Análisis de resultados: Primera aplicación

Contents

Introducción

Este documento tiene como objetivo reportar el análisis de resultados psicométricos y pre-post de la aplicación de pruebas de Actitudes, Funciones ejecutivas, Motivación y Habilidades socioemocionales, realizados en el marco del Proyecto de evaluación del Plan Todo al Cole desarrollado por la Fundación Pies Descalzos.

El análisis psicométrico consiste en la obtención de indicadores de calidad de los ítems y pruebas. Los indicadores utilizado se listan a continuación:

Ítems dicotómicos

- *Sample.SD* representa la desviación estándar del ítem.
- *Item.total* muestra la correlación ítem-total.
- *Item.Tot.woi* representa la correlación del ítem con el total de la prueba, excluyendo al ítem en cuestión. Este indicador está muy ligado a la confiabilidad, por lo que valores inferiores a .10 no son deseados, y valores negativos representan ítems con problemas.
- *Difficulty* la dificultad según la TCT. Para este caso, lo mejor sería que los indicadores se encontraran entre el 0.10 y el 0.90
- *Discrimination* la disriminación entre tercios. Se recomiendan valores superiores a 0.20
- *Item.Reliab* la confiabilidad del ítem. Su función es medir la contribución del ítem a la medida final del test.
- *Item.Rel.woi* la confiabilidad del ítem, excluyendo al ítem en el total del test utilizado en la fórmula. Su función es medir la contribución del ítem a la medida final del test. Este indicador es interesante a la hora de mezclar ítems de ambas formas de prueba ya que da una guía de su posible comportamiento.

Ítems en escala likert

- *Dificulty*: Dificultad desde TCT
- *Mean*: Media del ítem
- *SD*: Desviación estandar del ítem
- *Prop.max.score*: La proporción de sujetos que escogió la máxima categoría

- *RIR*: Correlación entre el ítem y el resultado de la prueba sin contar el ítem.
- *RIT*: Correlación entre el ítem y el resultado de la prueba
- *ULI*: Discriminación upper-lower
- *Alpha.drop*: Alpha de Cronbach sin el ítem
- *Index.rel*: Índice de confiabilidad del ítem

Adicionalmente, se realizó un análisis pre y post de los resultados de los estudiantes en las pruebas. Dicho análisis consistió en una comparación de medias para muestras relacionadas, mediante la prueba *W de Wilcoxon*, así también se estimó el tamaño del efecto mediante el estadístico *d de Cohen*.

```
## Librerías

# Datos

library(readxl)
library(tidyr)
library(dplyr)
library(googlesheets4)
library(stringr)

# Análsis de datos
library(likert)
library(nortest)
library(effsize)
library(psychometric)
library(ShinyItemAnalysis)

#Graficas
library(ggplot2)
library(gt)
library(DT)

#Funciones propias

source("./Functions/min_max_scaler.R")
source("./Functions/calificacion.R")

# Otros
options(digits=5, scipen = 50)
set.seed(321)
# rmarkdown::render(input="1.0-bapinedam-Primera_aplicacion.Rmd",
#                   output_file = "../Pdf/Resultados primera aplicacion.pdf")
```

Obtención de datos

Para este proyecto las bases de datos se obtienen directamente desde internet, especificamente, desde google drive, debido a que pueden agregarse datos y es necesario que cada vez que se ejecute el script, los datos estén actualizados.

```
# Data

### Autenticación de usuario
```

```
gs4_auth()
```

```
# pre

### Obtención de los datos

url_pre = paste("https://docs.google.com/spreadsheets/d",
                "/1Ry73ckzruTQxtDnkCSscs16M3cv3u9XgJlcg4sN94BE/edit?usp=sharing",
                sep = "")
sheet_names(url_pre)
```

```
##  [1] "Memoria Audi"       "Memoria Vis"        "Inhibiciòn"         "Flexibilidad"       "Actitudes"
##  [6] "Motivación"         "Socioemocional"     "Comparación listas" "Hoja 9"             "Hoja 11"
## [11] "Hoja 12"
```

```
# post

url_post = paste("https://docs.google.com/spreadsheets/d/",
                 "1tFkxbH5N9HMr5qRA-VF4JXh-MAJqW38-iidB4QBuKU4/edit#gid=1877736074",
                 sep = "")
sheet_names(url_post)
```

```
## [1] "Memoria Audi"   "Memoria Vis"    "Inhibición"     "Flexibilidad"   "Actitudes"      "Motivación"
```

Análisis de datos

Actitudes

En el caso de la prueba pre de actitudes, todas las claves con la B, es por ello que podemos calificar siguiendo la instrucción: Si es B entonces 1, si no, entonces 0.

```
# Pre
actitudes_pre = read_sheet(url_pre, sheet = "Actitudes")
actitudes_pre = actitudes_pre[,1:15]

col_items = paste(rep("Grupo", 9), rep(1:3, each = 3), rep(paste("_", rep(1:3, 3))))

vector = c(colnames(actitudes_pre)[1:6], col_items)
colnames(actitudes_pre) = vector

actitudes_pre = filter(actitudes_pre,
                       !is.na(`Código`),
                       !is.na(`Grupo 1 _ 1`),
                       `Grupo 1 _ 1` %in% c("A", "B", "O", "X"))
dim(actitudes_pre)
```

```
## [1] 1161   15
```

```
actitudes_pre[,7:15] = apply(actitudes_pre[,7:15], 2, function(x) str_to_upper(x))

actitudes_pre[,7:15] = apply(actitudes_pre[,7:15], 2,
                             function(x) {ifelse(x == "B", 1, 0)})
```

En el caso de la prueba ppost de actitudes, no todos los ítems tienen la misma clave. Es por ello que creamos una función que tome un vector con las claves y nos califique una a una las columnas.

```r
actitudes_post = read_sheet(url_post, sheet = "Actitudes")
actitudes_post = actitudes_post[,1:15]

col_items = paste(rep("Grupo", 9), rep(1:3, each = 3), rep(paste("_", rep(1:3, 3))))

vector = c(colnames(actitudes_post)[1:6], col_items)
colnames(actitudes_post) = vector

actitudes_post[,7:15] = apply(actitudes_post[,7:15], 2, function(x) as.character(x))



actitudes_post = filter(actitudes_post,
                        !is.na(`Código`),
                        !is.na(`Grupo 1 _ 1`),
                        `Grupo 1 _ 1` %in% c("A", "B", "O", "X"))

actitudes_post[,7:15] = apply(actitudes_post[,7:15], 2, function(x) str_to_upper(x))

claves_matematicas = c('B','A','A','A','A','B','A','A','B')

claves_lenguaje = c('A','A','B','B','A','A','A','B','A')
```

Actitudes hacia el lenguaje

Todos los estudiantes tienen un código. Si el mismo empieza en 1, es porque el estudiante estuvo en el programa de lenguaje, si tiene dos, es porque estuvo en el programa de mejora de matemáticas. En este caso filtramos por el 1.

```r
actitudes_lenguaje_pre = filter(actitudes_pre, substr(`Código`, 1, 1) == "1")
actitudes_lenguaje_post = filter(actitudes_post, substr(`Código`, 1, 1) == "1")
```

```r
# Calificación post
actitudes_lenguaje_post[,7:15] = calificacion(actitudes_lenguaje_post[,7:15],
                                              claves_lenguaje)
```

Finalmente, ya que tenemos calificados todos los ítems, obtenemos puntuaciones generales.

```r
# Calificación

# Total

actitudes_lenguaje_pre$Total_pre = apply(actitudes_lenguaje_pre[,7:15], 1,
                                         function(x) sum(x, na.rm= FALSE))
actitudes_lenguaje_post$Total_post = apply(actitudes_lenguaje_post[,7:15], 1,
                                           function(x) sum(x, na.rm= FALSE))

# Afectivo

actitudes_lenguaje_pre$Afectivo_pre = apply(actitudes_lenguaje_pre[,c(7, 10, 13)], 1,
                                            function(x) sum(x, na.rm= FALSE))
actitudes_lenguaje_post$Afectivo_post = apply(actitudes_lenguaje_post[,c(7, 10, 13)], 1,
                                              function(x) sum(x, na.rm= FALSE))

# Cognitivo
```

```r
actitudes_lenguaje_pre$Cognitivo_pre = apply(actitudes_lenguaje_pre[,c(8, 11, 14)], 1,
                                             function(x) sum(x, na.rm= FALSE))
actitudes_lenguaje_post$Cognitivo_post = apply(actitudes_lenguaje_post[,c(8, 11, 14)], 1,
                                               function(x) sum(x, na.rm= FALSE))

# Conativo

actitudes_lenguaje_pre$Conativo_pre = apply(actitudes_lenguaje_pre[,c(9, 12, 15)], 1,
                                            function(x) sum(x, na.rm= FALSE))
actitudes_lenguaje_post$Conativo_post = apply(actitudes_lenguaje_post[,c(9, 12, 15)], 1,
                                              function(x) sum(x, na.rm= FALSE))

# Matriz pre y post para comparación de muestras

pre_post = inner_join(actitudes_lenguaje_post,
                      dplyr::select(actitudes_lenguaje_pre, c("Código",
                                                              "Total_pre",
                                                              "Afectivo_pre",
                                                              "Cognitivo_pre",
                                                              "Conativo_pre")),
                      by = "Código")
```

Estadísticos psicométricos

Alpha

```r
alpha(actitudes_lenguaje_pre[,7:15])
```

Total pre

```
## [1] 0.80124
```

```r
for(i in seq(length(colnames(actitudes_lenguaje_pre[,7:15])))){
    x = alpha(actitudes_lenguaje_pre[,7:15][,-i])
    print(paste("El índice de confiabilidad cambia a", round(x, 2), "al eliminar el ítem",
                colnames(actitudes_lenguaje_pre[,7:15])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.8 al eliminar el ítem Grupo 1 _ 1"
## [1] "El índice de confiabilidad cambia a 0.79 al eliminar el ítem Grupo 1 _ 2"
## [1] "El índice de confiabilidad cambia a 0.78 al eliminar el ítem Grupo 1 _ 3"
## [1] "El índice de confiabilidad cambia a 0.78 al eliminar el ítem Grupo 2 _ 1"
## [1] "El índice de confiabilidad cambia a 0.79 al eliminar el ítem Grupo 2 _ 2"
## [1] "El índice de confiabilidad cambia a 0.78 al eliminar el ítem Grupo 2 _ 3"
## [1] "El índice de confiabilidad cambia a 0.78 al eliminar el ítem Grupo 3 _ 1"
## [1] "El índice de confiabilidad cambia a 0.77 al eliminar el ítem Grupo 3 _ 2"
## [1] "El índice de confiabilidad cambia a 0.77 al eliminar el ítem Grupo 3 _ 3"
```

```r
alpha(actitudes_lenguaje_post[,7:15])
```

Total post

```
## [1] 0.79536
```

```
for(i in seq(length(colnames(actitudes_lenguaje_post[,7:15])))){
    x = alpha(actitudes_lenguaje_post[,7:15][,-i])
    print(paste("El índice de confiabilidad cambia a", round(x, 2), "al eliminar el ítem",
                colnames(actitudes_lenguaje_post[,7:15])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.79 al eliminar el ítem Grupo 1 _ 1"
## [1] "El índice de confiabilidad cambia a 0.78 al eliminar el ítem Grupo 1 _ 2"
## [1] "El índice de confiabilidad cambia a 0.79 al eliminar el ítem Grupo 1 _ 3"
## [1] "El índice de confiabilidad cambia a 0.77 al eliminar el ítem Grupo 2 _ 1"
## [1] "El índice de confiabilidad cambia a 0.77 al eliminar el ítem Grupo 2 _ 2"
## [1] "El índice de confiabilidad cambia a 0.77 al eliminar el ítem Grupo 2 _ 3"
## [1] "El índice de confiabilidad cambia a 0.77 al eliminar el ítem Grupo 3 _ 1"
## [1] "El índice de confiabilidad cambia a 0.77 al eliminar el ítem Grupo 3 _ 2"
## [1] "El índice de confiabilidad cambia a 0.77 al eliminar el ítem Grupo 3 _ 3"
```

```
# Creamos una base para guardar los valores alpha generales

alfa = data.frame(matrix(ncol = 2))
colnames(alfa) = c("Prueba", "Alfa")


# Amacenamos los valores

alfa = rbind(alfa,
             c("Actitudes lenguaje pre",
               alpha(actitudes_lenguaje_pre[,7:15])))

alfa = rbind(alfa,
             c("Actitudes lenguaje post",
               alpha(actitudes_lenguaje_post[,7:15])))

alfa
```

```
##                     Prueba              Alfa
## 1                     <NA>              <NA>
## 2  Actitudes lenguaje pre 0.801241357662175
## 3 Actitudes lenguaje post 0.795363951952661
```

```
alpha(actitudes_lenguaje_pre[,c(8, 11, 14)])
```

