Introducción a Métodos Econométricos en R

Professor: Horacio Larreguy
TA: Eduardo Zago

ITAM Investigación Aplicada 1 / Microeconometría Aplicada,
30/08/2023

General Perspective

Difference-in-Difference

- Datos Panel
- Tratamiento Dicotómico
  - TWFE
  - Event Study
  - Leads y Lags
- Tratamiento Continuo
  - Event Study

¿Qué son los datos panel?

- Para realizar un análisis de DiD, se necesita cierta estructura en nuestra base de datos.
- Los **datos panel** combinan las dimensiones de corte transversal y serie de tiempo.
- Las unidades pueden ser observadas repetidamente a través de distintos periodos en el tiempo.
- Estas unidades pueden ser individuos, estados, países y para los proyectos que ustedes verán regularmente serán municipios.

¿Qué son los datos panel?

- Se pueden definir de la siguiente manera:

$$\{x_{it}: \quad i = 1, 2, ...n, \quad t = 1, 2, ..., T\}$$

Table: Datos Panel Balanceados

unidades	tiempo	Y	X
i_1	t_1	1.3	.0064436723
i_1	t_2	1.2	.51859677
i_1	t_3	3	.3410252
i_2	t_1	1.25	.88488311
i_2	t_2	1.32	2.954581
i_2	t_3	4	4.2451215
		...	
i_n	t_1	0.94	3.49329
i_n	t_2	1.1	3.8499
i_n	t_3	4	2.629

Balanceados vs. No-Balanceados

- Los paneles balanceados tienen los mismos periodos de tiempo para cada unidad. Una base no balanceada se vería así:

Table: Datos Panel No-Balanceados

unidades	tiempo	Y	X
i_1	t_1	1.3	.0064436723
i_1	t_2	1.2	.51859677
i_1	t_3	3	.3410252
i_1	t_4	1.25	.88488311
i_2	t_1	1.32	2.954581
i_2	t_2	4	4.2451215
		...	
i_{n-1}	t_1	0.94	3.49329
i_{n-1}	t_2	1.1	3.8499
i_n	t_1	4	2.629

Tratamiento Dicotómico

- Empezamos el tutorial observando un ejemplo sencillo, con tratamiento dicotómico y además, todas las unidades del grupo de tratamiento seran tratadas al mismo tiempo.
- Para eso es necesario hacer dos cosas: (1) cargar los paquetes necesarios y (2) generar una base de datos.

```
library(tidyverse)
library(plm)
library(haven)
library(fixest)
library(lfe)
```

Simulación de la Base

- Con el objetivo de encontrar los resultados desados (en terminos didácticos), generaremos una base de datos a partir de la siguiente simulación:

```
set.seed(32) # Semilla aleatoria

# Vector de errores que diferenciara entre individuos sin
# generarnos errores estandar muy amplios.
error <- runif(80, min=0, max=1)

# Variable de Salario
control <- rep(floor(runif(10, min=10, max=20)), 8)

# Variable de tiempo (T=10)
time <- rep(c(1:10), 8)

# Variable que distingue entre tratamiento y control (n=8)
treatment <- c(rep(0,10), rep(1,10), rep(0,10), rep(1,10),
               rep(0,10), rep(1,10), rep(0,10), rep(1,10))

# Generamos la variable para distinguir entre individuos
unidades <- c(rep(1, 10), rep(2, 10), rep(3, 10), rep(4, 10),
              rep(5, 10), rep(6, 10), rep(7, 10), rep(8, 10))

# Variable indicadora despues de tratamiento
post <- rep(c(rep(0,5), rep(1,5)) , 8)
post_pretrend <- rep(c(rep(0,4), rep(1,6)), 8)
```

Simulación de la Base

- Necesitamos ahora generar nuestras variables dependientes de salario.
- Noten la diferencia entre la variable con **tendencias paralelas** y la que tiene **tendencias diferenciales**.

```
# Tendencias paralelas
wage <- control + 2.3 * post_pretrend * treatment +
        2 * post_pretrend * treatment * (time-6) + error

# Tendencias diferenciales
wage_pretrend <- control + 3 * post_pretrend * treatment +
            2 * post_pretrend * treatment * (time-6) + error

dd_data <- cbind.data.frame(unidades, time, post, treatment, wage, wage_pretrend)
```

Datos Panel Simulados

Figure:

unidades	time	post	treatment	wage	wage_pretrend
1	1	0	0	15.50584	15.50584
1	2	0	0	15.59481	15.59481
1	3	0	0	15.80875	15.80875
1	4	0	0	10.72882	10.72882
1	5	0	0	11.15199	11.15199
1	6	1	0	15.95619	15.95619
1	7	1	0	14.75354	14.75354
1	8	1	0	18.85206	18.85206
1	9	1	0	11.67344	11.67344
1	10	1	0	18.38713	18.38713
2	1	0	1	15.65800	15.65800
2	2	0	1	15.32137	15.32137
2	3	0	1	15.61208	15.61208
2	4	0	1	10.77264	10.77264
2	5	0	1	11.56621	12.26621
2	6	1	1	18.04880	18.74880
2	7	1	1	19.09945	19.79945
2	8	1	1	24.71947	25.41947
2	9	1	1	19.92221	20.62221
2	10	1	1	29.20759	29.90759

Análisis Gráfico

- Cuando trabajamos con datos en el tiempo regularmente es útil graficar la tendencia de las variables.
- En este caso sera util ver la tendencia del grupo de control y el de tratamiento.
- Comenzamos modificando la base:

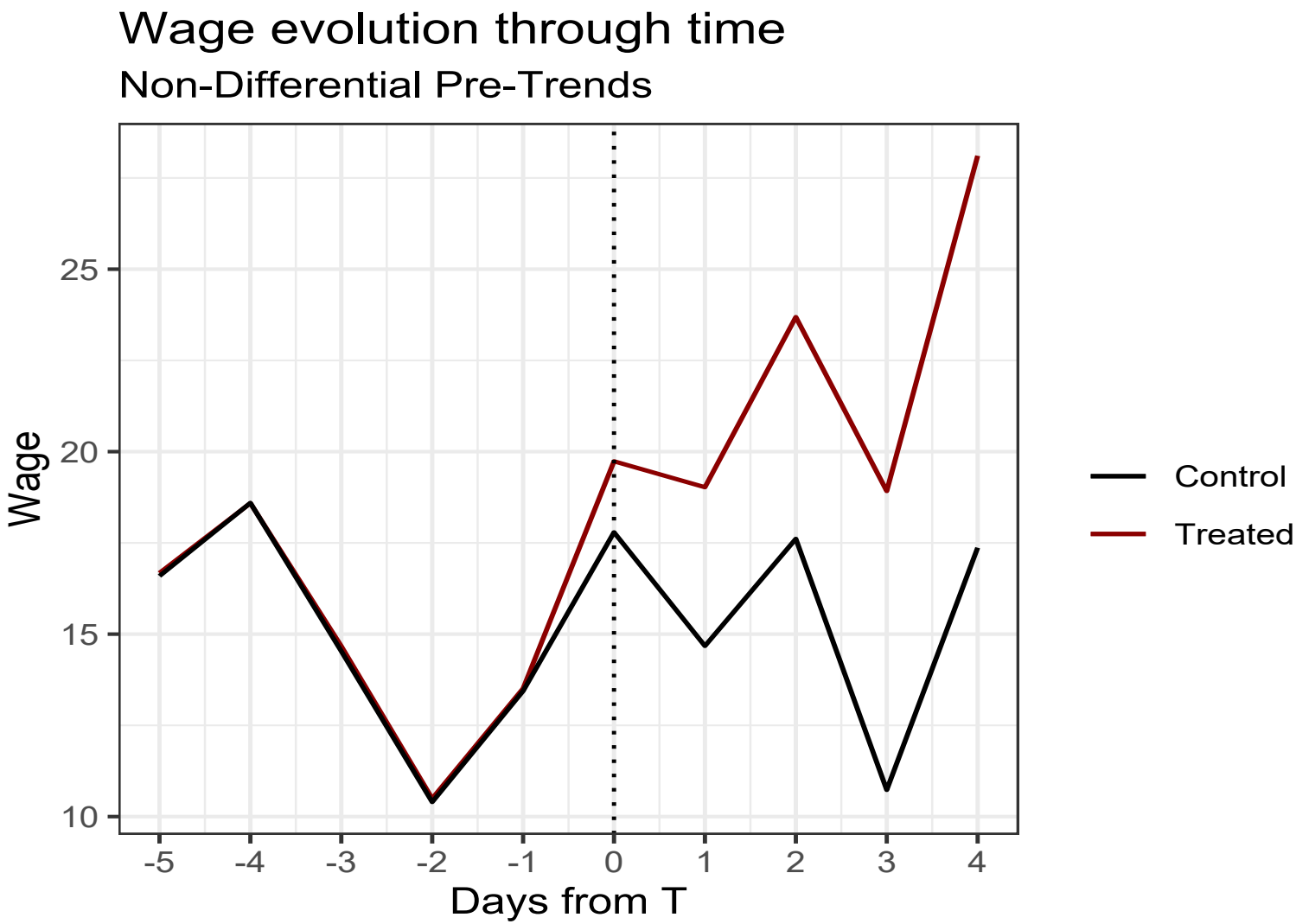
```
# Generamos el eje X
dd_data <- dd_data %>% mutate(time_dd = time - 6)

# 2. Agregamos las variables usando la media para tener una observación para cada periodo/grupo

dd_graph <- dd_data %>% group_by(treatment, time_dd) %>%
            summarise(wage_p_mean = mean(wage_pretrend),
                      wage_mean = mean(wage)) %>% ungroup()

# 3. Separamos en dos datasets para graficar:

graph_treat <- dd_graph %>% filter(treatment == 1)
graph_control <- dd_graph %>% filter(treatment == 0)
```

