating α to underlying $ATT(g,t)$'s was

$$\alpha = \sum_{t=2}^{T} \sum_{g \in \bar{\mathcal{G}}} h(g,t) \Big(\mathbb{E}[Y_{it} - Y_{i1}) | G = g] - \mathbb{E}[Y_{it} - Y_{i1}) | U = 0] \Big) p_g \Big/ \sum_{t=1}^{T} \mathbb{E}[\ddot{D}_{it}^2]$$

We can use this to compute a "decomposition" of α that will be equal to what we actually estimated.

```r
# this computes "ATT(g,t)'s" using base period = 1
# everywhere which is analogous to above equation
results2 <- compute.attgt(data, use_base_period_1=TRUE)
# compute weights but allow for non-zero weights
# in pre-treatment periods
wTWFEgt_num2 <- sapply(1:nrow(results),
                       function(i) wTWFE_num(results$g[i],
                                             results$t[i],
                                             post0=FALSE))
wTWFEgt_den2 <- sapply(1:nrow(results),
                       function(i) wTWFE_num(results$g[i],
                                             results$t[i],
                                             post0=TRUE))
wTWFEgt2 <- wTWFEgt_num2/sum(wTWFEgt_den2)
# check if this delivers alpha
sum(results2$attgt*wTWFEgt2)
```

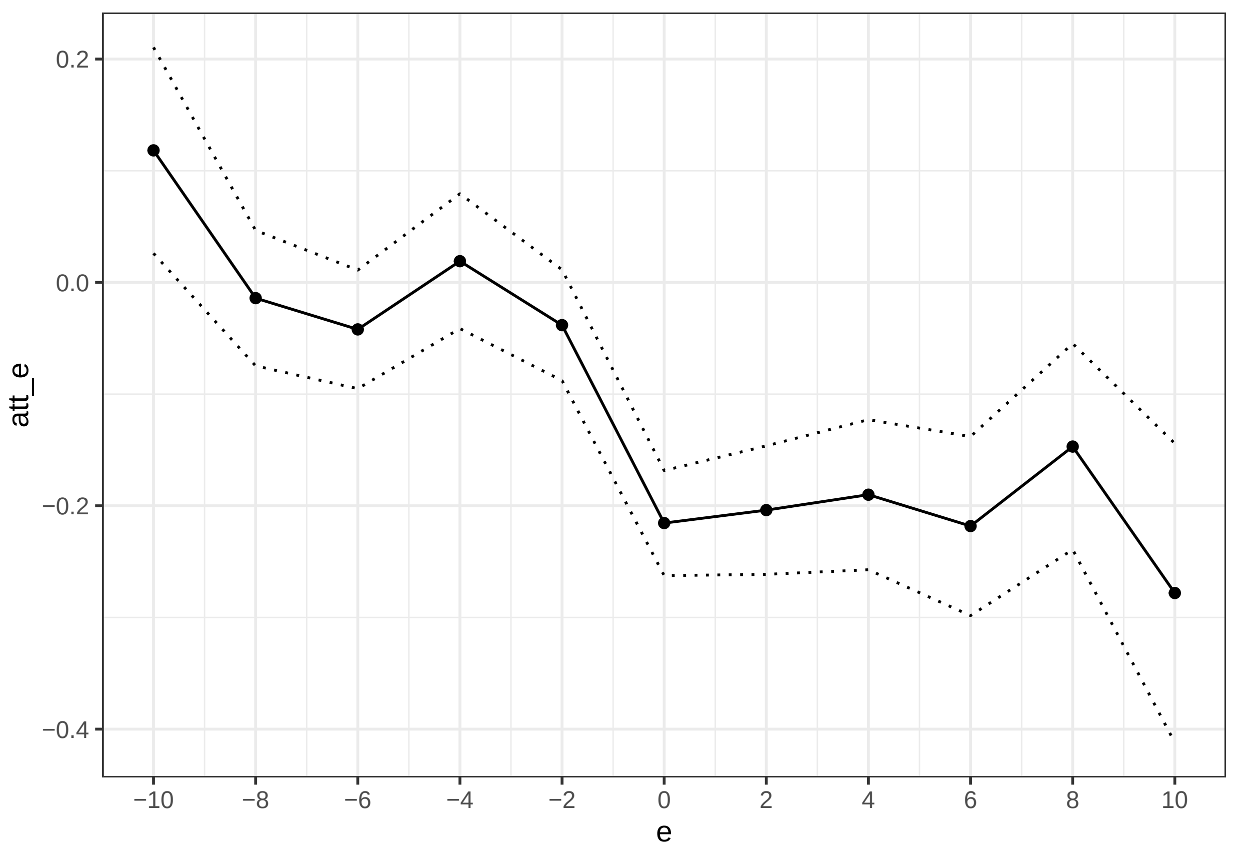
```
## [1] -0.2355798
```

which are now the same.

