Introducción a métodos econométricos en R

27 de septiembre de 2021

INSTITUTO TECNOLOGICO AUTONOMO DE MEXICO (ITAM)

Seminario de Investigación Económica

Instructor: Horacio Larreguy

Asistente: Manuel Quintero

Índice

Para este tutorial tenemos una base de datos tipo panel que tiene unidades, tiempo, un **tratamiento continuo** y una variable que llamamos *outcome*, véase la Tabla 1.

Tabla 1: Ejemplo de datos

unidades	tiempo	tratamiento	outcome
1	2010	0	.0064436723
1	2011	0	.51859677
1	2012	0	.3410252
1	2013	0	.88488311
1	2014	.13906856	2.954581
1	2015	.25217873	4.2451215
1	2016	.64983302	3.6849329
1	2017	.76195776	3.9613857
1	2018	1	4.1536264
1	2019	1	3.6901627
		...	
10	2010	0	.42301515
10	2011	0	.83902609
10	2012	0	.070191339
10	2013	0	.19813846
10	2014	0	.91105127
10	2015	0	.21894622
10	2016	.29448077	2.9036815
10	2017	.55710214	3.8275743
10	2018	1	4.1079011
10	2019	1	3.7057323

Veremos como correr un análisis de diferencias en diferencias y hacer dos pruebas del supuesto de tendencias paralelas: regresión *leads and lags* y un *event study*, tanto en

Stata como en R.

1. Stata

Primero leemos nuestro archivo de datos

```
use "DID_data.dta"
```

1.1. Diferencias en diferencias

Para correr una regresión en Stata que incluya múltiples efectos fijos y cluster standar errores utilizamos el paquete **reghdfe**

```
ssc install reghdfe, replace
reghdfe outcome tratamiento, a(unidades tiempo) cluster(unidades)
```

1.2. Leads y Lags

Para hacer nuestro análisis de *leads and lags* es importante reconocer si tenemos un grupo de control y uno de tratamiento o solo un grupo de tratamiento.

Lo primero es generar una variable que indique a que grupo de tratamiento pertenece cada unidad, esta variable sirve como auxiliar para reemplazar ciertos valores, como valores faltantes (NA) del grupo de control por 0.

```
* Creamos variable que especifica el grupo: tratamiento o control
gsort unidades -tratamiento
bysort unidades: gen treat = 1 if tratamiento[1] > 0
replace treat = 0 if treat == .
* Regresamos a la forma original de la base de datos
sort unidades tiempo
```

Luego, creamos las variables adelantadas y rezagadas de nuestro variable continua de tratamiento y reemplazamos los valores faltantes del grupo de control por 0.

```
* Generamos Leads y lags
* Llenamos las observacioens de control con 00
bysort unidades: gen lead3 = tratamiento[_n+3]
replace lead3 = 0 if treat == 0
bysort unidades: gen lead2 = tratamiento[_n+2]
replace lead2 = 0 if treat == 0
bysort unidades: gen lead1 = tratamiento[_n+1]
replace lead1 = 0 if treat == 0
bysort unidades: gen lag1 = tratamiento[_n-1]
replace lag1 = 0 if treat == 0
bysort unidades: gen lag2 = tratamiento[_n-2]
replace lag2 = 0 if treat == 0
bysort unidades: gen lag3 = tratamiento[_n-3]
replace lag3 = 0 if treat == 0
```

Corremos la especificación de leads y lags, con la variable de tratamiento continua, efectos fijos por unidades y tiempo, y por último, hacemos obtenemos errores estándar cluster por unidades.

```
*** Regresión leads y lags
reghdfe outcome lead* tratamiento lag*, a(unidades tiempo)
   cluster(unidades)
```

