# Demo Class 3

This is the code to implement the work for Demo Class 3

- estimate a DiD model
- apply cluster robust standard errors

## Introduction

We are going to estimate the following two models

## Preparing your workfile

We add the basic libraries needed for this week's work:

```
library(tidyverse)    # for almost all data handling tasks
library(ggplot2)      # to produce nice graphics
library(stargazer)    # to produce nice results tables
library(haven)        # to import stata file
library(ggplot2)      # for graphs
library(AER)          # access to HS robust standard errors
library(plm)          # for panel data methods
library(sandwich)     # for cluster robust se
library(lmtest)
library(coefplot)     # to create coefficient plots
```

You should also save the separately supplied `stargazer_HC.r` file in your working directory. This will make it straightforward to estimate and compare regressions with robust standard errors. Once you have done that you should include the following line into your code which basically makes this function available to you.

```
source("stargazer_HC.r")  # includes the robust regression
```

This has worked if you can see it loaded into your environment as a function.

## Data Prep

Read the data.

```
data <- read_dta("did_4.dta")
data <- as.data.frame(data)
```

Let's look at the data file.

```
str(data)
```

```
## 'data.frame':    2052 obs. of  11 variables:
##  $ id   : num  13 13 13 13 17 17 17 17 18 18 ...
##   ..- attr(*, "label")= chr "cross-sectional identifier"
##   ..- attr(*, "format.stata")= chr "%9.0g"
##  $ year : num  2012 2013 2014 2015 2012 ...
```

```
##   ..- attr(*, "label")= chr "2012 to 2015"
##   ..- attr(*, "format.stata")= chr "%9.0g"
##  $ y    : num  63.5 60 83.9 70 53.7 ...
##   ..- attr(*, "label")= chr "outcome variable"
##   ..- attr(*, "format.stata")= chr "%9.0g"
##  $ logy : num  4.15 4.09 4.43 4.25 3.98 ...
##   ..- attr(*, "label")= chr "log(y)"
##   ..- attr(*, "format.stata")= chr "%9.0g"
##  $ w    : num  0 0 0 0 0 0 0 0 0 0 ...
##   ..- attr(*, "label")= chr "=1 if treated"
##   ..- attr(*, "format.stata")= chr "%9.0g"
##  $ x1   : num  14 14 14 14 13 13 13 13 12 12 ...
##   ..- attr(*, "label")= chr "time constant control"
##   ..- attr(*, "format.stata")= chr "%9.0g"
##  $ x2   : num  0 0 0 0 0 0 0 0 0 0 ...
##   ..- attr(*, "label")= chr "time constant control"
##   ..- attr(*, "format.stata")= chr "%9.0g"
##  $ f2014: num  0 0 1 0 0 0 1 0 0 0 ...
##   ..- attr(*, "label")= chr "=1 if year == 2014"
##   ..- attr(*, "format.stata")= chr "%9.0g"
##  $ f2015: num  0 0 0 1 0 0 0 1 0 0 ...
##   ..- attr(*, "label")= chr "=1 if year == 2015"
##   ..- attr(*, "format.stata")= chr "%9.0g"
##  $ d    : num  0 0 0 0 0 0 0 0 0 0 ...
##   ..- attr(*, "label")= chr "=1 if eventually treated"
##   ..- attr(*, "format.stata")= chr "%9.0g"
##  $ post : num  0 0 1 1 0 0 1 1 0 0 ...
##   ..- attr(*, "label")= chr "=1 if year >= 2014"
##   ..- attr(*, "format.stata")= chr "%9.0g"
```

```
names(data)
```

```
##  [1] "id"    "year"  "y"     "logy"  "w"     "x1"    "x2"    "f2014" "f2015"
## [10] "d"     "post"
```

```
summary(data)
```

```
##        id             year            y                logy       
##  Min.   :   13   Min.   :2012   Min.   :  1.145   Min.   :0.1358  
##  1st Qu.: 2306   1st Qu.:2013   1st Qu.:  5.822   1st Qu.:1.7616  
##  Median : 4633   Median :2014   Median : 11.330   Median :2.4274  
##  Mean   : 5273   Mean   :2014   Mean   : 18.875   Mean   :2.4456  
##  3rd Qu.: 8496   3rd Qu.:2014   3rd Qu.: 22.224   3rd Qu.:3.1012  
##  Max.   :12534   Max.   :2015   Max.   :183.226   Max.   :5.2107  
##        w                x1             x2            f2014          f2015     
##  Min.   :0.0000   Min.   : 3.0   Min.   :0.000   Min.   :0.00   Min.   :0.00  
##  1st Qu.:0.0000   1st Qu.:11.0   1st Qu.:0.000   1st Qu.:0.00   1st Qu.:0.00  
##  Median :0.0000   Median :12.0   Median :0.000   Median :0.00   Median :0.00  
##  Mean   :0.1306   Mean   :11.8   Mean   :0.271   Mean   :0.25   Mean   :0.25  
##  3rd Qu.:0.0000   3rd Qu.:12.0   3rd Qu.:1.000   3rd Qu.:0.25   3rd Qu.:0.25  
##  Max.   :1.0000   Max.   :16.0   Max.   :1.000   Max.   :1.00   Max.   :1.00  
##        d               post    
##  Min.   :0.0000   Min.   :0.0  
##  1st Qu.:0.0000   1st Qu.:0.0  
##  Median :0.0000   Median :0.5  
```

```
##   Mean   :0.2612   Mean   :0.5
##   3rd Qu.:1.0000   3rd Qu.:1.0
##   Max.   :1.0000   Max.   :1.0
```