Componente cognitivo pre

```
## [1] 0.53779
```

```
for(i in seq(length(colnames(actitudes_lenguaje_pre[,c(8, 11, 14)])))){
  x = alpha(actitudes_lenguaje_pre[,c(8, 11, 14)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_lenguaje_pre[,c(8, 11, 14)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.469734544540396 al eliminar el ítem Grupo 1 _ 2"
## [1] "El índice de confiabilidad cambia a 0.422874341610233 al eliminar el ítem Grupo 2 _ 2"
## [1] "El índice de confiabilidad cambia a 0.408609400059248 al eliminar el ítem Grupo 3 _ 2"
```

```
alpha(actitudes_lenguaje_post[,c(8, 11, 14)])
```

Componente cognitivo post

```
## [1] 0.46791
```

```
for(i in seq(length(colnames(actitudes_lenguaje_post[,c(8, 11, 14)])))){
  x = alpha(actitudes_lenguaje_post[,c(8, 11, 14)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_lenguaje_post[,c(8, 11, 14)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.386946386946387 al eliminar el ítem Grupo 1 _ 2"
## [1] "El índice de confiabilidad cambia a 0.455425846279879 al eliminar el ítem Grupo 2 _ 2"
## [1] "El índice de confiabilidad cambia a 0.288 al eliminar el ítem Grupo 3 _ 2"
```

```
alpha(actitudes_lenguaje_pre[,c(7, 10, 13)])
```

Componente afectivo pre

```
## [1] 0.4658
```

```
for(i in seq(length(colnames(actitudes_lenguaje_pre[,c(7, 10, 13)])))){
  x = alpha(actitudes_lenguaje_pre[,c(7, 10, 13)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_lenguaje_pre[,c(7, 10, 13)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.420100382409178 al eliminar el ítem Grupo 1 _ 1"
## [1] "El índice de confiabilidad cambia a 0.278155418026868 al eliminar el ítem Grupo 2 _ 1"
## [1] "El índice de confiabilidad cambia a 0.407700609405394 al eliminar el ítem Grupo 3 _ 1"
```

```
alpha(actitudes_lenguaje_post[,c(7, 10, 13)])
```

Componente afectivo post

```
## [1] 0.51561
```

```
for(i in seq(length(colnames(actitudes_lenguaje_post[,c(7, 10, 13)])))){
  x = alpha(actitudes_lenguaje_post[,c(7, 10, 13)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_lenguaje_post[,c(7, 10, 13)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.505764346920505 al eliminar el ítem Grupo 1 _ 1"
## [1] "El índice de confiabilidad cambia a 0.360369881109643 al eliminar el ítem Grupo 2 _ 1"
## [1] "El índice de confiabilidad cambia a 0.335934134316213 al eliminar el ítem Grupo 3 _ 1"
```

```
alpha(actitudes_lenguaje_pre[,c(9, 12, 15)])
```

Componente conativo pre

```
## [1] 0.56206
```

```
for(i in seq(length(colnames(actitudes_lenguaje_pre[,c(9, 12, 15)])))){
  x = alpha(actitudes_lenguaje_pre[,c(9, 12, 15)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_lenguaje_pre[,c(9, 12, 15)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.443895859313364 al eliminar el ítem Grupo 1 _ 3"
## [1] "El índice de confiabilidad cambia a 0.421728685591018 al eliminar el ítem Grupo 2 _ 3"
## [1] "El índice de confiabilidad cambia a 0.524196670538134 al eliminar el ítem Grupo 3 _ 3"
```

```
alpha(actitudes_lenguaje_post[,c(9, 12, 15)])
```

Componente conativo post

```
## [1] 0.50883
```

```
for(i in seq(length(colnames(actitudes_lenguaje_post[,c(9, 12, 15)])))){
  x = alpha(actitudes_lenguaje_post[,c(9, 12, 15)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_lenguaje_post[,c(9, 12, 15)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.463281528057895 al eliminar el ítem Grupo 1 _ 3"
## [1] "El índice de confiabilidad cambia a 0.437565184412629 al eliminar el ítem Grupo 2 _ 3"
## [1] "El índice de confiabilidad cambia a 0.328165530554382 al eliminar el ítem Grupo 3 _ 3"
```

Indicadores psicométricos

```
analitem = item.exam(actitudes_lenguaje_pre[,7:15], discrim=T)
analitem = dplyr::select(analitem, -c("Item.Criterion", "Item.Validity"))
cols_num = colnames(analitem)
analitem$Item = row.names(analitem)
analitem %>%
  gt(rowname_col = "Item") %>%
  fmt_number(
    columns = cols_num,
    decimals = 3
  ) %>%
  tab_style(
    style = list(
      cell_text(align="center", weight="bold")
    ),
    locations=cells_column_labels()
  )%>%
  tab_style(
    style = list(
      cell_text(align="center")
    ),
    locations = cells_body())
```

Total pre

	Sample.SD	Item.total	Item.Tot.woi	Difficulty	Discrimination	Item.Reliab	Item.Rel.woi

Grupo 1 _ 1	0.444	0.535	0.377	0.730	0.538	0.238	0.168
Grupo 1 _ 2	0.347	0.559	0.443	0.860	0.369	0.194	0.154
Grupo 1 _ 3	0.406	0.645	0.524	0.792	0.518	0.262	0.213
Grupo 2 _ 1	0.403	0.599	0.469	0.797	0.456	0.241	0.189
Grupo 2 _ 2	0.327	0.569	0.463	0.879	0.303	0.186	0.151
Grupo 2 _ 3	0.382	0.619	0.501	0.823	0.446	0.237	0.192
Grupo 3 _ 1	0.473	0.670	0.533	0.664	0.723	0.317	0.252
Grupo 3 _ 2	0.471	0.712	0.588	0.669	0.764	0.335	0.277
Grupo 3 _ 3	0.459	0.683	0.553	0.700	0.703	0.313	0.254

```
# Creamos una base para guardar indicadores

indicadores_psicometricos = data.frame()

# Guardamos los indicadores importantes

psicometricos_pre = analitem %>%
  dplyr::select(c("Difficulty", "Discrimination")) %>%
  mutate(Prueba = "Actitudes hacia el lenguaje - Pre") %>%
  mutate(Item = row.names(.))

indicadores_psicometricos = rbind(indicadores_psicometricos,
                                  psicometricos_pre)
```

```
analitem = item.exam(actitudes_lenguaje_post[,7:15], discrim=T)
analitem = dplyr::select(analitem, -c("Item.Criterion", "Item.Validity"))
cols_num = colnames(analitem)
analitem$Item = row.names(analitem)
analitem %>%
  gt(rowname_col = "Item") %>%
  fmt_number(
    columns = cols_num,
    decimals = 3
  ) %>%
  tab_style(
    style = list(
      cell_text(align="center", weight="bold")
    ),
    locations=cells_column_labels()
  )%>%
  tab_style(
    style = list(
      cell_text(align="center")
    ),
    locations = cells_body())
```

Total post

	Sample.SD	Item.total	Item.Tot.woi	Difficulty	Discrimination	Item.Reliab	Item.Rel.woi
Grupo 1 _ 1	0.275	0.506	0.399	0.918	0.224	0.139	0.110
Grupo 1 _ 2	0.335	0.552	0.426	0.871	0.333	0.185	0.143

Grupo 1 _ 3	0.380	0.503	0.349	0.825	0.388	0.191	0.133
Grupo 2 _ 1	0.454	0.690	0.548	0.710	0.727	0.313	0.248
Grupo 2 _ 2	0.455	0.683	0.538	0.708	0.733	0.311	0.245
Grupo 2 _ 3	0.439	0.682	0.543	0.740	0.655	0.299	0.238
Grupo 3 _ 1	0.354	0.648	0.532	0.853	0.400	0.229	0.188
Grupo 3 _ 2	0.319	0.633	0.528	0.885	0.309	0.202	0.168
Grupo 3 _ 3	0.350	0.645	0.531	0.857	0.370	0.226	0.186

```
# Guardamos los indicadores importantes

psicometricos_post = analitem %>%
  dplyr::select(c("Difficulty", "Discrimination")) %>%
  mutate(Prueba = "Actitudes hacia el lenguaje - Post") %>%
  mutate(Item = row.names(.))

indicadores_psicometricos = rbind(indicadores_psicometricos,
                                  psicometricos_pre)
```

Comparación pre-post Pese a que la mayoría de las pruebas conservan el mismo número de preguntas finales, no es el caso para todas. Es por eso que, con el fin de permitir la comparación entre las pruebas pre y post, haremos un escalamiento min-max tal que todas las puntuaciones queden entre 0 y 1.

```
pre_post$Total_pre = min_max_scale(pre_post$Total_pre)
pre_post$Total_post = min_max_scale(pre_post$Total_post)
```

Prueba total Estadísticos de normalidad

```
print("Estadístico de normalidad pre")
```

```
## [1] "Estadístico de normalidad pre"
```

```
lillie.test(pre_post$Total_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Total_pre
## D = 0.209, p-value <0.0000000000000002
```

```
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```
lillie.test(pre_post$Total_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Total_post
## D = 0.25, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```r
summ = data.frame(unclass(psych::describe(pre_post[,c("Total_pre", "Total_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Total_pre", "Total_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Total_pre	434	0.77163	0.26358	0.88889	0.81418	0.16473	0	1	1
Total_post	434	0.82309	0.23157	0.88889	0.86654	0.16473	0	1	1

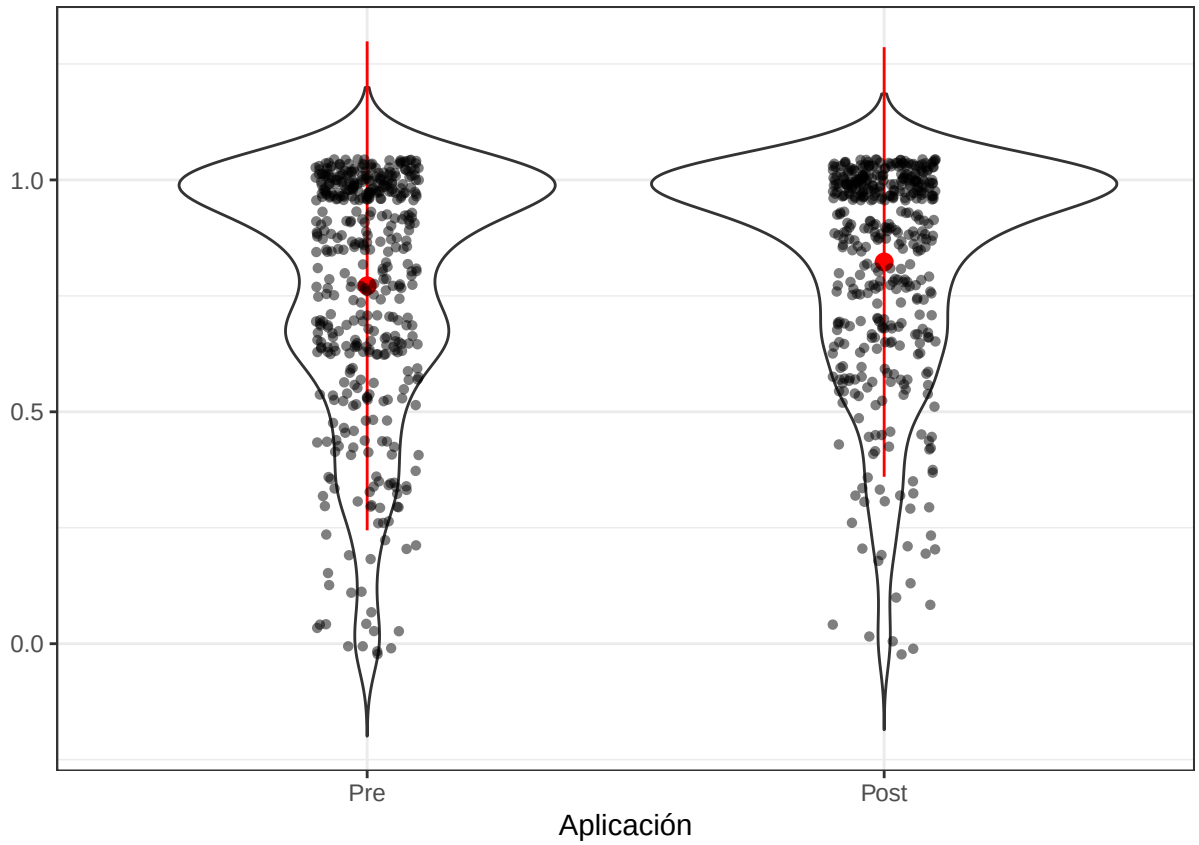
Comparación de medias