Después exportamos los que se presentan en la Tabla 2 para exportarlos a LaTeX.

```
* Exportamos resultados a LaTeX
sum outcome
return list
estadd local Cluster "unidades" // Var aux
```

```
estadd scalar Min = r(min) // valor min
estadd scalar Max = r(max) // valor max
estadd scalar Count = r(N) // obs
estadd scalar Mean = r(mean) // media
estadd scalar SD = r(sd) // std. dev.
est sto est1

label variable lead3   "Tratamiento Lead 3"
label variable lead2   "Tratamiento Lead 2"
label variable lead1   "Tratamiento Lead 1"
label variable tratamiento  "Tratamiento"
label variable outcome   "Outcome"

* Tabla de leads and lags, omitimos lags
esttab est1 using "./table_leadsLags_Stata.tex", replace noomitted
   nobaselevels label se r2 title("Leads and lags") keep(lead3 lead2
   lead1 tratamiento) scalars("DF" "Cluster") stats(N r2 Mean SD Min
   Max Cluster, label(N \(R^{2}\) "Outcome mean" "Outcome std. Dev."
   "Outcome min" "Outcome max" "Cluster")) star(* 0.10 ** 0.05 ***
   0.01) notes addnotes("Lag variables are included but not shown.")
   nonumbers

estimates clear // Borramos las regresiones guardadas
```

Tabla 2: Leads and lags

	Outcome
Tratamiento Lead 3	0.789
	(0.640)
Tratamiento Lead 2	0.345
	(0.447)
Tratamiento Lead 1	0.224
	(0.420)
Tratamiento	2.746***
	(0.402)
N	140
R^2	0.907
Outcome mean	1.362
Outcome std. Dev.	1.474
Outcome min	0.00644
Outcome max	4.491
Cluster	unidades

Standard errors in parentheses

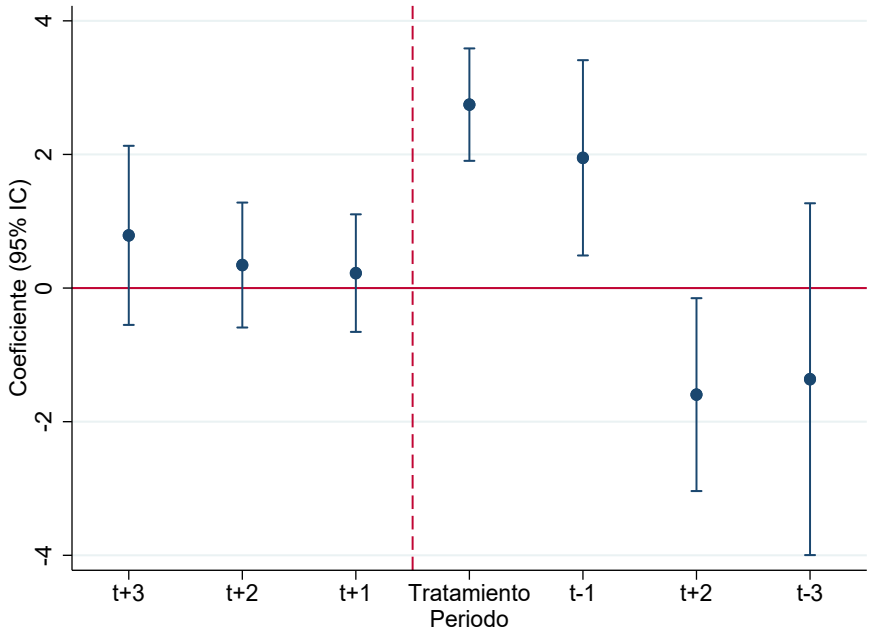
Lag variables are included but not shown.

* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

Una vez obtenidos los resultados hacemos una gráfica de coeficientes, como la que se muestra en la Gráfica 1 para ver de manera visual si se incumple el supuesto de tendencias paralelas.

```
* Gráfica de Leads and lags
ssc install coefplot, replace

coefplot, vertical drop(_cons) yline(0) coeflabels(lead3 = "t+3"
   lead2 = "t+2" lead1 = "t+1" post_treatment = "t" lag1 = "t-1" lag2
   = "t+2" lag3 = "t-3") xline(3.5, lp(dash)) ciopts(recast(rcap))
   xtitle(Periodo) ytitle(Coeficiente (95% IC))
   graphregion(color(white))

* Guardamos la gráfica en formato pdf
graph export "LeadsLags_Stata.pdf", replace
```

Figura 1: Gráfica de leads y lags



1.3. Gráfica DID (*Event study*)

Para realizar una gráfica de un *event study* necesitamos crear 3 variables distintas: **(1) post**, **(2) tiempo del inicio del evento de tratamiento** y **(3) Tiempo que indica la posición del periodo respecto al inicio tratamiento**.