We will convert some variables to factor (categorical) variables

```
data$year <- as_factor(data$year)
data$w <- as_factor(data$w)
data$d <- as_factor(data$d)
levels(data$d) <- c("control","treated")
data$id <- as_factor(data$id)
```

## Investigate the Panel Structure

Let's define the dataset as a panel dataset with `id` as the cross-sectional identifier and 1year1 as the time identifier.

```
pdata <- pdata.frame(data, index = c("id","year")) # defines the panel dimensions
```

The `plm` library we imported has a useful little function to check whether the panel is balanced.

```
is.pbalanced(pdata)
```

```
## [1] TRUE
```

This has returned `TRUE` indicating that it is indeed balanced. As there are four years of data, this means that we have 513 units of observation (4 x 513 = 2052).

Let's look again at the summary statistics for the variables `w` "treated in a particular year" and `d` "ever treated".

```
summary(data[,c("w","d")])
```

```
##   w              d
##   0:1784   control:1516
##   1: 268   treated: 536
```

You can see that 536 observations belong to individuals ever treated. As we have four years of observations for each individual this implies that $S_1 = 134$ individuals were ever treated. The remainder, $S_0 = 379$ is the size of the control group. The number of observations in treatment are only 268. Exactly half. this is best understood if we look at the data for one of the observations in the treatment group (`id=3591`):

```
pdata[data$id==3591,c("id","year","w","d","post")]
```

```
##             id year w       d post
## 3591-2012 3591 2012 0 treated    0
## 3591-2013 3591 2013 0 treated    0
## 3591-2014 3591 2014 1 treated    1
## 3591-2015 3591 2015 1 treated    1
```

You can see that this individual was treated in two of the years (2014 and 2015). This is the same for all treated individuals. The variable `w` is therefore the equivalent to the "TREATxPOST" or here `d*post` variable.

## Estimate the TWFE model

Let us estimate the TWFE model but when we output the result we shall only show the coefficient to `w`, our policy estimate.

```r
mod1 <- lm(logy~id+year+w, data = pdata)
stargazer(mod1, keep = "w", type="text", digits = 6)
```

```
##
## ===================================================
##                          Dependent variable:
##                      -----------------------------
##                                 logy
## ---------------------------------------------------
## w1                           0.185928***
##                              (0.020057)
##
## ---------------------------------------------------
## Observations                    2,052
## R2                            0.968607
## Adjusted R2                   0.958053
## Residual Std. Error     0.199563 (df = 1535)
## F Statistic         91.783900*** (df = 516; 1535)
## ===================================================
## Note:                 *p<0.1; **p<0.05; ***p<0.01
```