Análisis Gráfico

- Hacemos el line plot para la variable con tendencias paralelas:

```
stat2 <- ggplot(graph_treat, aes(time_dd, wage_mean)) +
    geom_line(aes(color="Treated"), size = .6) +
    geom_line(data = graph_control, aes(color = "Control"),
    size = .6) +
    labs(color="", x = "Days from T", y = "Wage") +
    theme_bw() +
    scale_color_manual(values=c("black", "darkred")) +
    geom_vline(xintercept=0, color="black",
        linetype="dotted") +
    scale_x_continuous(breaks=seq(-5,4,1)) +
    ggtitle("Wage evolution through time",
        subtitle = "Non-Differential Pre-Trends")
stat2
```

Análisis Gráfico

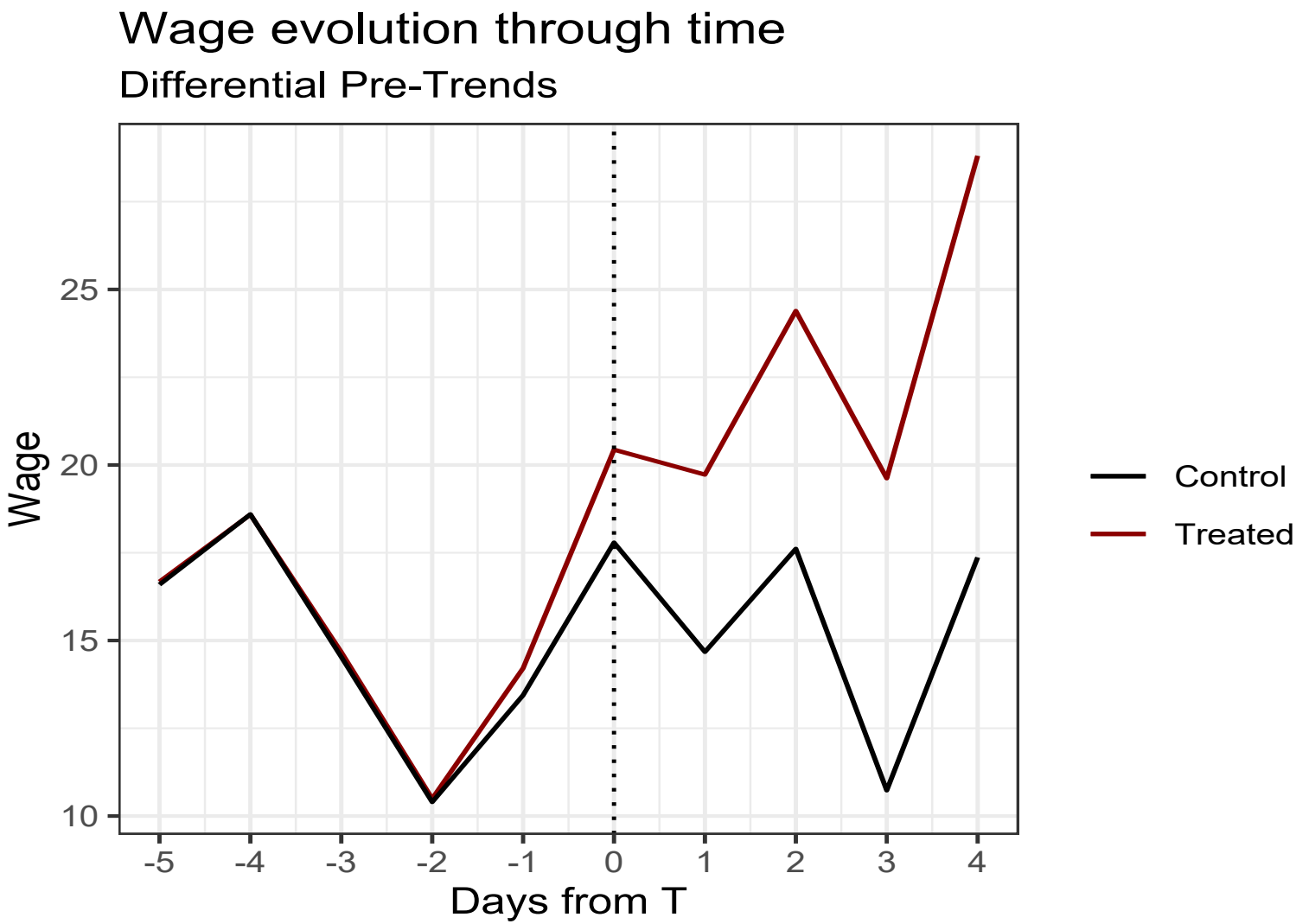


Análisis Gráfico

- Ahora para la variable sin tendencias paralelas:

```
stat <- ggplot(graph_treat, aes(time_dd, wage_p_mean)) +
    geom_line(aes(color="Treated"), size = .6) +
    geom_line(data = graph_control, aes(color = "Control"),
    size = .6) +
    labs(color="", x = "Days from T", y = "Wage") +
    theme_bw()+
    scale_color_manual(values=c("black", "darkred")) +
    geom_vline(xintercept=0, color="black", linetype="dotted") +
    scale_x_continuous(breaks=seq(-5,4,1)) +
    ggtitle("Wage evolution through time",
    subtitle = "Differential Pre-Trends")
stat
```