```r
comparacion =
  wilcox.test(
    x           = pre_post$Total_pre,
    y           = pre_post$Total_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Total_pre and pre_post$Total_post
## V = 16511, p-value = 0.0001
## alternative hypothesis: true location shift is not equal to 0
```

```r
pre_post %>%
  dplyr::select(Total_pre, Total_post) %>%
  pivot_longer(cols = c(Total_pre, Total_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/actitudes_lenguaje.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Total_post, pre_post$Total_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: 0.20706 (small)
## 95 percent confidence interval:
##    lower    upper
## 0.093858 0.320267
```

```
# Creamos una base para ir guardando los descriptivo

descriptivos = data.frame()

# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Actitudes hacia el lenguaje") %>%
          mutate(`Aplicación` = c("Pre", "Post"))
```

```
descriptivos = rbind(descriptivos, desc)
```

```
# Adicioalmente, guardaremos en otra tabla las comparaciones

comparaciones = data.frame()

# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Actitudes hacia el lenguaje - Total",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

```
pre_post$Afectivo_pre = min_max_scale(pre_post$Afectivo_pre)
pre_post$Afectivo_post = min_max_scale(pre_post$Afectivo_post)
```

Componente afectivo Estadísticos de normalidad

```
print("Estadístico de normalidad pre")
```

```
## [1] "Estadístico de normalidad pre"
```

```
lillie.test(pre_post$Afectivo_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Afectivo_pre
## D = 0.297, p-value <0.0000000000000002
```

```
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```
lillie.test(pre_post$Afectivo_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Afectivo_post
## D = 0.391, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```
summ = data.frame(unclass(psych::describe(pre_post[,c("Afectivo_pre",
                                                      "Afectivo_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Afectivo_pre", "Afectivo_post")
```

```
summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Afectivo_pre	434	0.73733	0.31000	0.66667	0.78161	0.4942	0	1	1
Afectivo_post	434	0.83180	0.26248	1.00000	0.88410	0.0000	0	1	1

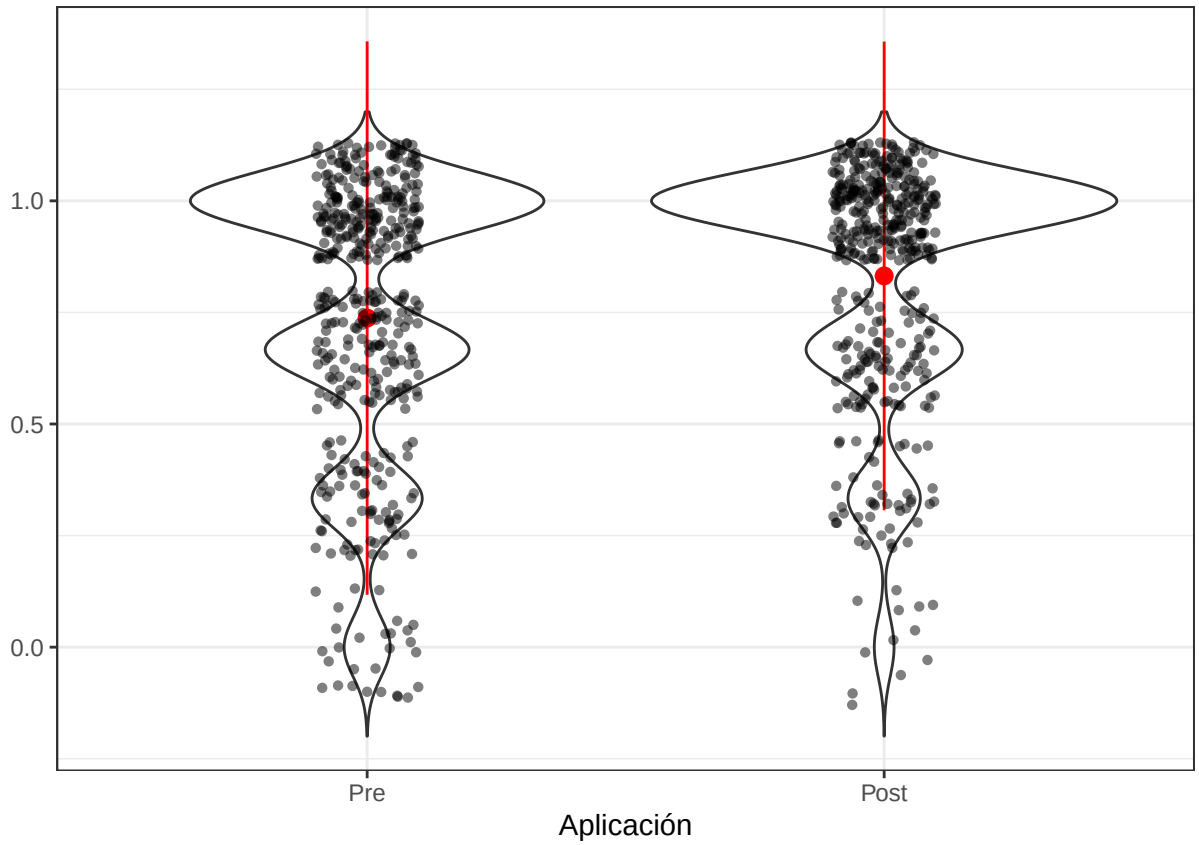
Comparación de medias

```
comparacion =
  wilcox.test(
    x           = pre_post$Afectivo_pre,
    y           = pre_post$Afectivo_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Afectivo_pre and pre_post$Afectivo_post
## V = 7034, p-value = 0.000000024
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Afectivo_pre, Afectivo_post) %>%
  pivot_longer(cols = c(Afectivo_pre, Afectivo_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/actitudes_lenguaje_afectivo.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Afectivo_post, pre_post$Afectivo_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: 0.32817 (small)
## 95 percent confidence interval:
##   lower   upper
## 0.20951 0.44683
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Actitudes hacia el lenguaje - Afectivo") %>%
          mutate(`Aplicación` = c("Pre", "Post"))

descriptivos = rbind(descriptivos, desc)
```

```r
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Actitudes hacia el lenguaje - Afectivo",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

```r
pre_post$Cognitivo_pre = min_max_scale(pre_post$Cognitivo_pre)
pre_post$Cognitivo_post = min_max_scale(pre_post$Cognitivo_post)
```

Componente Cognitivo Estadísticos de normalidad

```r
print("Estadístico de normalidad pre")
```

```
## [1] "Estadístico de normalidad pre"
```

```r
lillie.test(pre_post$Cognitivo_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Cognitivo_pre
## D = 0.366, p-value <0.0000000000000002
```

```r
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```r
lillie.test(pre_post$Cognitivo_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Cognitivo_post
## D = 0.375, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```r
summ = data.frame(unclass(psych::describe(pre_post[,c("Cognitivo_pre",
                                                      "Cognitivo_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Cognitivo_pre", "Cognitivo_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Cognitivo_pre	434	0.80722	0.27923	1	0.85824	0	0	1	1
Cognitivo_post	434	0.82565	0.25744	1	0.87548	0	0	1	1

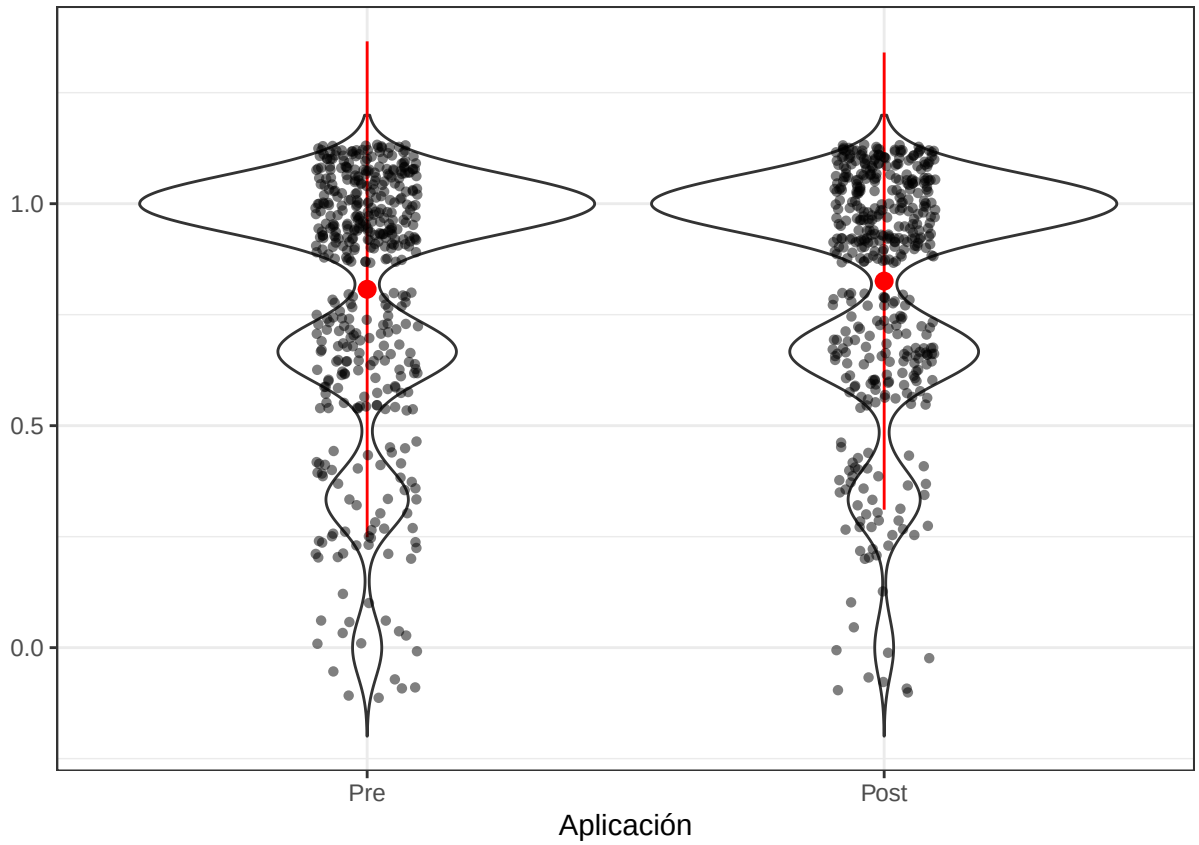
Comparación de medias

```
comparacion =
  wilcox.test(
    x           = pre_post$Cognitivo_pre,
    y           = pre_post$Cognitivo_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Cognitivo_pre and pre_post$Cognitivo_post
## V = 9908, p-value = 0.37
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Cognitivo_pre, Cognitivo_post) %>%
  pivot_longer(cols = c(Cognitivo_pre, Cognitivo_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/actitudes_lenguaje_cognitivo.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Cognitivo_post, pre_post$Cognitivo_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: 0.068602 (negligible)
## 95 percent confidence interval:
##     lower     upper
## -0.047008  0.184211
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Actitudes hacia el lenguaje - Cognitivo") %>%
          mutate(`Aplicación` = c("Pre", "Post"))

descriptivos = rbind(descriptivos, desc)
```

```r
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Actitudes hacia el lenguaje - Cognitivo",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

```r
pre_post$Conativo_pre = min_max_scale(pre_post$Conativo_pre)
pre_post$Conativo_post = min_max_scale(pre_post$Conativo_post)
```