La principal es tener claro que existe una variable **post** tratamiento que toma los valores de $\{0, 1\}$. Por ejemplo, en un caso donde la variable de tratamiento toma solo valores $\{0, 1\}$. Es claro que podemos tomar post, como una variable dicótoma que toma el valor de 0 antes del tratamiento y el valor de 1 después del tratamiento. Sin embargo, en el caso continuo también se puede obtener una variable post. En nuestro caso, tomaremos:

$$Post = \begin{cases} 0 & \text{if tratamiento} = 0 \\ 1 & \text{if tratamiento} > 0. \end{cases}$$

```
* Generamos una variable post:
bysort unidades: gen post = 1 if tratamiento > 0
replace post = 0 if post == .
```

Una vez que contamos con la variable post creamos una variable que indica el **tiempo del inicio del evento de tratamiento**, es decir, cuando $post = 1$ por primera vez.

```
bysort unidades: gen event_time = tiempo if post == 1
sort unidades event_time
bysort unidades: replace event_time = event_time[1]
sort unidades tiempo
replace event_time = 0 if treat == 0
```

Posteriormente, generamos una variable que nos indica el **(3) tiempo al tratamiento** (**time_to_event o tte**), es decir, cuantos periodos faltan para que inicie el tratamiento o han pasado desde que inicio el tratamiento.

```
gen time_to_event = tiempo - event_time // Creamos variable tiempo al
   tratamiento
replace time_to_event = 0 if treat == 0 // Cambiamos valores NA a 0
   del grupo control
```

Por último, una variable auxiliar, es una transformación de nuestra variable tte que nos da el tiempo al evento de tratamiento pero considerando solo valores positivos: **shifted_tte o tte**, debido a que Stata toma solo factores positivos. Esencialmente, estamos haciendo un mapeo de de la variable tiempo al evento de tratamiento a una variable que es $stt = tte - \text{mín}(ttt)$. Y creamos una variable local que específica la categoría base que usaremos en nuestra regresión.

```
summ time_to_event
g shifted_tte = time_to_event - r(min)


* Creamos variable local para usar como base: tte = 0
summ shifted_tte if time_to_event == -1
local true_neg1 = r(mean)
```

Una vez creadas estas variables, corremos una regresión especificando la categoría base de la variable , con efectos fijos por unidad y tiempo y con errores estándar cluster por municipio.

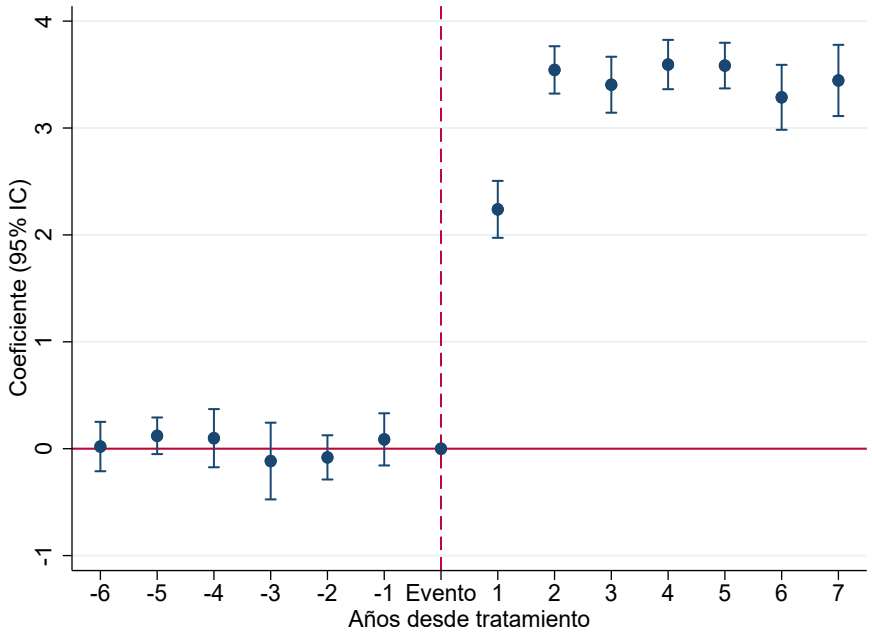
```
* Corremos la regresión especificando la categoria base, FE y cluster
reghdfe outcome ib`true_neg1'.shifted_tte, a(unidades tiempo)
   vce(cluster unidades)
```