If you want to estimate cluster (here by `id`) robust standard errors we use the following function

```r
mod1_cr_se <- sqrt(diag(vcovCL(mod1, cluster = ~ id)))
stargazer(mod1, keep = "w", type="text", se=list(mod1_cr_se), digits = 6)
```

```
##
## ===================================================
##                          Dependent variable:
##                      -----------------------------
##                                 logy
## ---------------------------------------------------
## w1                           0.185928***
##                              (0.021322)
##
## ---------------------------------------------------
## Observations                    2,052
## R2                            0.968607
## Adjusted R2                   0.958053
## Residual Std. Error     0.199563 (df = 1535)
## F Statistic         91.783900*** (df = 516; 1535)
## ===================================================
## Note:                 *p<0.1; **p<0.05; ***p<0.01
```

You can see that there is a difference in the standard error.

Let us now replace the `id`-level fixed effect by merely adding the ever treated dummy `d`

```r
mod2 <- lm(logy~year+d+w, data = pdata)
mod2_cr_se <- sqrt(diag(vcovCL(mod2, cluster = ~ id)))
stargazer(mod2, type="text", se=list(mod2_cr_se), digits = 6)
```

```
##
## ===============================================
##                         Dependent variable:
##                      ---------------------------
##                                logy
```

```
## -----------------------------------------------
## year2013                      0.018548
##                              (0.012448)
##
## year2014                     0.054975***
##                              (0.013848)
##
## year2015                    -0.037914***
##                              (0.013230)
##
## dtreated                    -0.344730***
##                              (0.090445)
##
## w1                           0.185928***
##                              (0.018469)
##
## Constant                     2.502501***
##                              (0.051841)
##
## -----------------------------------------------
## Observations                    2,052
## R2                            0.016432
## Adjusted R2                   0.014028
## Residual Std. Error     0.967528 (df = 2046)
## F Statistic         6.836235*** (df = 5; 2046)
## ===============================================
## Note:               *p<0.1; **p<0.05; ***p<0.01
```

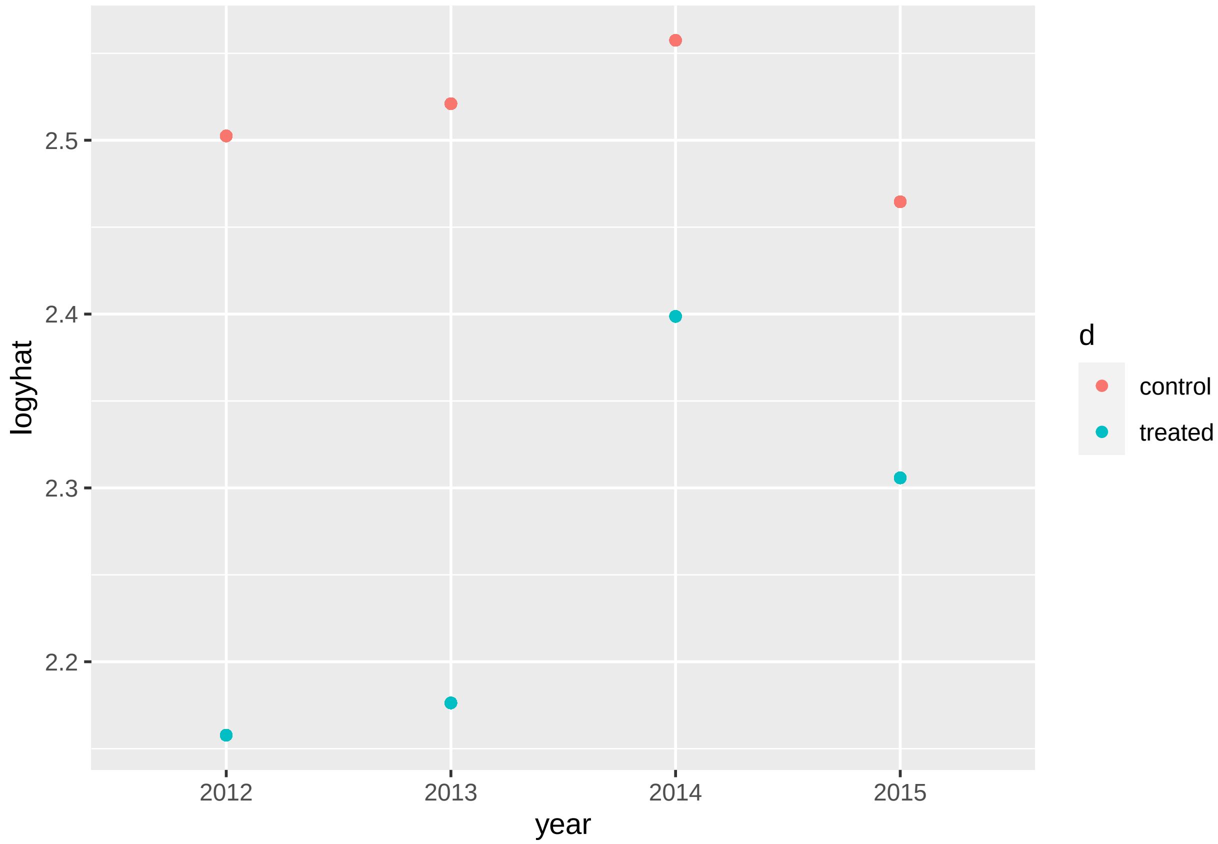
As this model only estimates 6 parameters we can actually look at all estimated coefficients. The standard errors are incorrect as we are actually estimating $S + T - 1 + 1 = 517$ coefficients. The correct standard errors are the ones from `mod1`.