Componente Conativo Estadísticos de normalidad

```r
print("Estadístico de normalidad pre")
```

```
## [1] "Estadístico de normalidad pre"
```

```r
lillie.test(pre_post$Conativo_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Conativo_pre
## D = 0.332, p-value <0.0000000000000002
```

```r
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```r
lillie.test(pre_post$Conativo_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Conativo_post
## D = 0.364, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```r
summ = data.frame(unclass(psych::describe(pre_post[,c("Conativo_pre",
                                                      "Conativo_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Conativo_pre", "Conativo_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Conativo_pre	434	0.77035	0.30436	1	0.82184	0	0	1	1
Conativo_post	434	0.81183	0.27499	1	0.86494	0	0	1	1

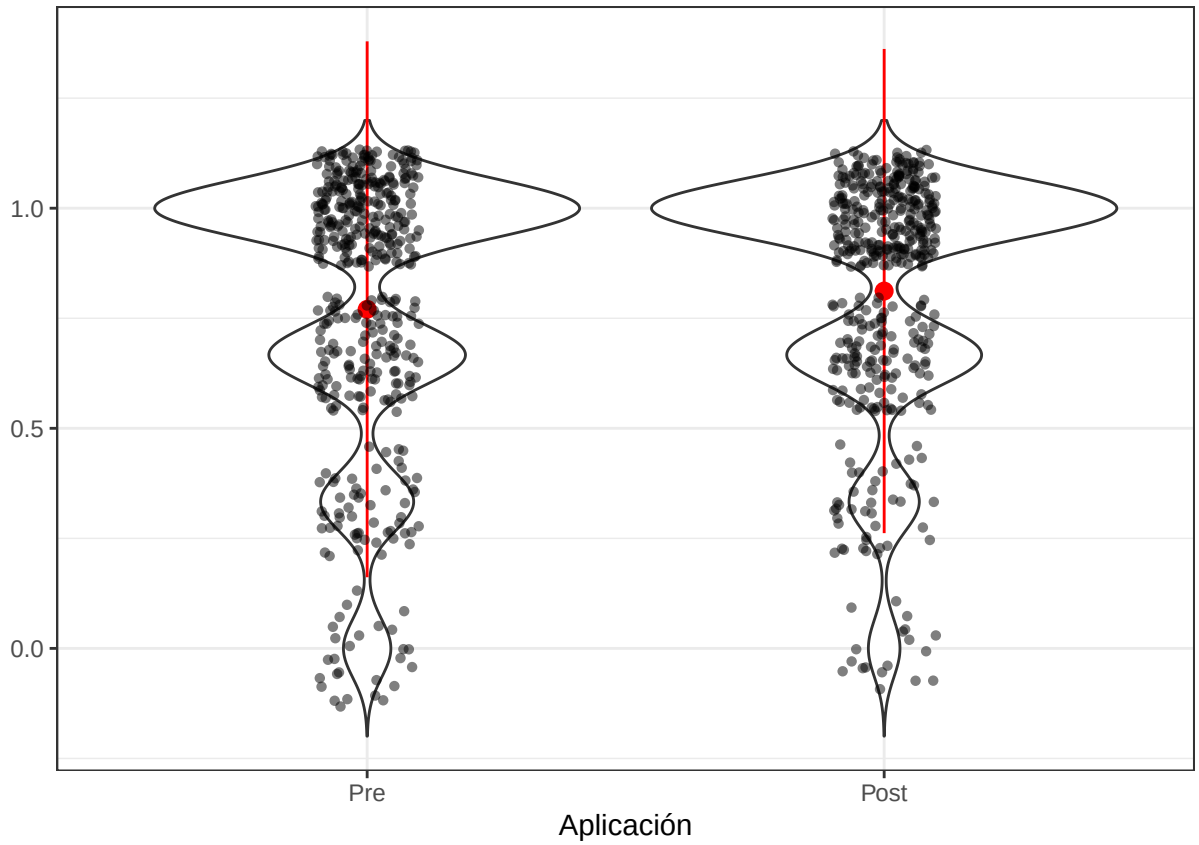
Comparación de medias

```
comparacion =
  wilcox.test(
    x           = pre_post$Conativo_pre,
    y           = pre_post$Conativo_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Conativo_pre and pre_post$Conativo_post
## V = 9699, p-value = 0.015
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Conativo_pre, Conativo_post) %>%
  pivot_longer(cols = c(Conativo_pre, Conativo_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/actitudes_lenguaje_conativo.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Conativo_post, pre_post$Conativo_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: 0.14286 (negligible)
## 95 percent confidence interval:
##    lower    upper
## 0.027998 0.257722
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Actitudes hacia el lenguaje - Conativo") %>%
          mutate(`Aplicación` = c("Pre", "Post"))

descriptivos = rbind(descriptivos, desc)
```

```r
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Actitudes hacia el lenguaje - Conativo",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

Actitudes hacia las matemáticas

```r
actitudes_matematicas_pre = filter(actitudes_pre,
                                   substr(`Código`, 1, 1) == "2")
actitudes_matematicas_post = filter(actitudes_post,
                                    substr(`Código`, 1, 1) == "2")
```

```r
actitudes_matematicas_post[,7:15] =
  calificacion(actitudes_matematicas_post[,7:15],
               claves_matematicas)
```

```r
# Calificación

# Total

actitudes_matematicas_pre$Total_pre =
  apply(actitudes_matematicas_pre[,7:15], 1,
        function(x) sum(x, na.rm= FALSE))

actitudes_matematicas_post$Total_post =
  apply(actitudes_matematicas_post[,7:15], 1,
        function(x) sum(x, na.rm= FALSE))

# Afectivo

actitudes_matematicas_pre$Afectivo_pre =
  apply(actitudes_matematicas_pre[,c(7, 10, 13)], 1,
        function(x) sum(x, na.rm= FALSE))

actitudes_matematicas_post$Afectivo_post =
  apply(actitudes_matematicas_post[,c(7, 10, 13)], 1,
        function(x) sum(x, na.rm= FALSE))

# Cognitivo

actitudes_matematicas_pre$Cognitivo_pre =
  apply(actitudes_matematicas_pre[,c(8, 11, 14)], 1,
        function(x) sum(x, na.rm= FALSE))

actitudes_matematicas_post$Cognitivo_post =
  apply(actitudes_matematicas_post[,c(8, 11, 14)], 1,
        function(x) sum(x, na.rm= FALSE))
```

```r
# Conativo

actitudes_matematicas_pre$Conativo_pre =
  apply(actitudes_matematicas_pre[,c(9, 12, 15)], 1,
        function(x) sum(x, na.rm= FALSE))

actitudes_matematicas_post$Conativo_post =
  apply(actitudes_matematicas_post[,c(9, 12, 15)], 1,
        function(x) sum(x, na.rm= FALSE))

pre_post = inner_join(actitudes_matematicas_post,
                      dplyr::select(actitudes_matematicas_pre,
                                    c("Código",
                                      "Total_pre",
                                      "Afectivo_pre",
                                      "Cognitivo_pre",
                                      "Conativo_pre")),
                      by = "Código")
```

Alpha

```r
alpha(actitudes_matematicas_pre[,7:15])
```

Total pre

```
## [1] 0.86738
```

```r
for(i in seq(length(colnames(actitudes_matematicas_pre[,7:15])))){
  x = alpha(actitudes_matematicas_pre[,7:15][,-i])
  print(paste("El índice de confiabilidad cambia a", x,
              "al eliminar el ítem",
              colnames(actitudes_matematicas_pre[,7:15])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.866927870344236 al eliminar el ítem Grupo 1 _ 1"
## [1] "El índice de confiabilidad cambia a 0.855002453978297 al eliminar el ítem Grupo 1 _ 2"
## [1] "El índice de confiabilidad cambia a 0.859822098694714 al eliminar el ítem Grupo 1 _ 3"
## [1] "El índice de confiabilidad cambia a 0.850284551708851 al eliminar el ítem Grupo 2 _ 1"
## [1] "El índice de confiabilidad cambia a 0.850608545831597 al eliminar el ítem Grupo 2 _ 2"
## [1] "El índice de confiabilidad cambia a 0.847485051486142 al eliminar el ítem Grupo 2 _ 3"
## [1] "El índice de confiabilidad cambia a 0.854742110885768 al eliminar el ítem Grupo 3 _ 1"
## [1] "El índice de confiabilidad cambia a 0.850128758539855 al eliminar el ítem Grupo 3 _ 2"
## [1] "El índice de confiabilidad cambia a 0.84367079172633 al eliminar el ítem Grupo 3 _ 3"
```

```r
alpha(actitudes_matematicas_post[,7:15])
```

Total post

```
## [1] 0.83482
```

```r
for(i in seq(length(colnames(actitudes_matematicas_post[,7:15])))){
  x = alpha(actitudes_matematicas_post[,7:15][,-i])
  print(paste("El índice de confiabilidad cambia a", x,
              "al eliminar el ítem",
```

```
             colnames(actitudes_matematicas_post[,7:15])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.832877002611357 al eliminar el ítem Grupo 1 _ 1"
## [1] "El índice de confiabilidad cambia a 0.822848131203732 al eliminar el ítem Grupo 1 _ 2"
## [1] "El índice de confiabilidad cambia a 0.827904824319074 al eliminar el ítem Grupo 1 _ 3"
## [1] "El índice de confiabilidad cambia a 0.811013698126052 al eliminar el ítem Grupo 2 _ 1"
## [1] "El índice de confiabilidad cambia a 0.813988819360638 al eliminar el ítem Grupo 2 _ 2"
## [1] "El índice de confiabilidad cambia a 0.810171881842676 al eliminar el ítem Grupo 2 _ 3"
## [1] "El índice de confiabilidad cambia a 0.817386151569538 al eliminar el ítem Grupo 3 _ 1"
## [1] "El índice de confiabilidad cambia a 0.812197977425821 al eliminar el ítem Grupo 3 _ 2"
## [1] "El índice de confiabilidad cambia a 0.812435498348564 al eliminar el ítem Grupo 3 _ 3"
```

```
# Amacenamos los valores

alfa = rbind(alfa,
             c("Actitudes matemáticas pre",
               alpha(actitudes_matematicas_pre[,7:15])))

alfa = rbind(alfa,
             c("Actitudes matemáticas post",
               alpha(actitudes_matematicas_post[,7:15])))

alfa
```

```
##                         Prueba              Alfa
## 1                         <NA>              <NA>
## 2     Actitudes lenguaje pre 0.801241357662175
## 3    Actitudes lenguaje post 0.795363951952661
## 4  Actitudes matemáticas pre 0.867379859042105
## 5 Actitudes matemáticas post 0.834823761460849
```

```
alpha(actitudes_matematicas_pre[,c(8, 11, 14)])
```

Componente cognitivo pre

```
## [1] 0.68445
```

```
for(i in seq(length(colnames(actitudes_matematicas_pre[,c(8, 11, 14)])))){
  x = alpha(actitudes_matematicas_pre[,c(8, 11, 14)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_matematicas_pre[,c(8, 11, 14)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.661498095641134 al eliminar el ítem Grupo 1 _ 2"
## [1] "El índice de confiabilidad cambia a 0.51016325986322 al eliminar el ítem Grupo 2 _ 2"
## [1] "El índice de confiabilidad cambia a 0.593103719430683 al eliminar el ítem Grupo 3 _ 2"
```

```
alpha(actitudes_matematicas_post[,c(8, 11, 14)])
```

Componente cognitivo post

```
## [1] 0.64522
```

```
for(i in seq(length(colnames(actitudes_matematicas_post[,c(8, 11, 14)])))){
  x = alpha(actitudes_matematicas_post[,c(8, 11, 14)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_matematicas_post[,c(8, 11, 14)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.594075988132835 al eliminar el ítem Grupo 1 _ 2"
## [1] "El índice de confiabilidad cambia a 0.472544763971785 al eliminar el ítem Grupo 2 _ 2"
## [1] "El índice de confiabilidad cambia a 0.572413793103448 al eliminar el ítem Grupo 3 _ 2"
```

```
alpha(actitudes_matematicas_pre[,c(7, 10, 13)])
```

Componente afectivo pre

```
## [1] 0.62622
```

```
for(i in seq(length(colnames(actitudes_matematicas_pre[,c(7, 10, 13)])))){
  x = alpha(actitudes_matematicas_pre[,c(7, 10, 13)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_matematicas_pre[,c(7, 10, 13)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.586827548172536 al eliminar el ítem Grupo 1 _ 1"
## [1] "El índice de confiabilidad cambia a 0.493480662983425 al eliminar el ítem Grupo 2 _ 1"
## [1] "El índice de confiabilidad cambia a 0.509852012660955 al eliminar el ítem Grupo 3 _ 1"
```

```
alpha(actitudes_matematicas_post[,c(7, 10, 13)])
```

Componente afectivo post

```
## [1] 0.53699
```

```
for(i in seq(length(colnames(actitudes_matematicas_post[,c(7, 10, 13)])))){
  x = alpha(actitudes_matematicas_post[,c(7, 10, 13)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_matematicas_post[,c(7, 10, 13)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.483615540914968 al eliminar el ítem Grupo 1 _ 1"
## [1] "El índice de confiabilidad cambia a 0.365691621219633 al eliminar el ítem Grupo 2 _ 1"
## [1] "El índice de confiabilidad cambia a 0.462860546243512 al eliminar el ítem Grupo 3 _ 1"
```

```
alpha(actitudes_matematicas_pre[,c(9, 12, 15)])
```

Componente conativo pre

```
## [1] 0.70346
```

```
for(i in seq(length(colnames(actitudes_matematicas_pre[,c(9, 12, 15)])))){
  x = alpha(actitudes_matematicas_pre[,c(9, 12, 15)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_matematicas_pre[,c(9, 12, 15)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.720924499229584 al eliminar el ítem Grupo 1 _ 3"
```

```
## [1] "El índice de confiabilidad cambia a 0.570734262681528 al eliminar el ítem Grupo 2 _ 3"
## [1] "El índice de confiabilidad cambia a 0.544375 al eliminar el ítem Grupo 3 _ 3"
```

```
alpha(actitudes_matematicas_post[,c(9, 12, 15)])
```