Una vez obtenidos los resultados de la regresión pasamos a ver gráficamente si se incumple o no el supuesto de tendencias paralelas.

```
* Gráficamos el event study
coefplot, keep(*.shifted_tte) vertical  base ///
rename(0.shifted_tte="-6" 1.shifted_tte="-5" 2.shifted_tte="-4"
```

```
    3.shifted_tte="-3" 4.shifted_tte="-2" 5.shifted_tte="-1"
    6.shifted_tte="Evento" 7.shifted_tte = "1" 8.shifted_tte = "2"
    9.shifted_tte = "3" 10.shifted_tte = "4" 11.shifted_tte = "5"
    12.shifted_tte = "6" 13.shifted_tte = "7") ///
yline(0) xline(7, lp(dash)) ciopts(recast(rcap)) xtitle(Años desde
    tratamiento) ytitle(Coeficiente (95% IC)) graphregion(color(white))

* Guardamos la gráfica en formato pdf
graph export "DIDplot_Stata.pdf", replace
```

Figura 2: Gráfica DID



2. R

Primero leemos cargamos las librerías que vamos a utilizar:

```
list.of.packages <- c("stargazer", "ggplot2", "dplyr", "plm",
    "haven", "fixest", "varhandle", "lfe")


# Verificamos aquellos paquetes que no han sido instalados
new.packages <- list.of.packages[!(list.of.packages %in%
    installed.packages()[,"Package"])]


# Los instalamos y cargamos todos a la vez
if(length(new.packages)) install.packages(new.packages)
lapply(list.of.packages, library, character.only = TRUE)
```

Leemos la base de datos que utilizaremos

```
# Leemos datos
datos <- read_dta('DID_data.dta')
```

2.1. Diferencias en diferencias

En esta ocasión vamos a utilizar un paquete que nos produce los mismos errores estándar que en Stata: la función **feols** del paquete fixest. Es importante notar que el paquete que exporta resultados a LaTeX de una manera sencilla, **Stargazer**, no soporta los objetos generaros por el paquete fixest. Sin embargo, para estos casos utilizaremos la función **felm** del paquete lfe y los errores estándar de la función **feols** (Note que podríamos quedarnos con los resultados de felm(), pero para efectos comparativos con Stata hacemos uso de ambas funciones).

```
# La función feglm del paquete fixest tiene los mismos cluster S.E.
    que en STATA.
```

```
reg_did <- feols(outcome ~ tratamiento | unidades + tiempo, cluster =
   "unidades", datos)
```

2.2. Leads y Lags

Para hacer nuestro análisis de *leads and lags* es importante reconocer si tenemos un grupo de control y uno de tratamiento o solo un grupo de tratamiento.

Lo primero es generar una variable que indique a que grupo de tratamiento pertenece cada unidad, esta variable sirve como auxiliar para reemplazar ciertos valores, como valores faltantes (NA) del grupo de control por 0.

```
# Creamos indicadora de grupo de tratamiento y control
datos <- datos %>% group_by(unidades) %>% mutate(treat =
   ifelse(any(tratamiento != 0) , 1,0))
```