Análisis Gráfico



DiD Canónico

- Como visto en clase, nos interesa estimar este valor:

$$\tau = (\hat{Y}_t^1 - \hat{Y}_{t-1}^1) - (\hat{Y}_t^0 - \hat{Y}_{t-1}^0)$$

- Donde $\hat{Y}$ son medias de la variable dependiente en el tiempo t (antes y después del tratamiento) para $i = \{0, 1\}$.
- Para hacer esto hay dos formas, la antigua (canónica) y la nueva (TWFE).
- La especificación para la primera:

$$Y_{it} = \alpha + \beta Treat_i + \gamma Post_t + \tau Treat_i \cdot Post_t + \epsilon_{it}$$

```
# Interacción entre treatment y post
dd_data <- dd_data %>% mutate(treat_post = treatment*post)

reg1 <- lfe::felm(y ~ treatment + post + treat_post |
        0 | 0 | unidades, data = dd)
```

TWFE

- La forma en la que nosotros estimaremos a lo largo del curso estas regresiones es utilizando el modelo de *Two Way Fixed Effects*, el cual esta especificado de la siguiente forma:

$$Y_{it} = \alpha + \gamma_i + \lambda_t + \tau Treat_i \cdot Post_t + \epsilon_{it}$$

- Donde γ_i son *unit fixed effects* y λ_t son *time fixed effects*.
- En R:

```
reg2 <- lfe::felm(y ~ treat_post | unidades + time | 0 |
                  unidades, data = dd)
```

TWFE vs. Canónico

- Y observamos los resultados (solo para el caso donde se cumple el supuesto de tendencias paralelas):

	Dependent variable:	
	Canonic	TWFE
	(1)	(2)
DiD Estimate	6.185***	6.185***
	(0.067)	(0.066)
Treatment	0.076	
	(0.075)	
Post	2.123***	
	(0.051)	
Unit Fixed Effects	No	Yes
Time Fixed Effects	No	Yes
Observations	80	80
R^2	0.571	0.941

Notes: Standard errors are clustered at the unit level. * denotes p<0.1, ** denotes p<0.05, and *** denotes p<0.01.

TWFE vs. Canónico

- Lo primero que resalta es la diferencia en errores estándar entre especificaciones, siendo la de TWFE más eficiente (de menor varianza).
- A su vez, la R^2 es mayor, por lo que TWFE explica mejor la **variabilidad** en el outcome de interés.
- Con TWFE se controla por un mayor número de características constantes en el tiempo y entre unidades, que tan solo agregando las dummies.
- Veamos que pasa si agregamos efectos fijos y dummies de tiempo y tratamiento.

TWFE vs. Canónico

- Observamos que como las variables de tratamiento y tiempo son colineares a los efectos fijos, R las dropea (el modelo técnicamente esta mal especificado)

```
reg3 <- lfe::felm(wage ~ treat_post + treatment + post |
                    unidades + time, data = dd_data)

Call:
lfe::felm(formula = wage ~ treat_post + treatment + post | unidades +
            time, data = dd_data)

Residuals:
    Min      1Q  Median      3Q     Max
-2.64764 -0.37221  0.00372  0.32630  2.40302

Coefficients:
          Estimate Std. Error t value Pr(>|t|)
treat_post   6.1850     0.5573    11.1 2.29e-16 ***
treatment       NaN         NA     NaN      NaN
post            NaN         NA     NaN      NaN
```

Supuestos: Tendencias Paralelas y No Anticipación

- Como visto en clase, estos resultados no nos dicen nada sobre la magnitud y dirección del efecto causal si no se cumplen los supuestos de **tendencias paralelas** y **no anticipación**.
- Para verificar que se cumplan podemos realizar dos pruebas:
    1. **Event Study Design:** efectos dinámicos del tratamiento
    2. **Leads and Lags:** efecto de adelantar el tratamiento t periodos

Event Study

- El *Event Study Design* nos proporcionara los efectos dinámicos del tratamiento.
- La idea básica es interactuar las dummies de tiempo (λ_t), con la de tratamiento. La especificación es la siguiente:

$$Y_{it} = \alpha + \gamma_i + \lambda_t + \tau Treat_{it} + \sum_{t'=-T}^{0} \delta_{t'} \lambda_{t'} \cdot Treat_i + \sum_{t'=1}^{T} \delta_{t'} \lambda_{t'} \cdot Treat_i + \epsilon_{it}$$

- Nos interesa que los coeficientes de $-T$ a 0 sean estadísticamente iguales a 0.
- Si alguno de los coeficientes es distinto de 0, existe evidencia en favor de que no se cumple el supuesto de tendencias paralelas.
- Excluimos el periodo previo al tratamiento, que será nuestro periodo de referencia.

Event Study

- Modificamos la base de datos para poder realizarlo.
- No necesitamos generar las dummies de tiempo, solo generar una variable que nos diga la distancia de cada observación al periodo de tratamiento (si hay)

```
 dd_data <- dd_data %>% group_by(unidades) %>%
                        mutate(event_time = time[post > 0][1])

# Grupo de control a 0
dd_data$event_time[dd_data$treatment == 0] <- 0

# Creamos la variable tiempo a tratamiento
dd_data <- dd_data %>% group_by(unidades) %>%
                mutate(time_to_event = ifelse(treatment == 1,
                        time - event_time, 0))
```

Event Study

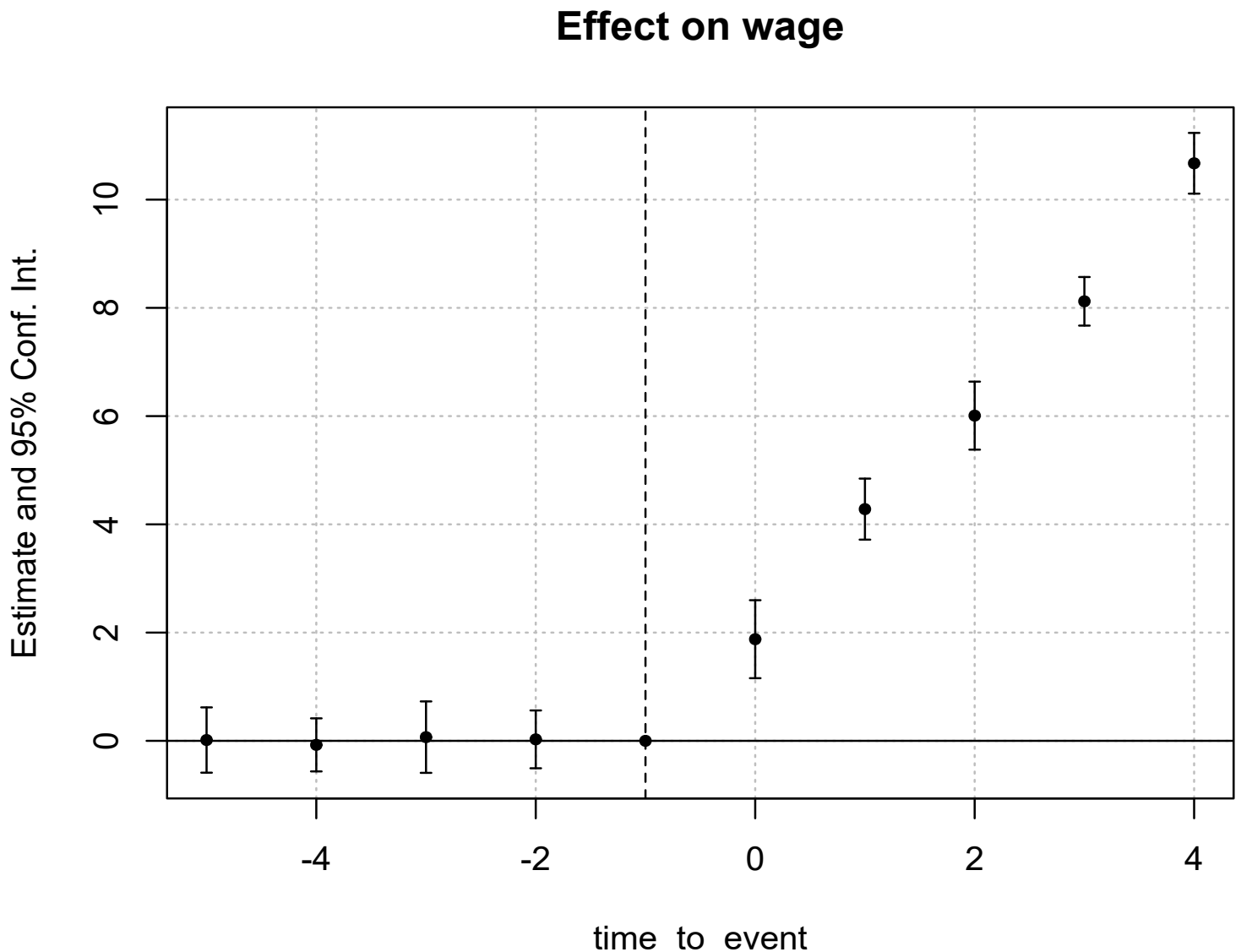
- Corremos la regresión usando el paquete **fixest**, dada la practicidad de la función i().
- Noten que la sintáxis es casi idéntica a la de felm().

```
dd_plot_reg <- feols(wage ~ i(time_to_event, treatment, ref = -1) |
                           unidades + time, cluster = "unidades",
                           data = dd_data)
```