Componente conativo post

```
## [1] 0.59954
```

```
for(i in seq(length(colnames(actitudes_matematicas_post[,c(9, 12, 15)])))){
  x = alpha(actitudes_matematicas_post[,c(9, 12, 15)][,-i])
  print(paste("El índice de confiabilidad cambia a", x, "al eliminar el ítem",
              colnames(actitudes_matematicas_post[,c(9, 12, 15)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.616940335695273 al eliminar el ítem Grupo 1 _ 3"
## [1] "El índice de confiabilidad cambia a 0.399916989147552 al eliminar el ítem Grupo 2 _ 3"
## [1] "El índice de confiabilidad cambia a 0.48302367461677 al eliminar el ítem Grupo 3 _ 3"
```

Indicadores psicométricos

```
analitem = item.exam(actitudes_matematicas_pre[,7:15], discrim=T)
analitem = dplyr::select(analitem, -c("Item.Criterion", "Item.Validity"))
cols_num = colnames(analitem)
analitem$Item = row.names(analitem)
analitem %>%
  gt(rowname_col = "Item") %>%
  fmt_number(
    columns = cols_num,
    decimals = 3
  ) %>%
  tab_style(
    style = list(
      cell_text(align="center", weight="bold")
    ),
    locations=cells_column_labels()
  )%>%
  tab_style(
    style = list(
      cell_text(align="center")
    ),
    locations = cells_body())
```

Total pre

	Sample.SD	Item.total	Item.Tot.woi	Difficulty	Discrimination	Item.Reliab	Item.Rel.woi
Grupo 1 _ 1	0.416	0.611	0.476	0.777	0.539	0.254	0.198
Grupo 1 _ 2	0.327	0.673	0.582	0.878	0.366	0.220	0.190
Grupo 1 _ 3	0.396	0.655	0.537	0.805	0.524	0.259	0.213
Grupo 2 _ 1	0.315	0.721	0.643	0.889	0.319	0.227	0.202
Grupo 2 _ 2	0.331	0.716	0.633	0.875	0.356	0.237	0.210
Grupo 2 _ 3	0.333	0.746	0.669	0.873	0.361	0.248	0.223
Grupo 3 _ 1	0.396	0.696	0.587	0.805	0.513	0.276	0.233

Grupo 3 _ 2	0.360	0.723	0.633	0.847	0.440	0.260	0.228
Grupo 3 _ 3	0.375	0.777	0.697	0.831	0.497	0.291	0.261

```
# Guardamos los indicadores importantes

psicometricos_pre = analitem %>%
  dplyr::select(c("Difficulty", "Discrimination")) %>%
  mutate(Prueba = "Actitudes hacia las matematicas - Pre") %>%
  mutate(Item = row.names(.))

indicadores_psicometricos = rbind(indicadores_psicometricos,
                                  psicometricos_pre)
```

```
analitem = item.exam(actitudes_matematicas_post[,7:15], discrim=T)
analitem = dplyr::select(analitem, -c("Item.Criterion", "Item.Validity"))
cols_num = colnames(analitem)
analitem$Item = row.names(analitem)
analitem %>%
  gt(rowname_col = "Item") %>%
  fmt_number(
    columns = cols_num,
    decimals = 3
  ) %>%
  tab_style(
    style = list(
      cell_text(align="center", weight="bold")
    ),
    locations=cells_column_labels()
  )%>%
  tab_style(
    style = list(
      cell_text(align="center")
    ),
    locations = cells_body())
```

Total post

	Sample.SD	Item.total	Item.Tot.woi	Difficulty	Discrimination	Item.Reliab	Item.Rel.woi
Grupo 1 _ 1	0.398	0.586	0.433	0.803	0.512	0.233	0.172
Grupo 1 _ 2	0.298	0.604	0.497	0.902	0.256	0.180	0.148
Grupo 1 _ 3	0.381	0.606	0.465	0.824	0.477	0.231	0.177
Grupo 2 _ 1	0.334	0.705	0.606	0.873	0.366	0.235	0.202
Grupo 2 _ 2	0.313	0.681	0.584	0.890	0.314	0.213	0.183
Grupo 2 _ 3	0.336	0.711	0.613	0.871	0.360	0.239	0.206
Grupo 3 _ 1	0.344	0.661	0.547	0.863	0.390	0.227	0.188
Grupo 3 _ 2	0.348	0.699	0.593	0.859	0.390	0.243	0.206
Grupo 3 _ 3	0.316	0.693	0.598	0.888	0.337	0.219	0.189

```
# Guardamos los indicadores importantes

psicometricos_post = analitem %>%
```

```
  dplyr::select(c("Difficulty", "Discrimination")) %>%
  mutate(Prueba = "Actitudes hacia las matematicas - Post") %>%
  mutate(Item = row.names(.))

indicadores_psicometricos = rbind(indicadores_psicometricos,
                                  psicometricos_pre)
```

Comparación pre-post Pese a que la mayoría de las pruebas conservan el mismo número de preguntas finales, no es el caso para todas. Es por eso que, con el fin de permitir la comparación entre las pruebas pre y post, haremos un escalamiento min-max tal que todas las puntuaciones queden entre 0 y 1.

```
pre_post$Total_pre = min_max_scale(pre_post$Total_pre)
pre_post$Total_post = min_max_scale(pre_post$Total_post)
```

Prueba total Estadísticos de normalidad

```
print("Estadístico de normalidad pre")
```

```
## [1] "Estadístico de normalidad pre"
```

```
lillie.test(pre_post$Total_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Total_pre
## D = 0.287, p-value <0.0000000000000002
```

```
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```
lillie.test(pre_post$Total_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Total_post
## D = 0.312, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```
summ = data.frame(unclass(psych::describe(pre_post[,c("Total_pre",
                                                      "Total_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Total_pre", "Total_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Total_pre	468	0.84639	0.25118	1	0.90632	0	0	1	1

Total_post	468	0.87132	0.22117	1	0.92494	0	0	1	1

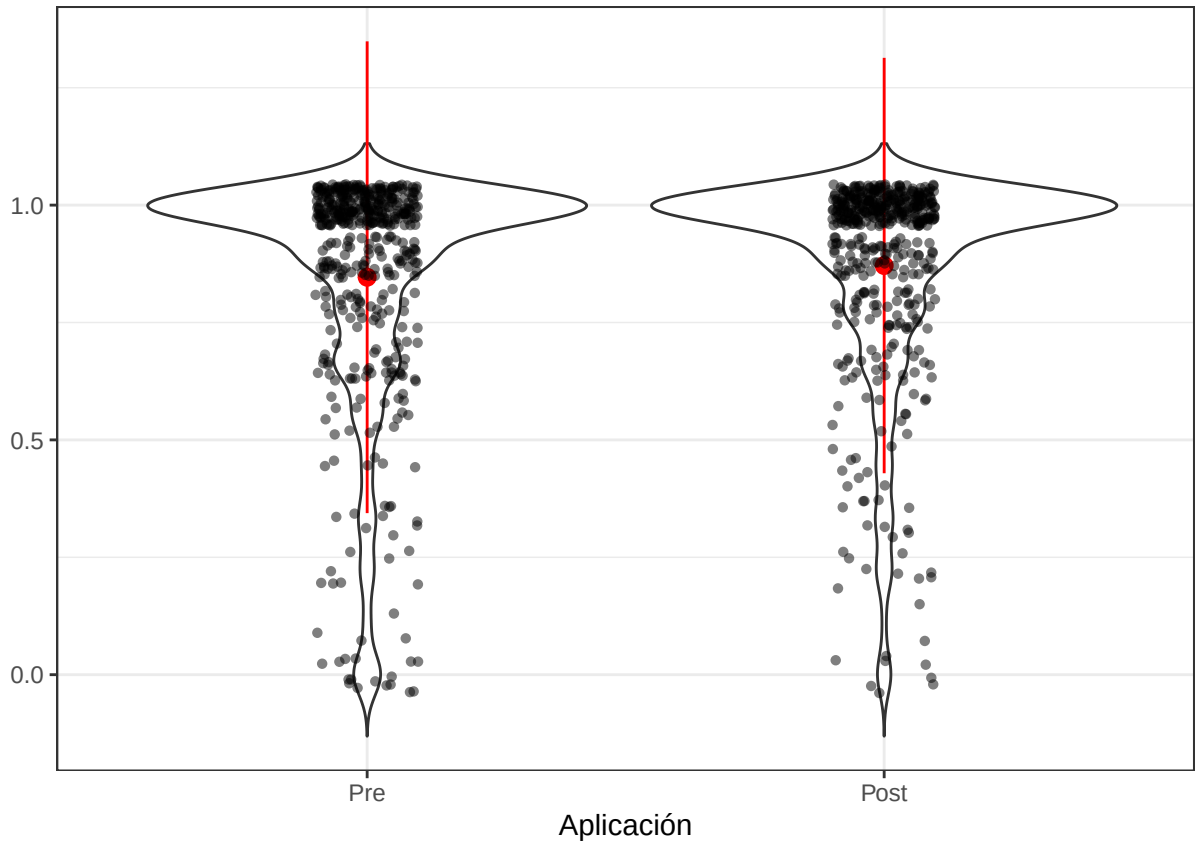
Comparación de medias

```
comparacion =
  wilcox.test(
    x           = pre_post$Total_pre,
    y           = pre_post$Total_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Total_pre and pre_post$Total_post
## V = 12310, p-value = 0.016
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Total_pre, Total_post) %>%
  pivot_longer(cols = c(Total_pre, Total_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/actitudes_matematicas.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Total_post, pre_post$Total_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: 0.10503 (negligible)
## 95 percent confidence interval:
##     lower     upper
## 0.0071053 0.2029640
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Actitudes hacia las matematicas") %>%
          mutate(`Aplicación` = c("Pre", "Post"))

descriptivos = rbind(descriptivos, desc)
```

```
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Actitudes hacia las matematicas - Total",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

```
pre_post$Afectivo_pre = min_max_scale(pre_post$Afectivo_pre)
pre_post$Afectivo_post = min_max_scale(pre_post$Afectivo_post)
```

Componente afectivo Estadísticos de normalidad

```
print("Estadístico de normalidad pre")
```

```
## [1] "Estadístico de normalidad pre"
```

```
lillie.test(pre_post$Afectivo_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Afectivo_pre
## D = 0.391, p-value <0.0000000000000002
```

```
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```
lillie.test(pre_post$Afectivo_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Afectivo_post
## D = 0.415, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```
summ = data.frame(unclass(psych::describe(pre_post[,c("Afectivo_pre",
                                                      "Afectivo_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Afectivo_pre", "Afectivo_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Afectivo_pre	468	0.82550	0.28723	1	0.88918	0	0	1	1
Afectivo_post	468	0.85399	0.25495	1	0.91312	0	0	1	1

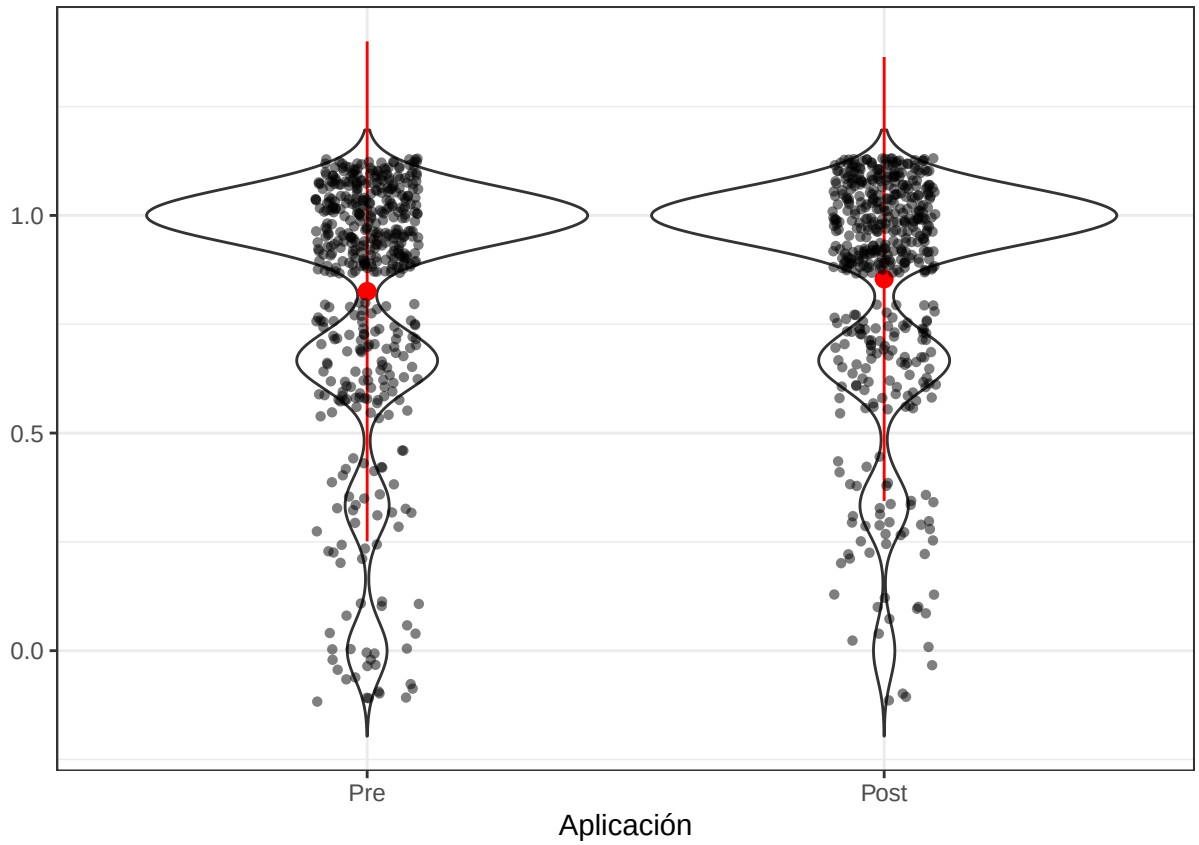
Comparación de medias