Luego, creamos las variables adelantadas y rezagadas de nuestro variable continua de tratamiento y reemplazamos los valores faltantes del grupo de control por 0.

```
# Creamos múltiples leads and lags
dd <- pdata.frame(datos,index = c("unidades","tiempo")) # creamos
   formato pdata de plm, por facilidad
dd <- cbind(dd, plm::lag(dd$tratamiento, c(-3:3)[-4])) # creamos lags
   and leads
dd <- as.data.frame(transform(dd)) # reveritmos el formato de pdata
   al formato original
names(dd)[(length(dd) - 5):length(dd)] <- c("tratamiento_Lead_3",
   "tratamiento_Lead_2",
                                              "tratamiento_Lead_1",
                                                 "tratamiento_Lag_1",
```

```
                                                "tratamiento_Lag_2",
                                                   "tratamiento_Lag_3") #
                                                   cambiamos los nombres
                                                   de las variables


# Revertimos los factores creados por plm
dd <- unfactor(dd)

# Cambiamos los valores creados del del grupo de control a 0
dd[dd$treat == 0, c("tratamiento_Lead_3", "tratamiento_Lead_2",
                    "tratamiento_Lead_1", "tratamiento_Lag_1",
                    "tratamiento_Lag_2", "tratamiento_Lag_3")] <- 0
```

Corremos la especificación de leads y lags, con la variable de tratamiento continua, efectos fijos por unidades y tiempo, y por último, hacemos obtenemos errores estándar cluster por unidades.

```
# Regresión Leads and Lags
reg1 <- felm(outcome ~ tratamiento_Lead_3 + tratamiento_Lead_2 +
   tratamiento_Lead_1 + tratamiento +
        tratamiento_Lag_1 + tratamiento_Lag_2 + tratamiento_Lag_3 |
           unidades + tiempo | 0 | unidades, dd)

# Regresión auxiliar para tener mismos s.e. as in STATA
reg1_aux <- feols(outcome ~ tratamiento_Lead_3 + tratamiento_Lead_2 +
   tratamiento_Lead_1 + tratamiento + tratamiento_Lag_1 +
   tratamiento_Lag_2 + tratamiento_Lag_3 | unidades + tiempo, cluster
   = ~ unidades, dd)
```

Después exportamos los que se presentan en la Tabla 3 para exportarlos a LaTeX.

```
table_leads_lags <- stargazer(reg1, header = FALSE, font.size =
   "footnotesize", dep.var.caption = "", se = list(se(reg1_aux)),
   label = "tab:tablaR", dep.var.labels.include = FALSE,
   table.placement = "H", omit = c("Constant", "year", "unidades",
   "Lag"), column.labels = "Outcome", covariate.labels =
   c("Tratamiento Lead 3", "Tratamiento Lead 2", "Tratamiento Lead
   1", "Tratamiento"), omit.stat = c("f", "ser","adj.rsq"), add.lines
   = list(c("Outcome mean", round(mean(dd$outcome),3)), c("Outcome
   std. Dev.", round(sd(dd$outcome),3)), c("Outcome min",
   round(min(dd$outcome),3)), c("Outcome max",
   round(max(dd$outcome),3)), c("Cluster", "unidades")), title =
   "Leads and lags", type = "latex")

note.latex <- "\\multicolumn{2}{l} {\\parbox[t]{6cm}{ \\textit{Notas:}
Lag variables are included but not shown. * denota p$<$0.1, ** denota
   p$<$0.05, y *** denota p$<$0.01.}} \\\\"
table_leads_lags[grepl("Note", table_leads_lags)] <- note.latex

cat(table_leads_lags, file = 'table_leadsLags_R.tex')
```

Tabla 3: Leads and lags

	Outcome
Tratamiento Lead 3	0.789
	(0.640)
Tratamiento Lead 2	0.345
	(0.447)
Tratamiento Lead 1	0.224
	(0.420)
Tratamiento	2.746***
	(0.402)
Outcome mean	1.362
Outcome std. Dev.	1.474
Outcome min	0.006
Outcome max	4.491
Cluster	unidades
Observations	140
R^2	0.907