Part (e)

```r
# function to compute event studies
compute.es <- function(attgt_results, w=rep(1,nrow(data))) {
  # event study weights
  eseq <- sort(unique(attgt_results$t - attgt_results$g))
  es_res <- list()
  counter <- 1
  for (e in eseq) {
    this_keepers <- which( (attgt_results$t - attgt_results$g) == e)
    this_attgt <- attgt_results$attgt[this_keepers]
    pg <- sapply(attgt_results$g[this_keepers],
                 function(g)  weighted.mean(data$G==g, w=w))
    pg <- pg / sum(pg)
    att_e <- sum(this_attgt*pg)
    es_res[[counter]] <- c(att_e=att_e, e=e)
    counter <- counter+1
  }
  # convert to data frame
  es_results <- as.data.frame(do.call("rbind", es_res))
  es_results
}

es_results <- compute.es(results)

# bootstrap event study
```

```r
B <- 100
id_list <- unique(data$id)
boot_es <- list()
for (b in 1:B) {
  boot_weights <- as.numeric(rmultinom(n=1, size=n, prob=rep(1/n,n)))
  this_boot_weights_id <- cbind.data.frame(id=id_list, boot_weights=boot_weights)
  boot_data <- merge(data, this_boot_weights_id, by="id")
  boot_attgt <- compute.attgt(data, w=boot_data$boot_weights)
  boot_es[[b]] <- compute.es(boot_attgt, w=boot_data$boot_weights)$att_e
}

boot_es <- do.call("rbind", boot_es)
se <- apply(boot_es, 2, sd)
es_results$se <- se
es_results$ciL <- es_results$att_e - 1.96*es_results$se
es_results$ciU <- es_results$att_e + 1.96*es_results$se

library(ggplot2)
ggplot(data=es_results, mapping=aes(x=e,y=att_e)) +
  geom_line() +
  geom_point(size=1.5) +
  geom_line(aes(y=ciU), linetype="dotted") +
  geom_line(aes(y=ciL), linetype="dotted") +
  scale_x_continuous(breaks=seq(-5,5), labels=seq(-10,10,2)) +
  theme_bw()
```



The figure suggests that job displacement causes earnings to drop by, on average, about 20% and that this effect is quite persistent; it appears to be roughly the same 10 years following job displacement. If you look at the estimates in pre-treatment periods, with the exception of 10 years before job displacement, the estimates are fairly close to 0 (and not statistically different from 0) suggesting that the parallel trends assumption is likely to be fairly reasonable in this application.

Additional Question 2

We can rewrite the expression in the problem as

$$\hat{\beta}_{gmm} = \underset{b}{\operatorname{argmin}}\ \mathbf{Y}'\mathbf{Z}\widehat{\mathbf{W}}\mathbf{Z}'Y - 2b'\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\mathbf{Z}'\mathbf{Y} + b'\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\mathbf{Z}'\mathbf{X}b$$

Taking the first order condition, we have that

$$\begin{aligned} 0 &= -\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\mathbf{Z}'\mathbf{Y} + \mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\mathbf{Z}'\mathbf{X}\hat{\beta}_{gmm} \\ \implies \hat{\beta}_{gmm} &= (\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\mathbf{Z}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\mathbf{Z}'\mathbf{Y} \end{aligned}$$

This completes the first part of the problem. For the asymptotic distribution, notice that we can re-write the previous equation as

$$\begin{aligned}
\hat{\beta}_{gmm} &= \left(\frac{1}{n}\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\frac{1}{n}\mathbf{Z}'\mathbf{X}\right)^{-1}\frac{1}{n}\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\frac{1}{n}\sum_{i=1}^{n} Z_iY_i \\
&= \left(\frac{1}{n}\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\frac{1}{n}\mathbf{Z}'\mathbf{X}\right)^{-1}\frac{1}{n}\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\frac{1}{n}\sum_{i=1}^{n} Z_i(X_i'\beta + e_i) \\
&= \beta + \left(\frac{1}{n}\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\frac{1}{n}\mathbf{Z}'\mathbf{X}\right)^{-1}\frac{1}{n}\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\frac{1}{n}\sum_{i=1}^{n} Z_ie_i
\end{aligned}$$