From the last model we can get the fitted values.

```
pdata$logyhat <- mod2$fitted.values
```

Let us plot the predicted `logyhat`, separate for the treatment and control group. We use the second version, as it basically averages across individuals in year/treatment groups.

```
p1 <- ggplot(pdata,aes(x=year,y=logyhat,color=d)) + geom_point()
p1
```




## TWFE -> Event Study

Now we create interactions between the ever treated variable `d` and the years. In order to nderstand what the following regression does we will actually calculate new variables into the dataset.

```
pdata <- pdata %>%  mutate(d2013 = (year=="2013")*(d=="treated"),
                           d2014 = (year=="2014")*(d=="treated"),
                           d2015 = (year=="2015")*(d=="treated"))
```

Now we estimate the extended TWFE model. First with the individual fixed effects included, producing the correct standard errors.

```
mod3 <- lm(logy~id+year+d2013+d2014+d2015, data = pdata)
mod3_cr_se <- sqrt(diag(vcovCL(mod3, cluster = ~ id)))
coef_keep = c("d2013","d2014","d2015")
stargazer(mod3, type="text", keep = coef_keep, se=list(mod3_cr_se), digits = 6)
```

```
##
## ================================================
##                          Dependent variable:
##                     ----------------------------
##                                 logy
## ------------------------------------------------
## id2014                       -1.662853***
##                               (0.000000)
##
## d2013                          -0.009554
```

```
##                                 (0.031916)
##
## d2014                           0.184897***
##                                 (0.032225)
##
## d2015                           0.177406***
##                                 (0.031101)
##
## ---------------------------------------------------
## Observations                       2,052
## R2                               0.968610
## Adjusted R2                      0.958004
## Residual Std. Error      0.199681 (df = 1533)
## F Statistic         91.321650*** (df = 518; 1533)
## ===================================================
## Note:                   *p<0.1; **p<0.05; ***p<0.01
```

Now using the algebraic trick (but incorrect standard errors)

```
mod4 <- lm(logy~year+d+d2013+d2014+d2015, data = pdata)
mod4_cr_se <- sqrt(diag(vcovCL(mod4, cluster = ~ id)))
stargazer(mod4, type="text", se=list(mod4_cr_se), digits = 6)
```