```
comparacion =
  wilcox.test(
    x           = pre_post$Afectivo_pre,
    y           = pre_post$Afectivo_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Afectivo_pre and pre_post$Afectivo_post
## V = 6980, p-value = 0.068
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Afectivo_pre, Afectivo_post) %>%
  pivot_longer(cols = c(Afectivo_pre, Afectivo_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/actitudes_matematicas_afectivo.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Afectivo_post, pre_post$Afectivo_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: 0.10472 (negligible)
## 95 percent confidence interval:
##      lower      upper
## -0.0002917  0.2097391
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Actitudes hacia las matematicas - Afectivo") %>%
          mutate(`Aplicación` = c("Pre", "Post"))

descriptivos = rbind(descriptivos, desc)
```

```r
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Actitudes hacia las matematicas - Afectivo",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

```r
pre_post$Cognitivo_pre = min_max_scale(pre_post$Cognitivo_pre)
pre_post$Cognitivo_post = min_max_scale(pre_post$Cognitivo_post)
```

Componente Cognitivo Estadísticos de normalidad

```r
print("Estadístico de normalidad pre")
```

```
## [1] "Estadístico de normalidad pre"
```

```r
lillie.test(pre_post$Cognitivo_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Cognitivo_pre
## D = 0.45, p-value <0.0000000000000002
```

```r
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```r
lillie.test(pre_post$Cognitivo_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Cognitivo_post
## D = 0.46, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```r
summ = data.frame(unclass(psych::describe(pre_post[,c("Cognitivo_pre",
                                                      "Cognitivo_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Cognitivo_pre", "Cognitivo_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Cognitivo_pre	468	0.87322	0.26371	1	0.94326	0	0	1	1
Cognitivo_post	468	0.89174	0.23728	1	0.95124	0	0	1	1

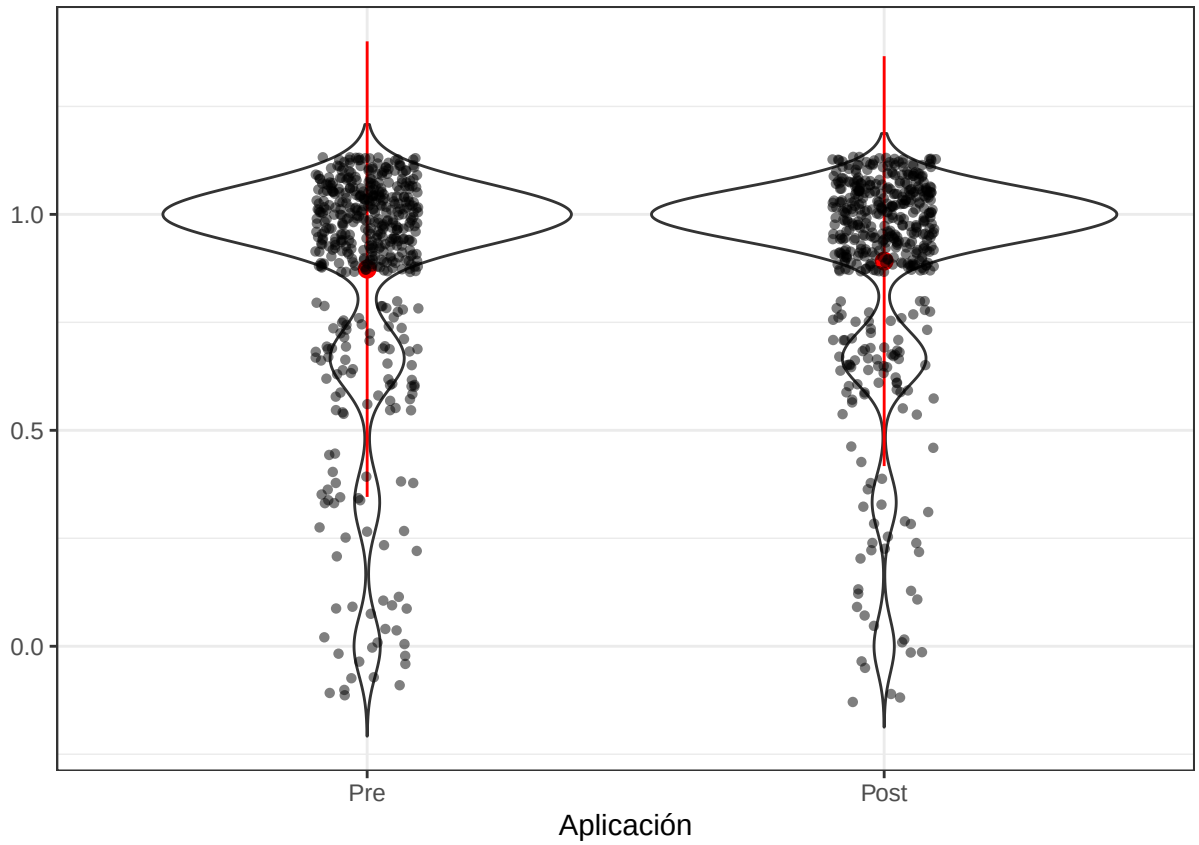
Comparación de medias

```
comparacion =
  wilcox.test(
    x           = pre_post$Cognitivo_pre,
    y           = pre_post$Cognitivo_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Cognitivo_pre and pre_post$Cognitivo_post
## V = 4249, p-value = 0.29
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Cognitivo_pre, Cognitivo_post) %>%
  pivot_longer(cols = c(Cognitivo_pre, Cognitivo_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/actitudes_matematicas_cognitivo.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Cognitivo_post, pre_post$Cognitivo_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: 0.073709 (negligible)
## 95 percent confidence interval:
##    lower    upper
## -0.02891  0.17633
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Actitudes hacia las matematicas - Cognitivo") %>%
          mutate(`Aplicación` = c("Pre", "Post"))

descriptivos = rbind(descriptivos, desc)
```

```r
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Actitudes hacia las matematicas - Cognitivo",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

```r
pre_post$Conativo_pre = min_max_scale(pre_post$Conativo_pre)
pre_post$Conativo_post = min_max_scale(pre_post$Conativo_post)
```

Componente Conativo Estadísticos de normalidad

```r
print("Estadístico de normalidad pre")
```

```
## [1] "Estadístico de normalidad pre"
```

```r
lillie.test(pre_post$Conativo_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Conativo_pre
## D = 0.424, p-value <0.0000000000000002
```

```r
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```r
lillie.test(pre_post$Conativo_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Conativo_post
## D = 0.435, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```r
summ = data.frame(unclass(psych::describe(pre_post[,c("Conativo_pre",
                                                      "Conativo_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Conativo_pre", "Conativo_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Conativo_pre	468	0.84046	0.29054	1	0.90780	0	0	1	1
Conativo_post	468	0.86823	0.25433	1	0.93174	0	0	1	1

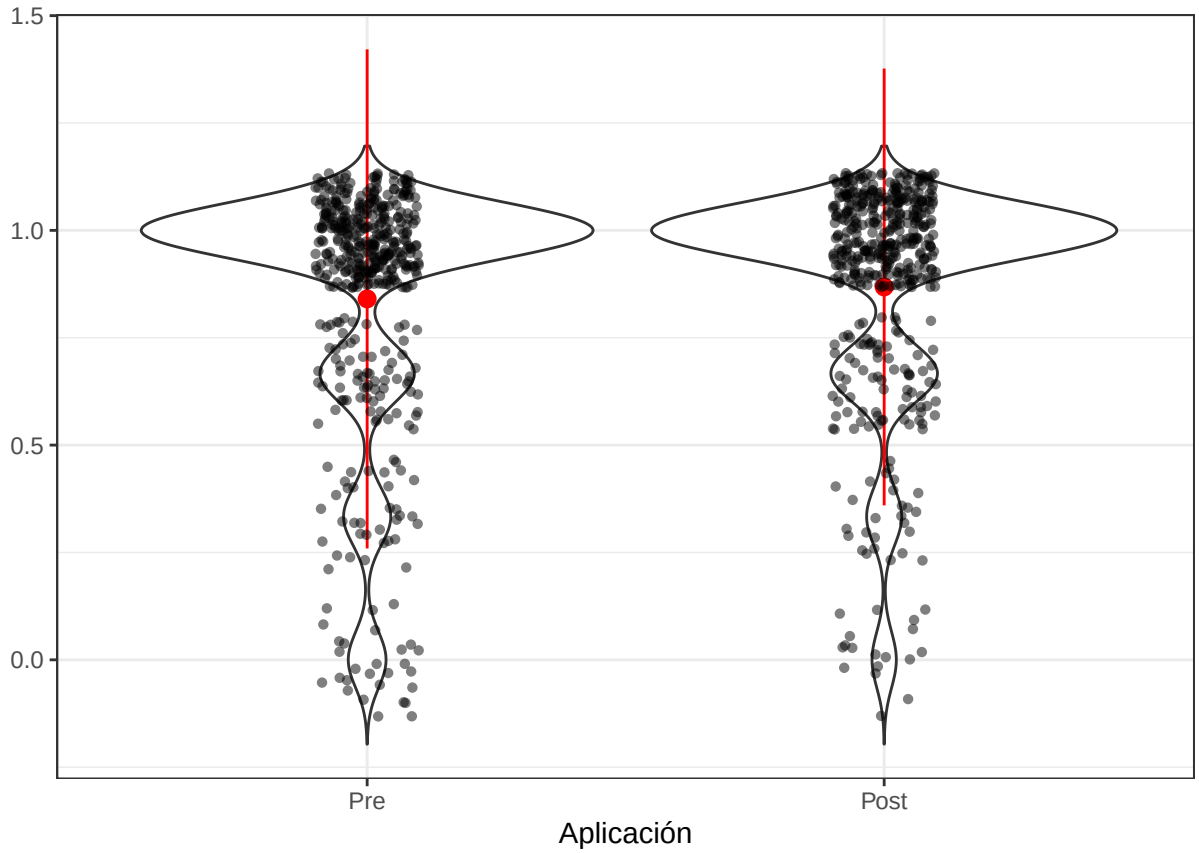
Comparación de medias

```
comparacion =
  wilcox.test(
    x           = pre_post$Conativo_pre,
    y           = pre_post$Conativo_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Conativo_pre and pre_post$Conativo_post
## V = 4972, p-value = 0.052
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Conativo_pre, Conativo_post) %>%
  pivot_longer(cols = c(Conativo_pre, Conativo_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/actitudes_matematicas_conativo.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Conativo_post, pre_post$Conativo_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: 0.10146 (negligible)
## 95 percent confidence interval:
##      lower      upper
## 0.00029882 0.20262527
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Actitudes hacia las matematicas - Conativo") %>%
          mutate(`Aplicación` = c("Pre", "Post"))

descriptivos = rbind(descriptivos, desc)
```

```r
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Actitudes hacia las matematicas - Conativo",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