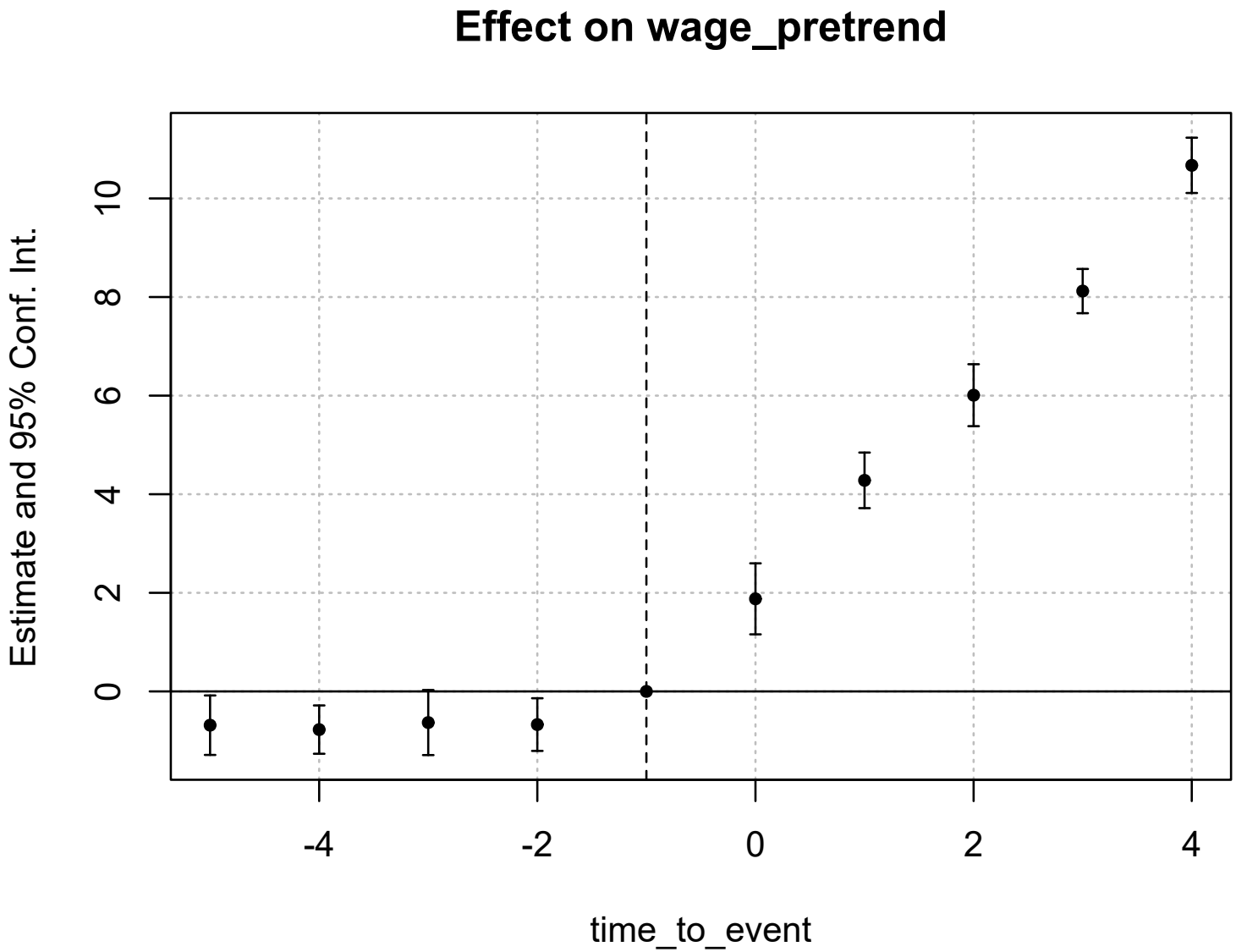
- Los resultados del Event Study siempre se presentan en gráfica, por lo que podemos usar **iplot()**.

```
iplot(dd_plot_reg)
```

Event Study: Parallel Trends



Event Study: Pre-Trends



Event Study

- El problema con iplot() es que no nos da mucha flexibilidad para manipular la gráfica (solo cambiar títulos, ejes, y así).
- Se vuelve entonces importante saber interactuar con los objetos que salen tanto de feols() como de felm():

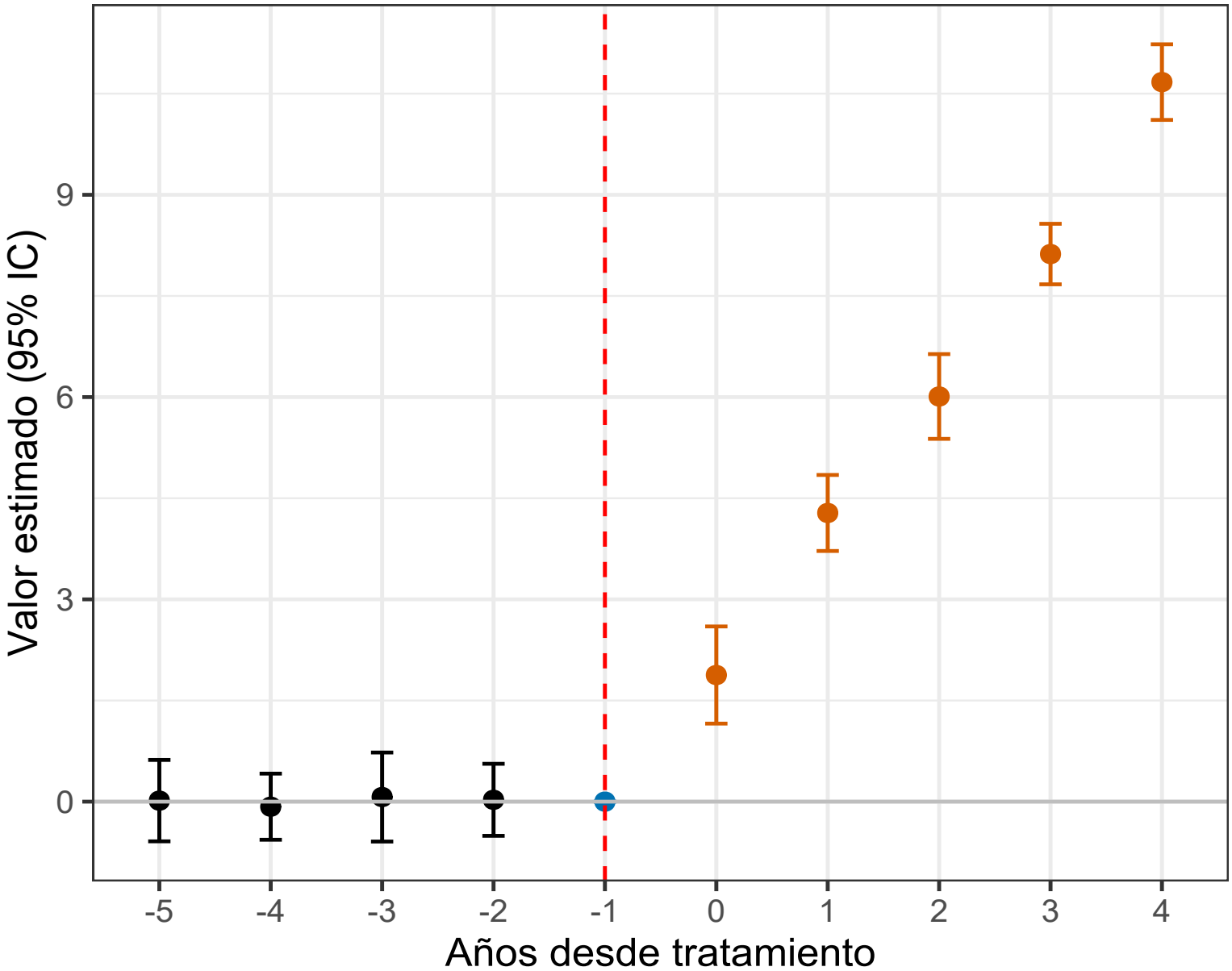
```
coefs = c(dd_plot_reg$coefficients[1:4], 0,
        dd_plot_reg$coefficients[(5):length(dd_plot_reg$coefficients)])

se = c(dd_plot_reg$se[1:4], 0,
        dd_plot_reg$se[(5):length(dd_plot_reg$se)])

# Concatenamos las variables en un df
datos_did <- data.frame(coeficientes = coefs, ses = se,
                time = -5:4, type = rep(1:3, c(4,1,5)))

datos_did$time <- factor(datos_did$time)
```