This implies that

$$\begin{aligned}
\sqrt{n}(\hat{\beta}_{gmm} - \beta) &= \left(\frac{1}{n}\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\frac{1}{n}\mathbf{Z}'\mathbf{X}\right)^{-1}\frac{1}{n}\mathbf{X}'\mathbf{Z}\widehat{\mathbf{W}}\frac{1}{\sqrt{n}}\sum_{i=1}^{n} Z_ie_i \\
&= \Big(\mathbb{E}[XZ']\mathbf{W}\mathbb{E}[ZX']\Big)^{-1}\mathbb{E}[XZ']\mathbf{W}\frac{1}{\sqrt{n}}\sum_{i=1}^{n} Z_ie_i + o_p(1)
\end{aligned}$$

where the second equality holds because $\widehat{\mathbf{W}} \xrightarrow{p} \mathbf{W}$ and because $\frac{1}{n}\mathbf{X}'\mathbf{Z} = \frac{1}{n}\sum_{i=1}^{n} X_iZ_i' \xrightarrow{p} \mathbb{E}[XZ']$ and by the continuous mapping theorem. Next, notice that $\frac{1}{\sqrt{n}}\sum_{i=1}^{n} Z_ie_i \xrightarrow{d} \mathcal{N}(0, \mathbf{\Omega})$ where $\mathbf{\Omega} := \mathbb{E}[ZZ'e^2]$. Thus, by the continuous mapping theorem, we have that,

$$\sqrt{n}(\hat{\beta}_{gmm} - \beta) \xrightarrow{d} \mathcal{N}(0, \mathbf{V})$$

where

$$\mathbf{V} = \Big(\mathbb{E}[XZ']\mathbf{W}\mathbb{E}[ZX']\Big)^{-1}\mathbb{E}[XZ']\mathbf{W}\mathbf{\Omega}\mathbf{W}\mathbb{E}[ZX']\Big(\mathbb{E}[XZ']\mathbf{W}\mathbb{E}[ZX']\Big)^{-1}$$

Additional Question 3

Part (a)

Just to keep the notation simple, I am going to refer to the 2003 group as group 2, the 2005 group and group 3, and the 2007 group as group 4, and use corresponding notation for the time periods.

In this setting, we have the following available moment conditions:

$$ATT(2,2) - \Big(\mathbb{E}[\Delta Y_2|G=2] - \mathbb{E}[\Delta Y_2|U=1]\Big) = 0$$

$$\mathbb{E}[\Delta Y_2|G=3] - \mathbb{E}[\Delta Y_2|U=1] = 0$$

$$\mathbb{E}[\Delta Y_2|G=4] - \mathbb{E}[\Delta Y_2|U=1] = 0$$

$$ATT(2,3) - ATT(2,2) - \Big(\mathbb{E}[\Delta Y_3|G=2] - \mathbb{E}[\Delta Y_3|U=1]\Big) = 0$$

$$ATT(3,3) - \Big(\mathbb{E}[\Delta Y_3|G=3] - \mathbb{E}[\Delta Y_3|U=1]\Big) = 0$$

$$\mathbb{E}[\Delta Y_3|G=4] - \mathbb{E}[\Delta Y_3|U=1] = 0$$

$$ATT(2,4) - ATT(2,3) - \Big(\mathbb{E}[\Delta Y_4|G=2] - \mathbb{E}[\Delta Y_4|U=1]\Big) = 0$$

$$ATT(3,4) - ATT(3,3) - \Big(\mathbb{E}[\Delta Y_4|G=3] - \mathbb{E}[\Delta Y_4|U=1]\Big) = 0$$

$$ATT(4,4) - \Big(\mathbb{E}[\Delta Y_4|G=4] - \mathbb{E}[\Delta Y_4|U=1]\Big) = 0$$

Thus, there are 9 non-redundant moment conditions.[1] Using the notation in the problem, these

[1]Notice that, from back substituting across equations, you can see how the more complicated moment conditions lead to the correct expressions for $ATT(g,t)$. For example, by solving for $ATT(3,3)$ in the fifth equation and plugging this into $ATT(3,3)$ in the 8th equation, we have that $ATT(3,4) = \mathbb{E}[\Delta Y_4|G=3] - \mathbb{E}[\Delta Y_4|U=1] + \Big(\mathbb{E}[\Delta Y_3|G=3] - \mathbb{E}[\Delta Y_3|U=1]\Big) = \mathbb{E}[Y_4 - Y_2|G=3] - \mathbb{E}[Y_4 - Y_2|U=1]$ which is correct. Similar arguments apply for the other terms.