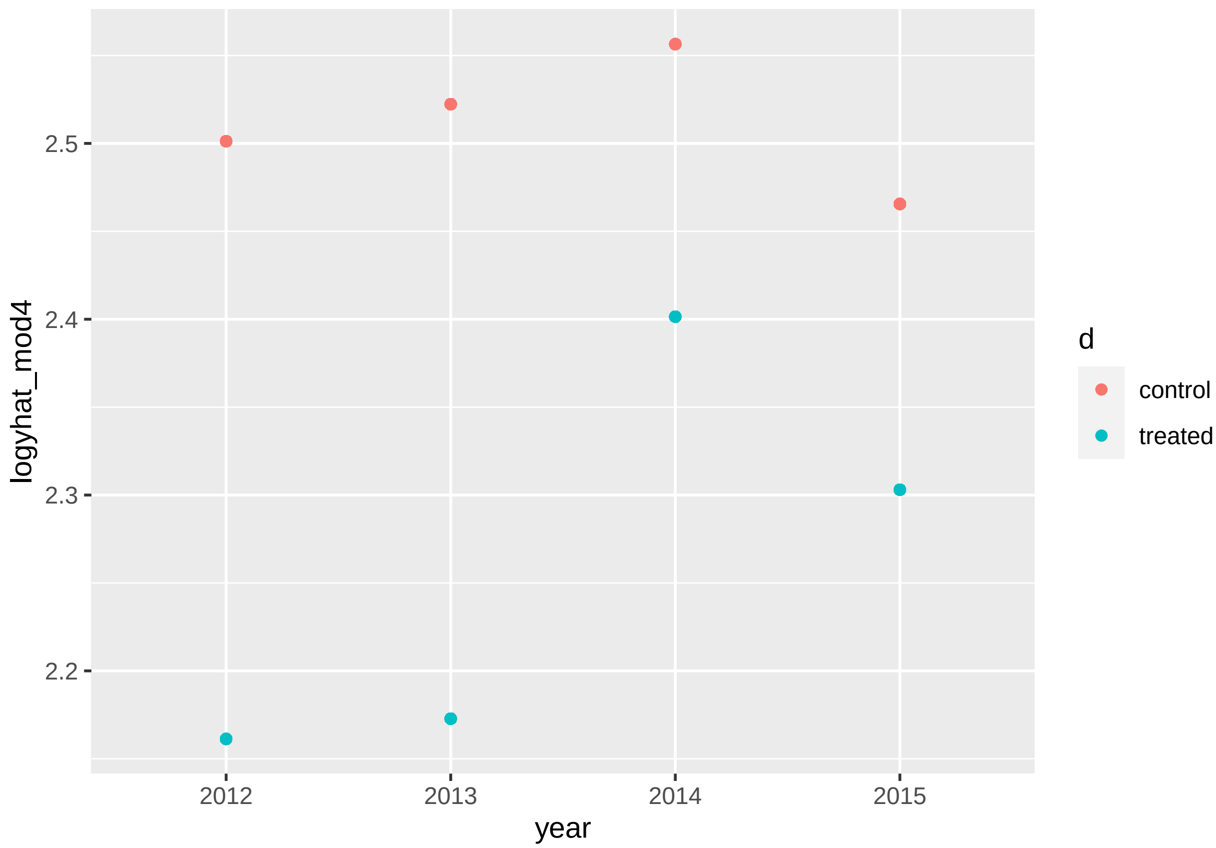
```
##
## ================================================
##                         Dependent variable:
##                     ----------------------------
##                                logy
## ------------------------------------------------
## year2013                     0.021044
##                             (0.014682)
##
## year2014                    0.055244***
##                             (0.014952)
##
## year2015                    -0.035688**
##                             (0.014154)
##
## dtreated                   -0.339953***
##                             (0.091424)
##
## d2013                        -0.009554
##                             (0.027640)
##
## d2014                       0.184897***
##                             (0.027908)
##
## d2015                       0.177406***
##                             (0.026934)
##
## Constant                    2.501253***
##                             (0.051993)
##
## ------------------------------------------------
## Observations                   2,052
```

```
## R2                            0.016436
## Adjusted R2                   0.013067
## Residual Std. Error     0.967999 (df = 2044)
## F Statistic          4.879383*** (df = 7; 2044)
## ===============================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