Event Study

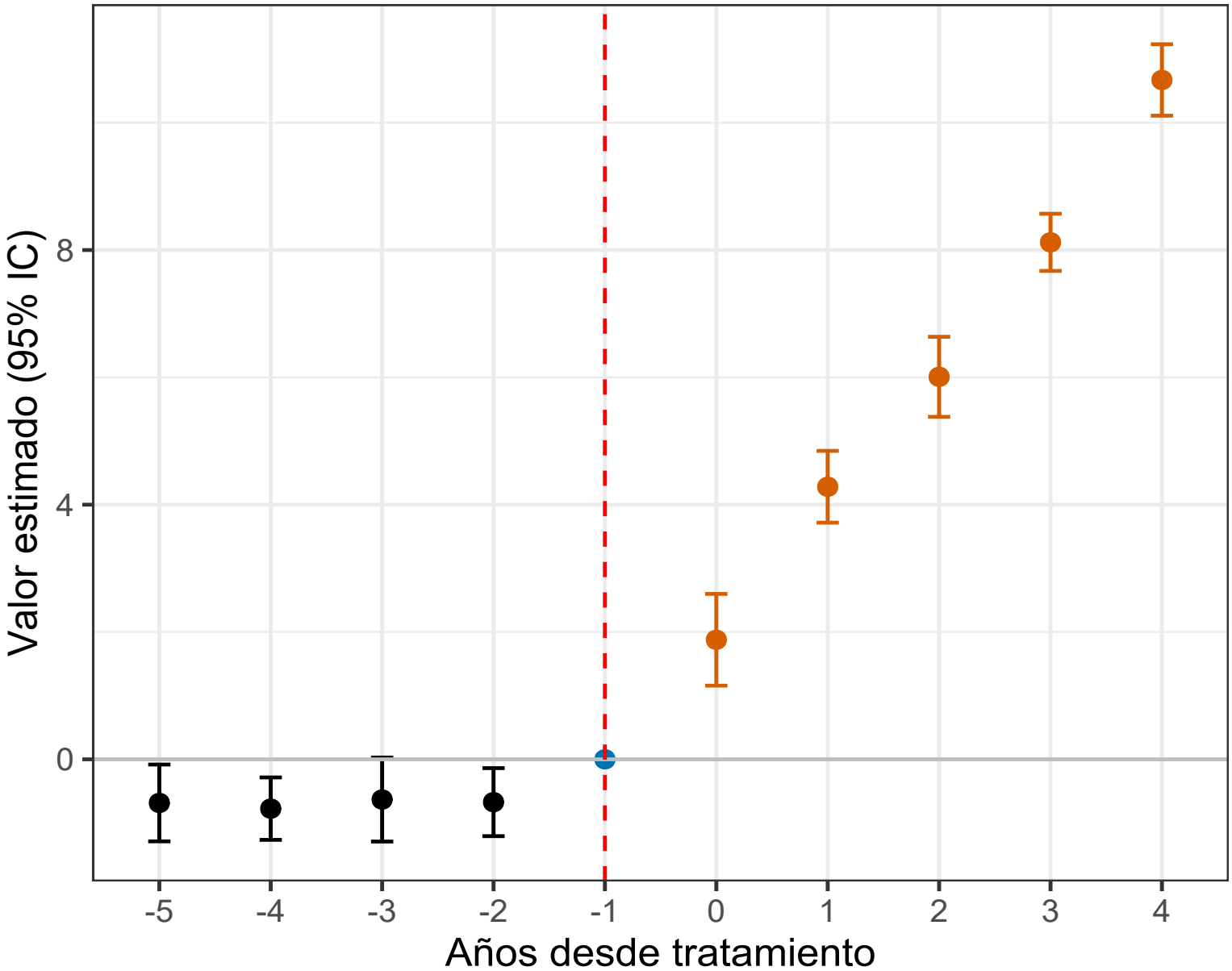
- Utilizando ggplot(), noten que eje X es el tiempo al tratamiento, el eje y es el valor del coeficiente:

```
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 = 5,
                    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")
```

Event Study: Parallel Trends



Event Study: Pre-Trends



Leads y Lags

- Para probar que se cumplen tanto los supuestos de tendencias paralelas como los de no anticipación, podemos también correr una regresión con Leads y Lags.
- Como veremos más adelante, si podemos hacer un Event Study (el tratamiento es discreto) nunca recurriremos a este test ya que perdemos observaciones.
- La idea general es analizar que sucede si adelantamos el tratamiento n periodos, a traves de la siguiente especificación:

$$\begin{aligned} Y_{it} =& \alpha + \gamma_i + \lambda_t + \tau Treat_i \cdot Post_t + \sum_{t'=1}^{f} \delta_{t'} Treat_i \cdot Post_{t+t'} + \\ & \sum_{t'=1}^{l} \delta_{-t'} Treat_i \cdot Post_{t-t'} + \epsilon_{it} \end{aligned} \tag{1}$$

- Donde f es el número de adelantos y l el de retrasos.

Leads y Lags

- Lo primero que hay que hacer es generar estas variables.

```
# Creamos múltiples leads and lags
dd <- pdata.frame(dd_data, index = c("unidades", "time"))
dd <- cbind(dd, plm::lag(dd$treat_post, c(-2, -1, 1, 2)))
dd <- as.data.frame(transform(dd))
names(dd)[(length(dd) - 3):length(dd)] <- c("tratamiento_Lead_2",
            "tratamiento_Lead_1", "tratamiento_Lag_1",
            "tratamiento_Lag_2")

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

# Cambiamos los valores NAs
dd[dd$treatment == 0, c("tratamiento_Lead_2",
                        "tratamiento_Lead_1", "tratamiento_Lag_1",
                        "tratamiento_Lag_2")] <- 0
```

Leads y Lags

- Y corremos la regresión utilizando felm()

```
leads1 <- felm(wage_pretrend ~  treat_post + tratamiento_Lead_1 +
                tratamiento_Lead_2 + tratamiento_Lag_1 +
                tratamiento_Lag_2 | unidades + time | 0 | unidades, dd)


leads2 <- felm(wage ~ treat_post + tratamiento_Lead_1  +
                tratamiento_Lead_2 + tratamiento_Lag_1 +
                tratamiento_Lag_2  |  unidades + time | 0 | unidades, dd)
```

- Los resultados se pueden presentar tanto en tabla, como en gráfica.