moment conditions can be re-written as

$$\mathbb{E}\left[\left(\frac{\mathbf{1}\{G=2\}}{p_2} - \frac{U}{p_U}\right)\Delta Y_2\right] - ATT(2,2) = 0$$

$$\mathbb{E}\left[\left(\frac{\mathbf{1}\{G=3\}}{p_3} - \frac{U}{p_U}\right)\Delta Y_2\right] = 0$$

$$\mathbb{E}\left[\left(\frac{\mathbf{1}\{G=4\}}{p_4} - \frac{U}{p_U}\right)\Delta Y_2\right] = 0$$

$$\mathbb{E}\left[\left(\frac{\mathbf{1}\{G=2\}}{p_2} - \frac{U}{p_U}\right)\Delta Y_3\right] - \Big(ATT(2,3) - ATT(2,2)\Big) = 0$$

$$\mathbb{E}\left[\left(\frac{\mathbf{1}\{G=3\}}{p_3} - \frac{U}{p_U}\right)\Delta Y_3\right] - ATT(3,3) = 0$$

$$\mathbb{E}\left[\left(\frac{\mathbf{1}\{G=4\}}{p_4} - \frac{U}{p_U}\right)\Delta Y_3\right] = 0$$

$$\mathbb{E}\left[\left(\frac{\mathbf{1}\{G=2\}}{p_2} - \frac{U}{p_U}\right)\Delta Y_4\right] - \Big(ATT(2,4) - ATT(2,3)\Big) = 0$$

$$\mathbb{E}\left[\left(\frac{\mathbf{1}\{G=3\}}{p_3} - \frac{U}{p_U}\right)\Delta Y_4\right] - \Big(ATT(3,4) - ATT(3,3)\Big) = 0$$

$$\mathbb{E}\left[\left(\frac{\mathbf{1}\{G=4\}}{p_4} - \frac{U}{p_U}\right)\Delta Y_4\right] - ATT(4,4) = 0$$

a similar argument as in the course notes, define

$$Y_i = \begin{bmatrix} \left(\frac{\mathbf{1}\{G_i=2\}}{p_2} - \frac{U_i}{p_U}\right)\Delta Y_{i2} \\ \left(\frac{\mathbf{1}\{G_i=3\}}{p_3} - \frac{U_i}{p_U}\right)\Delta Y_{i2} \\ \left(\frac{\mathbf{1}\{G_i=4\}}{p_4} - \frac{U_i}{p_U}\right)\Delta Y_{i2} \\ \left(\frac{\mathbf{1}\{G_i=2\}}{p_2} - \frac{U_i}{p_U}\right)\Delta Y_{i3} \\ \left(\frac{\mathbf{1}\{G_i=3\}}{p_3} - \frac{U_i}{p_U}\right)\Delta Y_{i3} \\ \left(\frac{\mathbf{1}\{G_i=4\}}{p_4} - \frac{U_i}{p_U}\right)\Delta Y_{i3} \\ \left(\frac{\mathbf{1}\{G_i=2\}}{p_2} - \frac{U_i}{p_U}\right)\Delta Y_{i4} \\ \left(\frac{\mathbf{1}\{G_i=3\}}{p_3} - \frac{U_i}{p_U}\right)\Delta Y_{i4} \\ \left(\frac{\mathbf{1}\{G_i=4\}}{p_4} - \frac{U_i}{p_U}\right)\Delta Y_{i4} \end{bmatrix} \qquad \mathbf{Y} = \begin{pmatrix} Y_1' \\ Y_2' \\ \vdots \\ Y_n' \end{pmatrix} \qquad \alpha = \begin{pmatrix} ATT(2,2) \\ ATT(2,3) \\ ATT(2,4) \\ ATT(3,3) \\ ATT(3,4) \\ ATT(4,4) \end{pmatrix} \qquad \mathbf{R} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

where Y_i is a 9×1 vector, $\mathbf{Y}$ is an $n \times 9$ matrix, α is an 6×1 vector, and $\mathbf{R}$ is a 9×6 matrix, and define $\mathbf{1}$ to be an $n \times 1$ vector of ones. Then, we can collect all 9 moment conditions as

$$\mathbb{E}[Y - \mathbf{R}\alpha] = 0$$

Part (b)