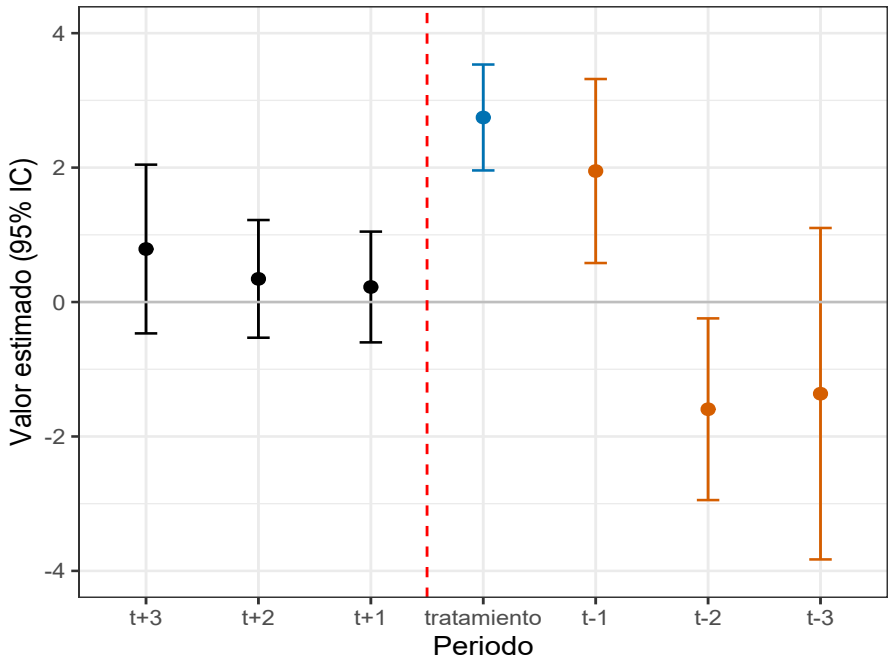
Notas: Lag variables are included but not shown. * denota p<0.1, ** denota p<0.05, y *** denota p<0.01.

Una vez obtenidos los resultados hacemos una gráfica de coeficientes, como la que se muestra en la Gráfica 3 para ver de manera visual si se incumple el supuesto de tendencias paralelas.

```
# Concatenamos las variables

datos <- data.frame(coeficientes=coeficientes, ses = se, time = time,
```

```
   type = rep(1:3, c(3,1,3)))
datos$time <- factor(datos$time, levels = c(3,2,1,0,-1,-2,-3))

# Grafica de leads and lags
colors <- c("#000000", "#0072B2","#D55E00")
leads_lags <- ggplot(data = datos, mapping = aes(y = coeficientes, x
   = time)) +
  geom_point(aes(colour = factor(type)), size = 2) +
  geom_errorbar(aes(ymin=(coeficientes-1.96*ses),
     ymax=(coeficientes+1.96*ses), colour = factor(type)), width=0.2)
     +
  ylim(c(-4,4)) +
  geom_hline(yintercept = 0, linetype="solid", color ="grey", 2) +
  geom_vline(xintercept = 3.5,linetype="dashed", color ="red", 2) +
  theme_bw() +
  ylab("Valor estimado (95% IC)") +
  xlab("Periodo") +
  scale_x_discrete(labels = c("t+3", "t+2", "t+1", "tratamiento",
     "t-1", "t-2", "t-3"), breaks = 3:-3) +
  scale_color_manual(name = "Periodo", values= colors) +
  theme(legend.position = "none")

# GUardamos figura en formato pdf
ggsave(filename = 'leads_lags_R.pdf', device = cairo_pdf, dpi = 300,
   width = 12, height = 10, units = 'cm')
```

Figura 3: Gráfica de leads y lags



2.3. Gráfica DID (*Event study*)

Para realizar una gráfica de un *event study* necesitamos crear 3 variables distintas: **(1) post**, **(2) tiempo del inicio del evento de tratamiento** y **(3) Tiempo que indica la posición del periodo respecto al inicio tratamiento**.