Leads y Lags

- Para la tabla usamos stargazer()

```
# Tabla
dep_var <- c("\\shortstack{Differential \\\\ Pre-Trend}",
        "\\shortstack{Non-Differential \\\\ Pre-Trend}")
tablaleads <- stargazer(leads1, leads2,
                        header = FALSE,
                        font.size = "scriptsize",
                        dep.var.labels.include = FALSE,
                        table.placement = "H",
                        omit = c("Constant", "unidades", "time",
                             "tratamiento_Lag_1",
                             "tratamiento_Lag_2"),
                        column.labels = dep_var,
                        covariate.labels = c("Treatment", "Lead 1", "Lead 2"),
                        omit.stat = c("f", "ser","adj.rsq"),
                        add.lines = list(c("Unit Fixed Effects", "Yes", "Yes"),
                                  c("Time Fixed Effects", "Yes", "Yes")),
                        title = "Leads and Lags",
                        type = "latex")

note.latex <- "\\multicolumn{3}{c} {\\parbox[t]{7cm}{ \\textit{Notes:}
Standard errors are clustered at the time level.
Lag variables are included but not shown.
* denotes p$<$0.1, ** denotes p$<$0.05, and *** denotes p$<$0.01.}} \\\\"
tablaleads[grepl("Note", tablaleads)] <- note.latex

cat(tablaleads)
```

Leads y Lags

	Dependent variable:	
	Differential Pre-Trend	Non-Differential Pre-Trend
	(1)	(2)
Treatment	1.878***	1.878***
	(0.337)	(0.337)
Lead 1	0.673**	−0.027
	(0.250)	(0.250)
Lead 2	−0.041	−0.041
	(0.231)	(0.231)
Unit Fixed Effects	Yes	Yes
Time Fixed Effects	Yes	Yes
Observations	64	64
R^2	0.996	0.996

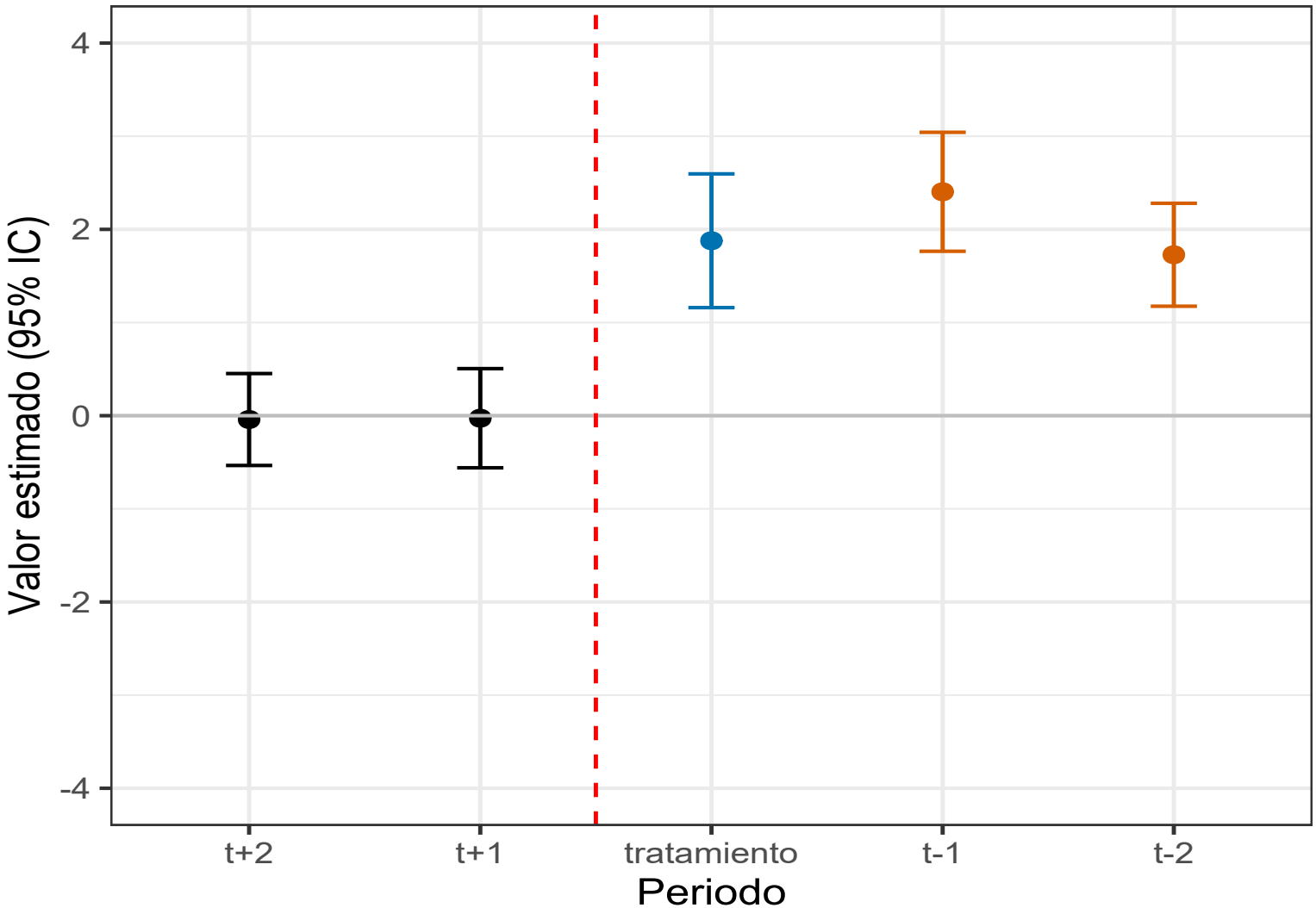
Notes: Standard errors are clustered at the time level. Lag variables are included but not shown. * denotes p<0.1, ** denotes p<0.05, and *** denotes p<0.01.