Notice that there are more moment conditions than there are parameters to estimate which implies that we will not be able to make the sample analogue of the moment conditions above exactly equal to 0. Instead, we can use GMM to estimate α. That is, we can estimate

$$\hat{\alpha} = \underset{a}{\text{argmin}} \left(\frac{1}{n}\mathbf{Y}'\mathbf{1} - \mathbf{R}a\right)' \widehat{\mathbf{W}} \left(\frac{1}{n}\mathbf{Y}'\mathbf{1} - \mathbf{R}a\right)$$

Taking the first order condition and solving (note that these steps are the same as are discussed in the notes, just with different particular expressions for α, $\mathbf{R}$, etc.), we have that

$$\hat{\alpha} = \left(\mathbf{R}'\widehat{\mathbf{W}}\mathbf{R}\right)^{-1}\mathbf{R}'\widehat{\mathbf{W}}\frac{1}{n}\mathbf{Y}'\mathbf{1}$$

For this part of the problem, we will take $\widehat{\mathbf{W}} = \mathbf{I}_9$ to compute α.

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'

## The following objects are masked from 'package:Matrix':
##
##     expand, pack, unpack
```

```
load("job_displacement_clean2.RData")
data <- subset(data, (year <= 2007))
data <- subset(data, first.displaced %in% c(0, 2003, 2005, 2007))

wide_data <- pivot_wider(data, id_cols=c("id","first.displaced"),
                         names_from=c("year"),
                         names_prefix="learn",
                         values_from=c("learn")) %>% as.data.frame()
head(wide_data)
```

```
##   id first.displaced learn2001 learn2003 learn2005 learn2007
## 1  6               0  10.68052  10.94200  11.08214  11.36210
## 2  8            2005  10.35774  10.43412  10.49127  10.62133
## 3  9               0  10.81978  10.91509  11.08214  10.81978
## 4 14            2007  11.09741  11.39639  11.66135  11.35041
## 5 16               0  10.57132  11.08214  11.17044  11.09741
## 6 17               0  11.25156  11.03489  10.66896  10.81978
```

```
G2003 <- 1*(wide_data$first.displaced==2003)
G2005 <- 1*(wide_data$first.displaced==2005)
G2007 <- 1*(wide_data$first.displaced==2007)
U <- 1*(wide_data$first.displaced==0)

p2003 <- mean(G2003)
p2005 <- mean(G2005)
p2007 <- mean(G2007)
```

```r
p0 <- mean(U)

Y1 <- (G2003/p2003 - U/p0)*(wide_data$learn2003 - wide_data$learn2001)
Y2 <- (G2005/p2005 - U/p0)*(wide_data$learn2003 - wide_data$learn2001)
Y3 <- (G2007/p2007 - U/p0)*(wide_data$learn2003 - wide_data$learn2001)
Y4 <- (G2003/p2003 - U/p0)*(wide_data$learn2005 - wide_data$learn2003)
Y5 <- (G2005/p2005 - U/p0)*(wide_data$learn2005 - wide_data$learn2003)
Y6 <- (G2007/p2007 - U/p0)*(wide_data$learn2005 - wide_data$learn2003)
Y7 <- (G2003/p2003 - U/p0)*(wide_data$learn2007 - wide_data$learn2005)
Y8 <- (G2005/p2005 - U/p0)*(wide_data$learn2007 - wide_data$learn2005)
Y9 <- (G2007/p2007 - U/p0)*(wide_data$learn2007 - wide_data$learn2005)

Y <- cbind(Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8, Y9)

R <- matrix(c(1,0,0,0,0,0,
              0,0,0,0,0,0,
              0,0,0,0,0,0,
              -1,1,0,0,0,0,
              0,0,0,1,0,0,
              0,0,0,0,0,0,
              0,-1,1,0,0,0,
              0,0,0,-1,1,0,
              0,0,0,0,0,1), byrow=TRUE, nrow=9)

n <- nrow(Y)
ones <- as.matrix(rep(1,n))
W <- diag(nrow=9)

alp <- as.numeric( solve(t(R)%*%W%*%R)%*%t(R)%*%W%*%t(Y)%*%ones/n )
# print the estimates
as.matrix(round(alp,5))
```