Motivación

```r
# Motivación Pre
motivacion_pre = read_sheet(url_pre, sheet = "Motivación")
motivacion_pre = motivacion_pre[,1:18]

motivacion_pre = filter(motivacion_pre,
                        !is.na(`Código`),
                        !is.na(`1`))

motivacion_pre[,7:18] = apply(motivacion_pre[,7:18], 2,
                              function(x) str_to_upper(x))

motivacion_pre[,7:18] = apply(motivacion_pre[,7:18], 2,
                              function(x) {ifelse(x == "A", 1, 0)})

# Motivación Post

motivacion_post = read_sheet(url_post, sheet = "Motivación")
motivacion_post = motivacion_post[,1:16]

motivacion_post = filter(motivacion_post,
                        !is.na(`Código`),
                        !is.na(`1`))

motivacion_post[,7:16] = apply(motivacion_post[,7:16], 2,
                               function(x) str_to_upper(x))

claves_motivacion_post = c("B", "A", "A", "B", "B", "B", "B", "B", "A", "A")

calificacion = function(data, claves){
  for(i in 1:dim(data)[2]){
    data[,i] = apply(data[,i], 1,
                     function(x) {ifelse(x == claves[i], 1, 0)})
  }
  data
}

motivacion_post[,7:16] = calificacion(motivacion_post[,7:16], claves_motivacion_post)
```

```r
# Calificación
```

```r
# General

motivacion_pre$Total_pre = apply(motivacion_pre[,7:18], 1,
                                 function(x) sum(x, na.rm= FALSE))

motivacion_post$Total_post = apply(motivacion_post[,7:16], 1,
                                   function(x) sum(x, na.rm= FALSE))

# Interés

motivacion_pre$Interes_pre = apply(motivacion_pre[,c(7,10,13,16)], 1,
                                 function(x) sum(x, na.rm= FALSE))

motivacion_post$Interes_post = apply(motivacion_post[,c(7,10,13)], 1,
                                   function(x) sum(x, na.rm= FALSE))


## Metas

# Orientación al aprendizaje

motivacion_pre$Orientacion_Aprendizaje_pre = apply(motivacion_pre[,c(8,11,14,17)], 1,
                                                   function(x) sum(x, na.rm= FALSE))

motivacion_post$Orientacion_Aprendizaje_post = apply(motivacion_post[,c(8,11,15)], 1,
                                                     function(x) sum(x, na.rm= FALSE))

# Orientación al resultado

motivacion_pre$Orientacion_Resultado_pre = 4 - motivacion_pre$Orientacion_Aprendizaje_pre
motivacion_post$Orientacion_Resultado_post = 3 - motivacion_post$Orientacion_Aprendizaje_post

# Atribución interna

motivacion_pre$Atribucion_interna_pre = apply(motivacion_pre[,c(15,18)], 1,
                                              function(x) sum(x, na.rm= FALSE))

motivacion_post$Atribucion_interna_post = apply(motivacion_post[,c(9,16)], 1,
                                                function(x) sum(x, na.rm= FALSE))

# Atribución externa

motivacion_pre$Atribucion_externa_pre = 2 - motivacion_pre$Atribucion_interna_pre

motivacion_post$Atribucion_externa_post = 2 - motivacion_post$Atribucion_interna_post

# Expectativa

motivacion_pre$Expectativa_pre = apply(motivacion_pre[,c(9,12)], 1,
                                       function(x) sum(x, na.rm= FALSE))

motivacion_post$Expectativa_post = apply(motivacion_post[,c(12,14)], 1,
                                         function(x) sum(x, na.rm= FALSE))
```

```r
columnas_pre = c("Total_pre",
                 "Interes_pre",
                 "Orientacion_Aprendizaje_pre",
                 "Orientacion_Resultado_pre",
                 "Atribucion_interna_pre",
                 "Atribucion_externa_pre",
                 "Expectativa_pre")

columnas_post = c("Total_post",
                 "Interes_post",
                 "Orientacion_Aprendizaje_post",
                 "Orientacion_Resultado_post",
                 "Atribucion_interna_post",
                 "Atribucion_externa_post",
                 "Expectativa_post")

motivacion_pre[,columnas_pre] = apply(motivacion_pre[,columnas_pre],2,
                                      function(x) min_max_scale(x) )

motivacion_post[,columnas_post] = apply(motivacion_post[,columnas_post],2,
                                        function(x) min_max_scale(x) )

pre_post = inner_join(motivacion_post,
                      dplyr::select(motivacion_pre, c("Código", columnas_pre)),
                      by = "Código")
```

Alpha

```r
alpha(motivacion_pre[,7:18])
```

Total pre

```
## [1] 0.79369
```

```r
for(i in seq(length(colnames(motivacion_pre[,7:18])))){
  x = alpha(motivacion_pre[,7:18][,-i])
  print(paste("El índice de confiabilidad cambia a", x,
              "al eliminar el ítem",
              colnames(motivacion_pre[,7:18])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.782468757023904 al eliminar el ítem 1"
## [1] "El índice de confiabilidad cambia a 0.78819576622514 al eliminar el ítem 2"
## [1] "El índice de confiabilidad cambia a 0.777242723784103 al eliminar el ítem 3"
## [1] "El índice de confiabilidad cambia a 0.777254987885137 al eliminar el ítem 4"
## [1] "El índice de confiabilidad cambia a 0.774370799725118 al eliminar el ítem 5"
## [1] "El índice de confiabilidad cambia a 0.775519749049695 al eliminar el ítem 6"
## [1] "El índice de confiabilidad cambia a 0.777894516973956 al eliminar el ítem 7"
## [1] "El índice de confiabilidad cambia a 0.779398242441121 al eliminar el ítem 8"
## [1] "El índice de confiabilidad cambia a 0.782217463765396 al eliminar el ítem 9"
## [1] "El índice de confiabilidad cambia a 0.771353650494053 al eliminar el ítem 10"
## [1] "El índice de confiabilidad cambia a 0.786498739766338 al eliminar el ítem 11"
## [1] "El índice de confiabilidad cambia a 0.777066074058406 al eliminar el ítem 12"
```

```
alpha(motivacion_post[,7:16])
```

Total post

```
## [1] 0.61932
```

```
for(i in seq(length(colnames(motivacion_post[,7:16])))){
  x = alpha(motivacion_post[,7:16][,-i])
  print(paste("El índice de confiabilidad cambia a", x,
              "al eliminar el ítem",
              colnames(motivacion_post[,7:16])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.57606144905541 al eliminar el ítem 1"
## [1] "El índice de confiabilidad cambia a 0.590121273884408 al eliminar el ítem 2"
## [1] "El índice de confiabilidad cambia a 0.613084310423595 al eliminar el ítem 3"
## [1] "El índice de confiabilidad cambia a 0.563507724352878 al eliminar el ítem 4"
## [1] "El índice de confiabilidad cambia a 0.596604790923643 al eliminar el ítem 5"
## [1] "El índice de confiabilidad cambia a 0.587063919017499 al eliminar el ítem 6"
## [1] "El índice de confiabilidad cambia a 0.565355008206158 al eliminar el ítem 7"
## [1] "El índice de confiabilidad cambia a 0.567618138091583 al eliminar el ítem 8"
## [1] "El índice de confiabilidad cambia a 0.628365325537253 al eliminar el ítem 9"
## [1] "El índice de confiabilidad cambia a 0.64797584555563 al eliminar el ítem 10"
```

```
# Amacenamos los valores

alfa = rbind(alfa,
             c("Motivación pre",
               alpha(motivacion_pre[,7:18])))

alfa = rbind(alfa,
             c("Motivación post",
               alpha(motivacion_post[,7:16])))

alfa
```

```
##                       Prueba              Alfa
## 1                       <NA>              <NA>
## 2     Actitudes lenguaje pre 0.801241357662175
## 3    Actitudes lenguaje post 0.795363951952661
## 4  Actitudes matemáticas pre 0.867379859042105
## 5 Actitudes matemáticas post 0.834823761460849
## 6             Motivación pre 0.793691852259175
## 7            Motivación post 0.619322253240666
```

```
alpha(motivacion_pre[,c("1","4","7","10")])
```

Interés pre

```
## [1] 0.66417
```

```
for(i in seq(length(colnames(motivacion_pre[,c("1","4","7","10")])))){
  x = alpha(motivacion_pre[,c("1","4","7","10")][,-i])
  print(paste("El índice de confiabilidad cambia a", x,
              "al eliminar el ítem",
```

```
              colnames(motivacion_pre[,c("1","4","7","10")])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.628148720424915 al eliminar el ítem 1"
## [1] "El índice de confiabilidad cambia a 0.58778116368899 al eliminar el ítem 4"
## [1] "El índice de confiabilidad cambia a 0.57936535648069 al eliminar el ítem 7"
## [1] "El índice de confiabilidad cambia a 0.597143818187136 al eliminar el ítem 10"
```

```
alpha(motivacion_post[,c(7,10,13)])
```

Interés post

```
## [1] 0.66124
```

```
for(i in seq(length(colnames(motivacion_post[,c(7,10,13)])))){
  x = alpha(motivacion_post[,c(7,10,13)][,-i])
  print(paste("El índice de confiabilidad cambia a", x,
              "al eliminar el ítem",
              colnames(motivacion_post[,c(7,10,13)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.653650308499103 al eliminar el ítem 1"
## [1] "El índice de confiabilidad cambia a 0.506022640049801 al eliminar el ítem 4"
## [1] "El índice de confiabilidad cambia a 0.541195957862625 al eliminar el ítem 7"
```

```
alpha(motivacion_pre[,c("2","5","8","11")])
```

Metas pre

```
## [1] 0.55185
```

```
for(i in seq(length(colnames(motivacion_pre[,c("2","5","8","11")])))){
  x = alpha(motivacion_pre[,c("2","5","8","11")][,-i])
  print(paste("El índice de confiabilidad cambia a", x,
              "al eliminar el ítem",
              colnames(motivacion_pre[,c("2","5","8","11")])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.498833121907289 al eliminar el ítem 2"
## [1] "El índice de confiabilidad cambia a 0.486262273143154 al eliminar el ítem 5"
## [1] "El índice de confiabilidad cambia a 0.464445929493966 al eliminar el ítem 8"
## [1] "El índice de confiabilidad cambia a 0.470984867049863 al eliminar el ítem 11"
```

```
alpha(motivacion_post[,c(8,11,15)])
```

Metas post

```
## [1] 0.28162
```

```
for(i in seq(length(colnames(motivacion_post[,c(8,11,15)])))){
  x = alpha(motivacion_post[,c(8,11,15)][,-i])
  print(paste("El índice de confiabilidad cambia a", x,
              "al eliminar el ítem",
              colnames(motivacion_post[,c(8,11,15)])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.201963896933784 al eliminar el ítem 2"
## [1] "El índice de confiabilidad cambia a 0.32108343696875 al eliminar el ítem 5"
## [1] "El índice de confiabilidad cambia a 0.0621751167934566 al eliminar el ítem 9"
```

Expectativas Este sólo está en el pre, ya que en el post la escala se subdividió

```
alpha(motivacion_pre[,c("3","6","9","12")])
```

```
## [1] 0.60279
```

```
for(i in seq(length(colnames(motivacion_pre[,c("3","6","9","12")])))){
  x = alpha(motivacion_pre[,c("3","6","9","12")][,-i])
  print(paste("El índice de confiabilidad cambia a", x,
              "al eliminar el ítem",
              colnames(motivacion_pre[,c("3","6","9","12")])[i]))
}
```

```
## [1] "El índice de confiabilidad cambia a 0.525780524891882 al eliminar el ítem 3"
## [1] "El índice de confiabilidad cambia a 0.513256556391975 al eliminar el ítem 6"
## [1] "El índice de confiabilidad cambia a 0.550812435675047 al eliminar el ítem 9"
## [1] "El índice de confiabilidad cambia a 0.54186286050618 al eliminar el ítem 12"
```

```
alpha(motivacion_post[,c(9,16)])
```

Atribución interna

```
## [1] 0.10725
```

No puede obtener el indicador al eliminar un ítem, ya que si queda un único ítem el indicador no tiene solución.

```
alpha(motivacion_post[,c(12,14)])
```

Expectativa positiva

```
## [1] 0.56704
```

Indicadores psicométricos

```
analitem = item.exam(motivacion_pre[,7:18], discrim=T)
analitem = dplyr::select(analitem, -c("Item.Criterion", "Item.Validity"))
cols_num = colnames(analitem)
analitem$Item = row.names(analitem)
analitem %>%
  gt(rowname_col = "Item") %>%
  fmt_number(
    columns = cols_num,
    decimals = 3
  ) %>%
  tab_style(
    style = list(
      cell_text(align="center", weight="bold")
    ),
    locations=cells_column_labels()
```

```
  )%>%
  tab_style(
    style = list(
      cell_text(align="center")
    ),
    locations = cells_body())
```