```
pdata$logyhat_mod4 <- mod4$fitted.values
```

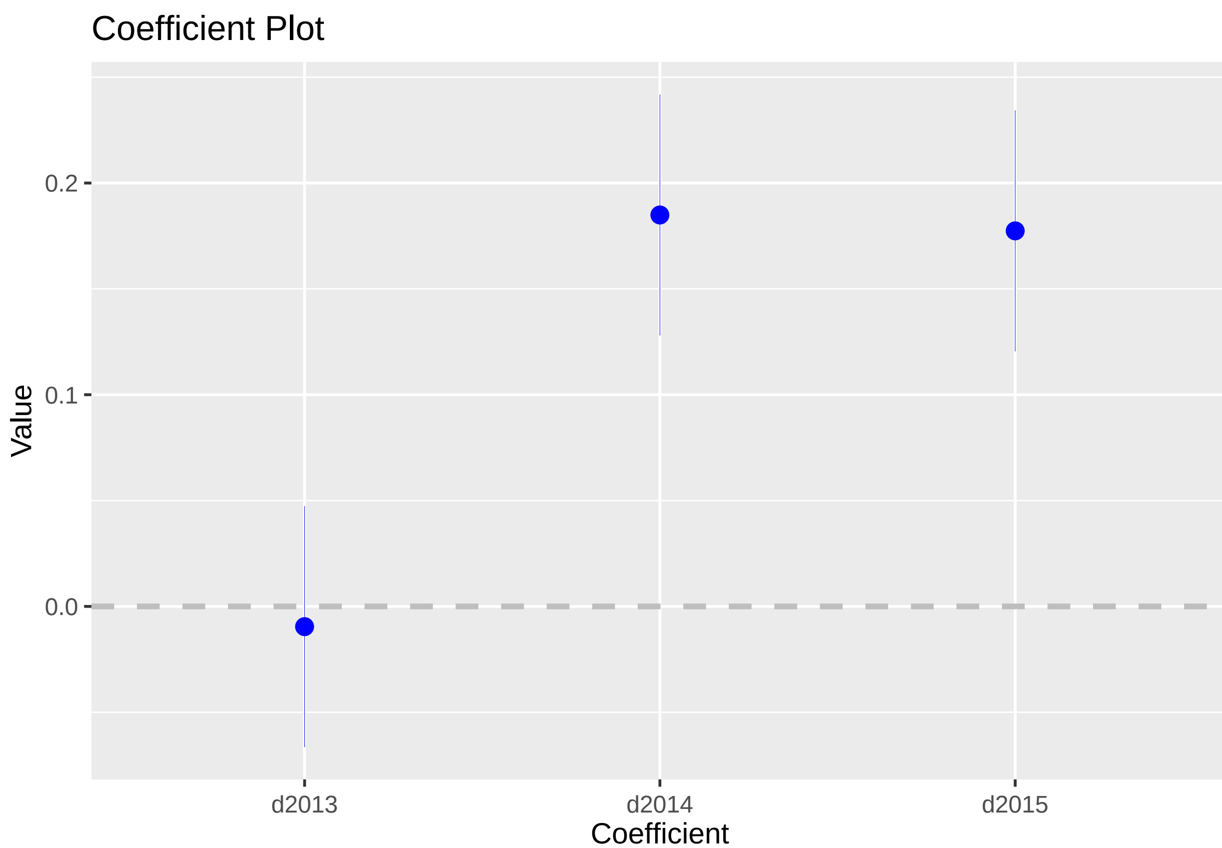
Let us plot the predicted `logyhat`, separate for the treatment and control group.

```
p2 <- ggplot(pdata,aes(x=year,y=logyhat_mod4,color=d)) + geom_point()
p2
```




The most common way to display these results is by showing the coefficients of the `d` variable interacted with the years.

```
coefplot(mod3, coefficients = coef_keep, innerCI = 0, horizontal = TRUE)
```




### Collapse the data to group means

data_collapse <- data %>% group_by(d,post)