Leads y Lags

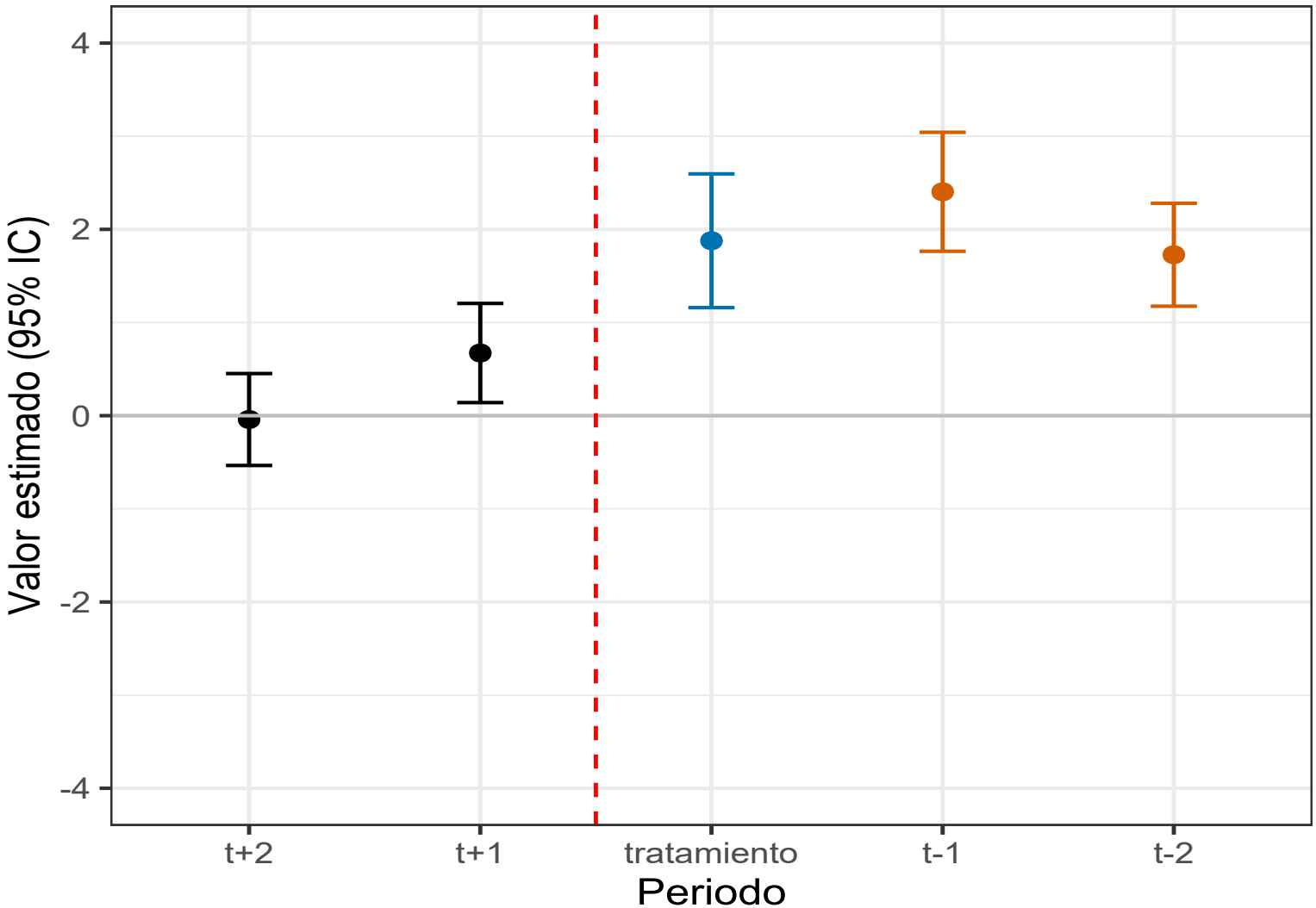
- Para la gráfica, generamos la base y usamos ggplot:

```
# leads2 para no pre-trends
datos <- data.frame(coeficientes=coef(leads1), ses = leads1$se , time <- c(0:2, -1, -2),
                    type = rep(1:3, c(2,1,2)))
datos$time <- factor(datos$time, levels = c(2,1,0,-1,-2))

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 = 2.5,linetype="dashed", color ="red", 2) +
     theme_bw() +
     ylab("Valor estimado (95% IC)") +
     xlab("Periodo") +
     scale_x_discrete(labels = c("t+2", "t+1", "tratamiento", "t-1", "t-2"), breaks = 2:-2) +
     scale_color_manual(name = "Periodo", values= colors) +
     theme(legend.position = "none")
```

Parallel Trends



Pre-Trends



Leads y Lags

- Uno de los problemas de correr leads y lags es que pierdes observaciones igual al número de rezagos/adelantos que incluyas $\times$ el número de unidades que uno tiene.
- Esto puede darnos coeficientes sesgados y llevarnos a conclusiones erróneas.
- Un truco que podemos utilizar, si tenemos una base de tratamiento con mayor número de periodos que la de outcomes, es realizar los Leads/Lags en esa base.
- Supongamos que tenemos dos bases distintas, tratamiento y outcomes:

```
df_outcomes <- dd_data |> select(unidades, time, wage)

df_tratamiento <- dd_data |> select(unidades, time, treat_post)
```

Leads y Lags

- Más aún, supongamos que la base de tratamiento tiene 2 periodos más que la de outcomes:

```
df_2020 <- tibble(unidades = c(1:10), time = 11,
                  treat_post = rep(c(0,1), 5))
df_2021 <- tibble(unidades = c(1:10), time = 12,
                  treat_post = rep(c(0,1), 5))

df_tratamiento <- rbind(df_tratamiento, df_2020, df_2021)

df_tratamiento <- df_tratamiento[order(df_tratamiento[,1],
                                       df_tratamiento[,2]),]
```

- Noten que nos acoplamos a la misma simulación.

Leads y Lags

- Ahora nos conviene realizar los leads en esta base, ya que tenemos dos periodos más de donde conseguir información:

```
df_tratamiento <- pdata.frame(df_tratamiento,
                  index = c("unidades","time"))

df_tratamiento <- cbind(df_tratamiento,
                    plm::lead(df_tratamiento$treat_post, c(1,2)))
df_tratamiento <- as.data.frame(transform(df_tratamiento))

names(df_tratamiento)[(length(df_tratamiento) - 1):length(df_tratamiento)] <-
             c("tratamiento_Lead_1", "tratamiento_Lead_2")

df_tratamiento <- unfactor(df_tratamiento)
```

Leads y Lags

- Unimos con nuestra base de outcomes, noten que no perdemos ninguna observación:

```
df_outcomes <- df_outcomes |> left_join(df_tratamiento,
                                        by = c("unidades", "time"))

sum(is.na(df_outcomes$tratamiento_Lead_1))
#### result [1]: 0
sum(is.na(dd$tratamiento_Lead_1))
#### result [1]: 4
```

Leads y Lags

- Corremos la regresión y hacemos la tabla:

```
reg_leads <- felm(wage ~ treat_post + tratamiento_Lead_1 +
                    tratamiento_Lead_2 | unidades + time | 0 |
                    unidades, data = df_outcomes)

tablaleads <- stargazer(reg_leads,
                        omit = c("Constant", "unidades", "time"),
                        covariate.labels = c("Tratamiento", "Adelanto 1",
                                             "Adelanto 2"),
                        omit.stat = c("f", "ser","adj.rsq"),
                        add.lines = list(c("Unit Fixed Effects", "Yes"),
                                         c("Time Fixed Effects", "Yes")),
                        title = "Leads and Lags",
                        type = "latex")
```

Leads y Lags

	Dependent variable:	
	wage	
	(1)	(2)
Tratamiento	6.192***	6.185***
	(0.210)	(0.066)
Adelanto 1	−0.027	
	(0.244)	
Adelanto 2	0.024	
	(0.237)	
Unit Fixed Effects	Yes	Yes
Time Fixed Effects	Yes	Yes
Observations	80	80
R^2	0.941	0.941
Note:	*p<0.1; **p<0.05; ***p<0.01	

Tratamiento Continuo

- Supongamos ahora que tenemos una base de datos tipo panel que tiene unidades, tiempo, un **tratamiento continuo** y una variable que llamamos *outcome*.
- Cuando hablamos de tratamiento continuo, regularmente nos referimos a *differences in intensity*.
- Un buen ejemplo es el de cobertura 3G, donde la variación entre municipios es en intensidad (porcentaje de cobertura).
- En general el planteamiento es el mismo, pero la forma de correr las especificaciones difiere.