```
##          [,1]
## [1,] -0.20911
## [2,] -0.15616
## [3,] -0.17746
## [4,] -0.11380
## [5,] -0.11238
## [6,] -0.19890
```

These estimates are exactly the same as the corresponding ones that we estimated in the earlier problem. The reason for this is (i) our choice of the way to set up the non-redundant moment conditions, and (ii) the weighting matrix being the identity matrix. It implies that the expressions for each $ATT(g, t)$ are exactly the same as in the previous case. Below, you will see that, when we change the weighting matrix, we do not get exactly the same estimates.

Part (c)

To compute the efficient GMM estimates, we need to first estimate $\mathbf{\Omega} = \mathbb{E}[ee']$ where $e_i := Y_i - \mathbf{R}\alpha$ (which is a 9×1 vector here). To do this, define $\hat{\mathbf{e}} := \mathbf{Y} - \mathbf{1} \otimes \hat{\alpha}'\mathbf{R}'$, and then we can estimate $\mathbf{\Omega}$ by

$$\hat{\mathbf{\Omega}} = \frac{1}{n}\hat{\mathbf{e}}'\hat{\mathbf{e}}$$

and then compute $\hat{\alpha}^o_{gmm} = \left(\mathbf{R}'\hat{\mathbf{\Omega}}^{-1}\mathbf{R}\right)^{-1}\mathbf{R}'\hat{\mathbf{\Omega}}^{-1}\frac{1}{n}\mathbf{Y}'\mathbf{1}$. Code is below.

```
# subtracts vector from each row of Y
ehat <- sweep(Y, 2, as.numeric(R%*%alp))
Omeg <- t(ehat)%*%ehat/n

alp_gmm <- solve(t(R)%*%solve(Omeg)%*%R)%*%t(R)%*%solve(Omeg)%*%t(Y)%*%ones/n
alp_gmm <- as.numeric(alp_gmm)
round(cbind.data.frame(alp_gmm, alp), 5)
```

```
##    alp_gmm      alp
## 1 -0.21185 -0.20911
## 2 -0.15654 -0.15616
## 3 -0.17845 -0.17746
## 4 -0.11173 -0.11380
## 5 -0.11117 -0.11238
## 6 -0.17191 -0.19890
```

You can see that these estimates are similar to the previous ones, but they are not exactly the same. This is because we are putting different weights on the different moment conditions here. To get a sense of this, it is interesting to show the weighting matrix $\hat{\mathbf{\Omega}}^{-1}$

```
round(solve(Omeg),2)
```

```
##       Y1    Y2    Y3    Y4   Y5    Y6   Y7   Y8   Y9
## Y1  0.26 -0.01 -0.01  0.14 0.00  0.00 0.09 0.00 0.00
## Y2 -0.01  0.19 -0.01  0.00 0.00  0.00 0.00 0.00 0.00
## Y3 -0.01 -0.01  0.18 -0.01 0.00  0.07 0.00 0.00 0.04
## Y4  0.14  0.00 -0.01  0.21 0.00 -0.01 0.07 0.00 0.00
## Y5  0.00  0.00  0.00  0.00 0.12  0.00 0.00 0.04 0.00
## Y6  0.00  0.00  0.07 -0.01 0.00  0.19 0.00 0.00 0.01
## Y7  0.09  0.00  0.00  0.07 0.00  0.00 0.25 0.00 0.00
## Y8  0.00  0.00  0.00  0.00 0.04  0.00 0.00 0.09 0.00
## Y9  0.00  0.00  0.04  0.00 0.00  0.01 0.00 0.00 0.08
```

Part (d)

We can compute the J-test by computing

```
J <- n * t( (t(Y)%*%ones/n - R%*%alp_gmm) ) %*%
  solve(Omeg) %*% ( t(Y)%*%ones/n - R%*%alp_gmm )
J
```

```
##          [,1]
## [1,] 2.552087
```

Under $\mathbb{H}_0$ that all the moments conditions are actually equal to 0, then we know that $J \xrightarrow{d} \chi^2_{l-k}$ (here $l - k = 3$ because we 9 moment conditions and 6 parameters to estimate). We can compute a p-value for our test by

```
# p-value (prob. get test statistic at least this large if H0 true)
1-pchisq(J, 3)
```

```
##            [,1]
## [1,] 0.4659512
```

This suggests that we would fail to reject $\mathbb{H}_0$ that all the moment condition. In the particular context of this problem, it implies that we do not find violations of the parallel trends assumption across groups in their pre-treatment periods.