La principal es tener claro que existe una variable **post** tratamiento que toma los valores de $\{0, 1\}$. Por ejemplo, en un caso donde la variable de tratamiento toma solo valores $\{0, 1\}$. Es claro que podemos tomar post, como una variable dicótoma que toma el valor de 0 antes del tratamiento y el valor de 1 después del tratamiento. Sin embargo, en el caso continuo también se puede obtener una variable post. En nuestro caso, tomaremos:

$$Post = \begin{cases} 0 & \text{if tratamiento} = 0 \\ 1 & \text{if tratamiento} > 0. \end{cases}$$

```
# Variable post
dd <- dd %>% group_by(unidades) %>% mutate(post = ifelse(tratamiento
   > 0 , 1, 0))
```

Una vez que contamos con la variable post creamos una variable que indica el **tiempo del inicio del evento de tratamiento**, es decir, cuando $post = 1$ por primera vez.

```
# Variable post
dd <- dd %>% group_by(unidades) %>% mutate(event_time = tiempo[post >
   0][1])
# Grupo de control a 0
dd$event_time[dd$treat == 0] <- 0
```

Posteriormente, generamos una variable que nos indica el **(3) tiempo al tratamiento** (**time_to_event o tte**), es decir, cuantos periodos faltan para que inicie el tratamiento o han pasado desde que inicio el tratamiento.

```
# Creamos la variable tte
dd <- dd %>% group_by(unidades) %>% mutate(time_to_event =
   ifelse(treat == 1, tiempo - event_time, 0))
```

```
# Creamos stte
dd$shifted_tte <- dd$time_to_event - min(dd$time_to_event)


# Valor de categoría base
reference <- mean(dd$shifted_tte[dd$time_to_event == -1])
```

Por último, una variable auxiliar, es una transformación de nuestra variable tte que nos da el tiempo al evento de tratamiento pero considerando solo valores positivos: **shifted_tte o tte**, para tomar solo factores positivos. Esencialmente, estamos haciendo un mapeo de de la variable tiempo al evento de tratamiento a una variable que es $stt = tte - \text{mín}(ttt)$. Y creamos una variable local que específica la categoría base que usaremos en nuestra regresión.

```
# Creamos stte
dd$shifted_tte <- dd$time_to_event - min(dd$time_to_event)


# Valor de categoría base
reference <- mean(dd$shifted_tte[dd$time_to_event == -1])
```

Una vez creadas estas variables, corremos una regresión especificando la categoría base de la variable , con efectos fijos por unidad y tiempo y con errores estándar cluster por municipio.

```
# Corremos la regresión del event study
dd_plot_reg <- feols(outcome ~ i(shifted_tte, treat, ref = reference)
   | unidades + tiempo, cluster = "unidades", data = dd)
```

El paquete fixest incluye una función para hacer la gráfica DID:

```
# Plot rápido usando fixest
iplot(dd_plot_reg,
      xlab = 'Time to treatment',
      main = '')
```

Pero podemos personalizar aún más está gráfica construyendo nuestro propio código:

```
# Plot personalizado
coefs = c(dd_plot_reg$coefficients[1:6], 0,
   dd_plot_reg$coefficients[7:13])
se = c(dd_plot_reg$se[1:6], 0, dd_plot_reg$se[7:13] )


# Concatenamos las variables
datos_did <- data.frame(coeficientes = coefs, ses = se, time = -6:7,
   type = rep(1:3, c(6,1,7)))
datos_did$time <- factor(datos_did$time)
```

```
# Grafica de leads and lags
colors <- c("#000000", "#0072B2","#D55E00")
DID_plot <- ggplot(data = datos_did, mapping = aes(y = coeficientes,
   x = time)) +
  geom_point(aes(colour = factor(type)), size = 2) +
  geom_errorbar(aes(ymin=(coeficientes-1.96*ses),
     ymax=(coeficientes+1.96*ses), colour = factor(type)), width=0.2)
     +
  geom_hline(yintercept = 0, linetype="solid", color ="grey", 2) +
  geom_vline(xintercept = 7,linetype="dashed", color ="red", 2) +
  theme_bw() +
  ylab("Valor estimado (95% IC)") +
  xlab("Años desde tratamiento") +
  scale_color_manual(name = "Periodo", values= colors) +
  theme(legend.position = "none")

# Guardamos la gráfica como pdf
ggsave(filename = 'DID_plot_R.pdf', device = cairo_pdf, dpi = 300,
   width = 12, height = 10, units = 'cm')
```

Figura 4: Gráfica DID