Tratamiento Continuo

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

TWFE

- El planteamiento del TWFE es análogo al de tratamiento dicotómico.

```
reg_did <- felm(outcome ~ tratamiento | unidades + tiempo | 0 | unidades,
            data = data_continuo)

tabldedd <- stargazer(reg_did,
                      header = FALSE,
                      font.size = "scriptsize",
                      dep.var.labels.include = FALSE,
                      table.placement = "H",
                      omit = c("Constant", "unidades", "time"),
                      column.labels = c("TWFE"),
                      covariate.labels = c("DiD Estimate"),
                      omit.stat = c("f", "ser","adj.rsq"),
                      add.lines = list(c("Unit Fixed Effects","Yes"),
                                       c("Time Fixed Effects", "Yes")),
                      title = "DiD Continuo",
                      type = "latex")
```

TWFE

Table: DiD Continuo

	Dependent variable:
	TWFE
DiD Estimate	3.434***
	(0.123)
Unit Fixed Effects	Yes
Time Fixed Effects	Yes
Observations	200
R^2	0.900
Note:	*p<0.1; **p<0.05; ***p<0.01

Leads y Lags

- Para Leads y Lags, el código es casi idéntico al caso de tratamiento dicótomico:

```
dd <- pdata.frame(data_continuo, index = c("unidades","tiempo"))
dd <- cbind(dd, plm::lag(dd$tratamiento, c(-3:3)[-4]))
dd <- as.data.frame(transform(dd))

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")

dd <- unfactor(dd)

dd[dd$treat == 0,
c("tratamiento_Lead_3", "tratamiento_Lead_2",
    "tratamiento_Lead_1", "tratamiento_Lag_1",
      "tratamiento_Lag_2", "tratamiento_Lag_3")] <- 0
```

Leads y Lags

- Corremos la especificación:

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

table_leads_lags <- stargazer(reg1, header = FALSE,
                              font.size = "footnotesize",
                              dep.var.caption = "",
                              label = "tab:tablaR",
                              dep.var.labels.include = FALSE,
                              table.placement = "H",
                              omit = c("Constant", "year", "unidades", "Lag"),
                              column.labels = "Outcome",
                              covariate.labels = c("Tratamiento",
                                                   "Tratamiento Lead 1",
                                                   "Tratamiento Lead 2",
                                                   "Tratamiento Lead 3"),
                              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")
```

Leads y Lags

- Notamos la magnitud del problema, perdemos demasiadas observaciones ya que tenemos 3 lags y 3 leads.

	Outcome	Outcome
	(1)	(2)
Tratamiento	2.746***	3.434***
	(0.396)	(0.123)
Tratamiento Lead 1	0.224	
	(0.414)	
Tratamiento Lead 2	0.345	
	(0.440)	
Tratamiento Lead 3	0.789	
	(0.631)	
Observations	80	200
R^2	0.912	0.900
Note:	*p<0.1; **p<0.05; ***p<0.01	

Event Study

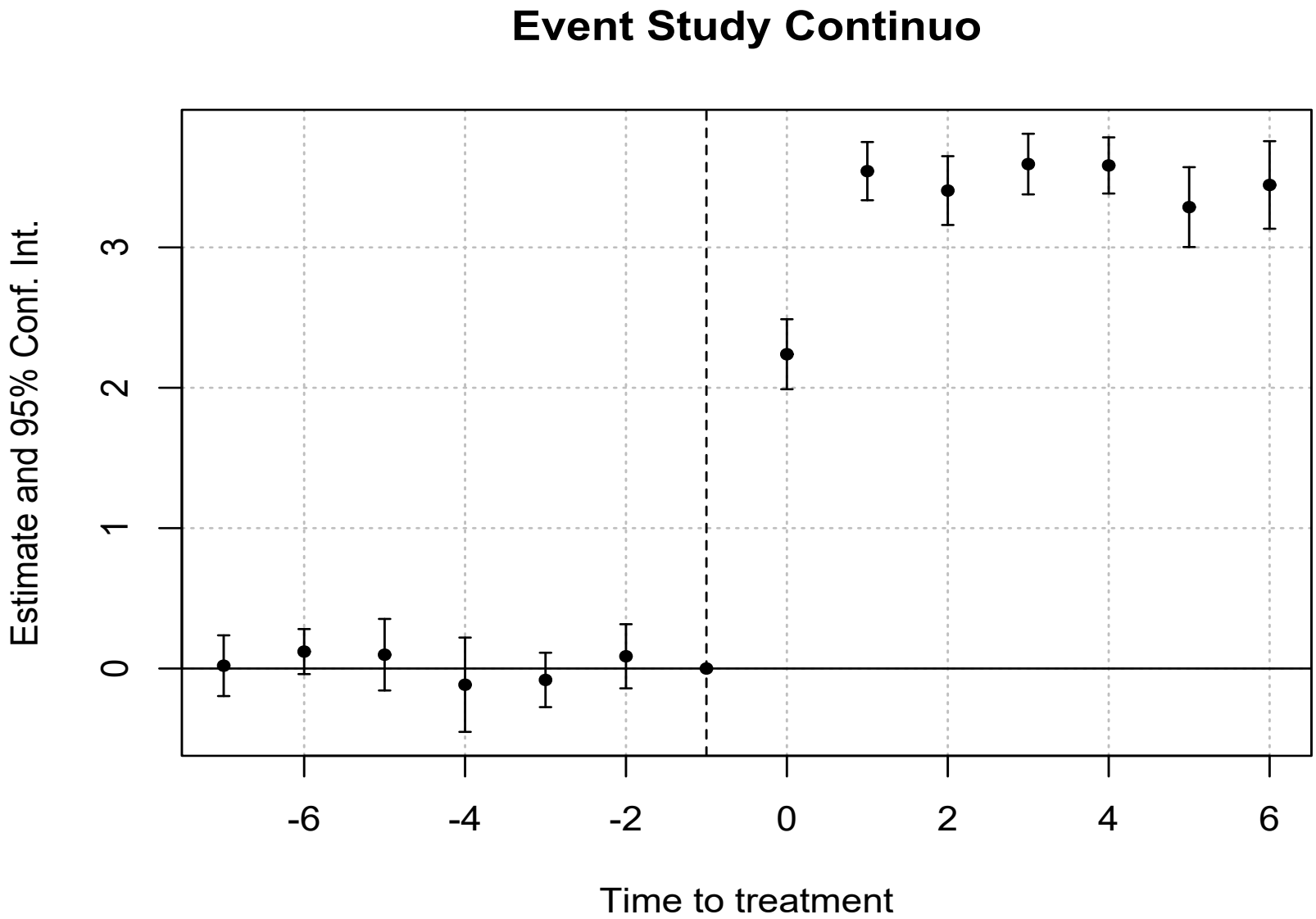
- Lo más eficiente y preciso para checar tendencias paralelas es hacer un Event Study.
- Sin embargo, en el caso continuo, para realizar el Event Study sin usar otros paquetes econométricos que se apoyen en utilizar *not-yet-treated units* como control, necesitamos definir que unidades son de tratamiento y cuáles de control.
- La forma más sencilla, es si en algún momento del tiempo el tratamiento continuo se vuelve mayor a cierta δ para esa unidad (en este caso la mediana).
- Es decir, detectar *never-treated units*. En código:

```
data_continuo <- data_continuo |> group_by(unidades) %>%
        mutate(treat = ifelse(any(tratamiento >=
                    median(data_continuo$tratamiento)), 1 ,0))
```

Event Study

- Hacemos unas últimas modificaciones, y corremos el código que ya habíamos visto:

```
  data_continuo <- data_continuo |> group_by(unidades) |>
                mutate(event_time = tiempo[tratamiento > 0][1]) |>
                ungroup()

  data_continuo <- data_continuo %>% group_by(unidades) %>%
                mutate(time_to_event = tiempo - event_time) |>
                ungroup() |>
                mutate(time_to_event = ifelse(is.na(time_to_event) == T,
                        0, time_to_event))

dd_plot_cont <- feols(outcome ~ i(time_to_event, treat, ref = -1) |
                      unidades + tiempo, cluster = "unidades",
                     data = data_continuo)

  ip <- iplot(dd_plot_cont,
                xlab = 'Time to treatment',
                main = 'Event Study Continuo')
```


Event Study Continuo
Estimate and 95% Conf. Int.
0
1
2
3
-6
-4
-2
0
2
4
6
Time to treatment