Total pre

	Sample.SD	Item.total	Item.Tot.woi	Difficulty	Discrimination	Item.Reliab	Item.Rel.woi
1	0.346	0.516	0.399	0.862	0.321	0.178	0.138
2	0.462	0.536	0.377	0.691	0.603	0.248	0.174
3	0.337	0.564	0.456	0.870	0.326	0.190	0.154
4	0.269	0.566	0.482	0.921	0.217	0.152	0.130
5	0.312	0.591	0.496	0.890	0.304	0.185	0.155
6	0.326	0.579	0.478	0.880	0.310	0.188	0.156
7	0.279	0.556	0.468	0.915	0.220	0.155	0.131
8	0.390	0.560	0.433	0.813	0.465	0.219	0.169
9	0.388	0.536	0.406	0.815	0.429	0.208	0.158
10	0.310	0.622	0.533	0.892	0.310	0.193	0.165
11	0.449	0.539	0.386	0.719	0.590	0.242	0.173
12	0.418	0.590	0.458	0.775	0.535	0.246	0.191

```
# Guardamos los indicadores importantes

psicometricos_pre = analitem %>%
  dplyr::select(c("Difficulty", "Discrimination")) %>%
  mutate(Prueba = "Motivación - Pre") %>%
  mutate(Item = row.names(.))

indicadores_psicometricos = rbind(indicadores_psicometricos,
                                  psicometricos_pre)
```

```
analitem = item.exam(motivacion_post[,7:16], discrim=T)
analitem = dplyr::select(analitem, -c("Item.Criterion", "Item.Validity"))
cols_num = colnames(analitem)
analitem$Item = row.names(analitem)
analitem %>%
  gt(rowname_col = "Item") %>%
  fmt_number(
    columns = cols_num,
    decimals = 3
  ) %>%
  tab_style(
    style = list(
      cell_text(align="center", weight="bold")
    ),
    locations=cells_column_labels()
  )%>%
  tab_style(
    style = list(
```

```
      cell_text(align="center")
    ),
    locations = cells_body())
```

Total post

	Sample.SD	Item.total	Item.Tot.woi	Difficulty	Discrimination	Item.Reliab	Item.Rel.woi
1	0.366	0.534	0.378	0.840	0.353	0.196	0.138
2	0.427	0.498	0.306	0.761	0.488	0.213	0.130
3	0.462	0.438	0.217	0.692	0.485	0.202	0.100
4	0.317	0.592	0.469	0.886	0.294	0.188	0.149
5	0.431	0.477	0.279	0.754	0.444	0.206	0.120
6	0.366	0.489	0.325	0.841	0.309	0.179	0.119
7	0.321	0.582	0.456	0.883	0.300	0.187	0.146
8	0.379	0.568	0.412	0.827	0.400	0.215	0.156
9	0.492	0.409	0.168	0.592	0.526	0.201	0.082
10	0.475	0.322	0.081	0.658	0.376	0.153	0.038

```
# Guardamos los indicadores importantes

psicometricos_pre = analitem %>%
  dplyr::select(c("Difficulty", "Discrimination")) %>%
  mutate(Prueba = "Motivación - Post") %>%
  mutate(Item = row.names(.))

indicadores_psicometricos = rbind(indicadores_psicometricos,
                                  psicometricos_pre)
```

Comparación pre-post

Prueba total

```
print("Estadístico de normalidad pre")
```

Estadísticos de normalidad

```
## [1] "Estadístico de normalidad pre"
```

```
lillie.test(pre_post$Total_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Total_pre
## D = 0.226, p-value <0.0000000000000002
```

```
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```
lillie.test(pre_post$Total_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
```

```
##
## data:  pre_post$Total_post
## D = 0.19, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```
summ = data.frame(unclass(psych::describe(pre_post[,c("Total_pre",
                                                      "Total_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Total_pre", "Total_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Total_pre	877	0.83799	0.20253	0.91667	0.87624	0.12355	0	1	1
Total_post	877	0.77913	0.19121	0.80000	0.80299	0.14826	0	1	1

Comparación de medias

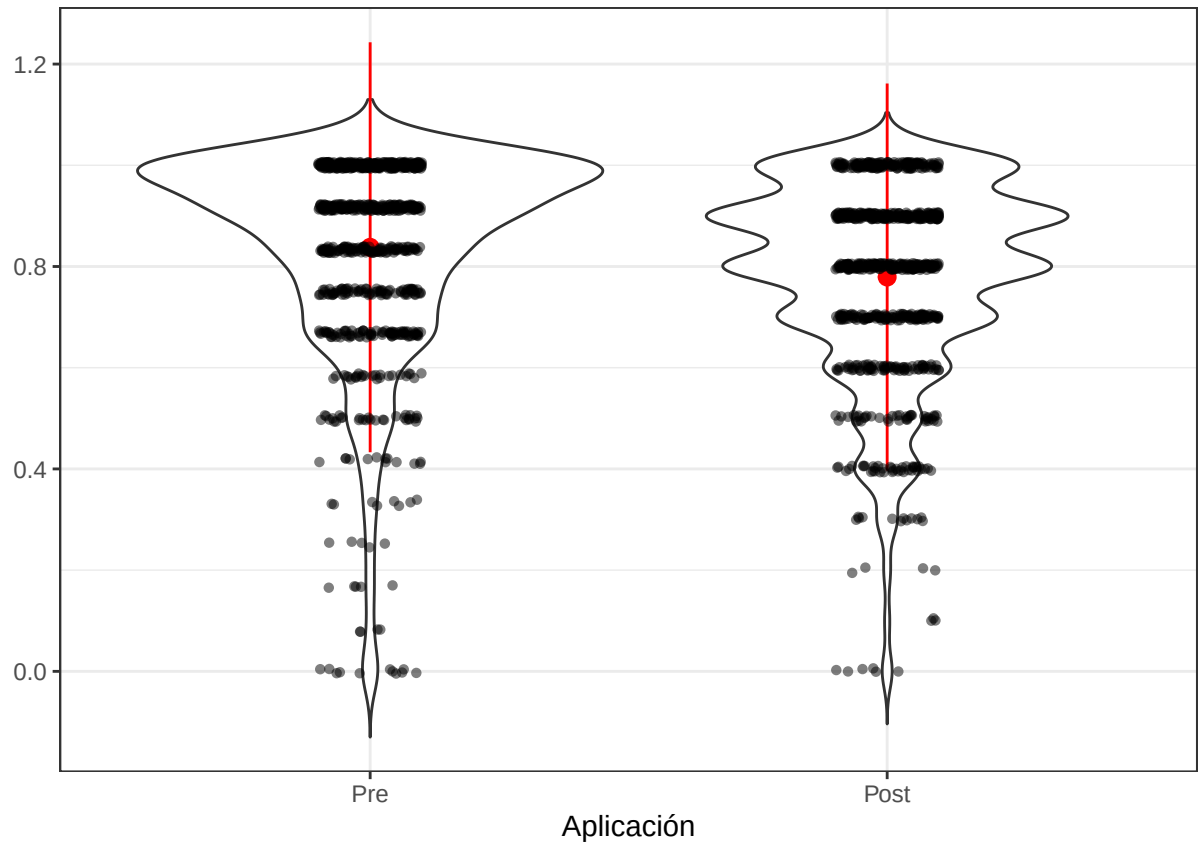
```
comparacion =
  wilcox.test(
    x           = pre_post$Total_pre,
    y           = pre_post$Total_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Total_pre and pre_post$Total_post
## V = 211722, p-value <0.0000000000000002
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Total_pre, Total_post) %>%
  pivot_longer(cols = c(Total_pre, Total_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
```

```
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/motivacion.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Total_post, pre_post$Total_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: -0.29874 (small)
## 95 percent confidence interval:
##    lower    upper
## -0.38085 -0.21664
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Motivación") %>%
          mutate(`Aplicación` = c("Pre", "Post"))
```

```
descriptivos = rbind(descriptivos, desc)
```

```
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Motivación - Total",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

Interés

```
print("Estadístico de normalidad pre")
```

Estadísticos de normalidad

```
## [1] "Estadístico de normalidad pre"
```

```
lillie.test(pre_post$Interes_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Interes_pre
## D = 0.442, p-value <0.0000000000000002
```

```
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```
lillie.test(pre_post$Interes_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Interes_post
## D = 0.442, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```
summ = data.frame(unclass(psych::describe(pre_post[,c("Interes_pre",
                                                      "Interes_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Interes_pre", "Interes_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Interes_pre	877	0.90080	0.21267	1	0.95697	0	0	1	1

Interes_post	877	0.87419	0.25336	1	0.93789	0	0	1	1

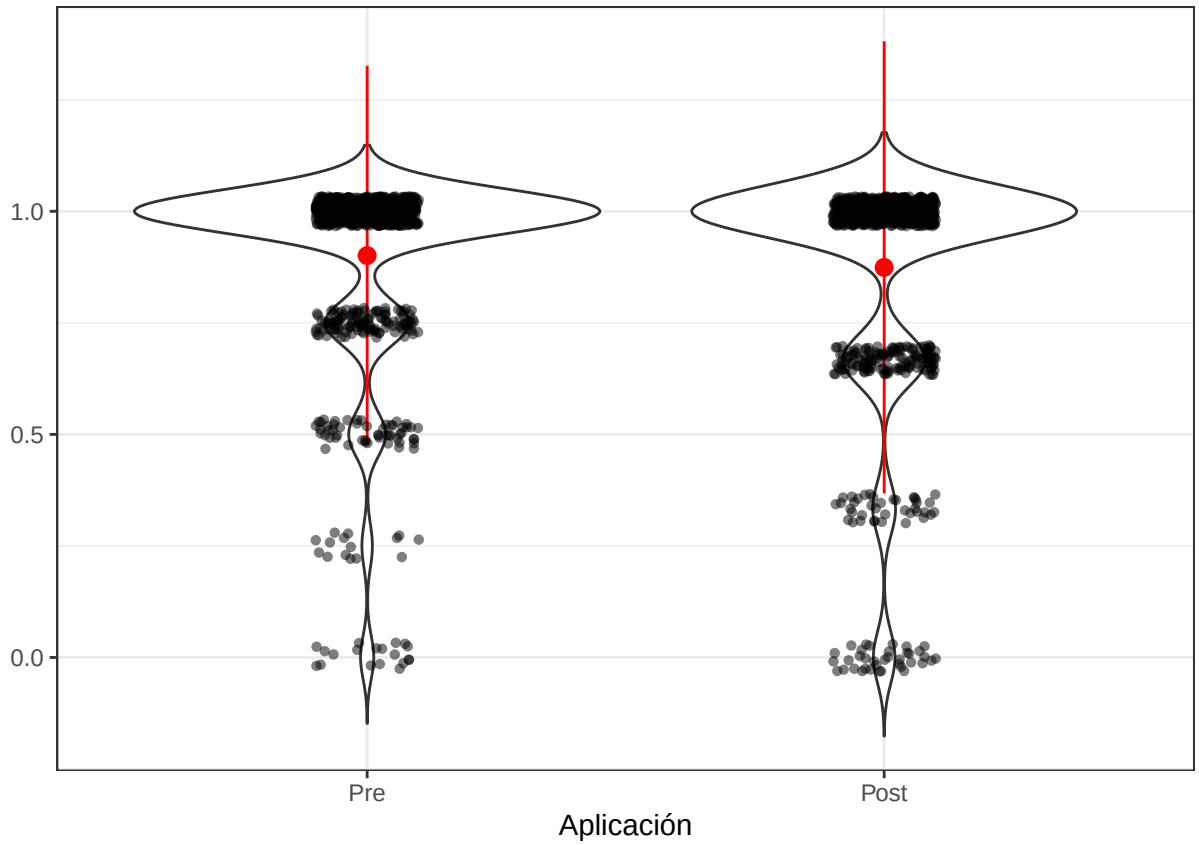
Comparación de medias

```
comparacion =
  wilcox.test(
    x           = pre_post$Interes_pre,
    y           = pre_post$Interes_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Interes_pre and pre_post$Interes_post
## V = 37137, p-value = 0.00012
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Interes_pre, Interes_post) %>%
  pivot_longer(cols = c(Interes_pre, Interes_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/motivacion_interes.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Interes_post, pre_post$Interes_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: -0.1135 (negligible)
## 95 percent confidence interval:
##     lower     upper
## -0.196102 -0.030892
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Motivación - Interés") %>%
          mutate(`Aplicación` = c("Pre", "Post"))

descriptivos = rbind(descriptivos, desc)
```

```r
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Motivación - Interés",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

Metas

```r

print("Estadístico de normalidad pre")
```

Estadísticos de normalidad

```
## [1] "Estadístico de normalidad pre"
```

```r
lillie.test(pre_post$Orientacion_Resultado_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Orientacion_Resultado_pre
## D = 0.282, p-value <0.0000000000000002
```

```r
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```r
lillie.test(pre_post$Orientacion_Resultado_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Orientacion_Resultado_post
## D = 0.241, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```r
summ = data.frame(unclass(psych::describe(pre_post[,c("Orientacion_Resultado_pre",
                                                      "Orientacion_Resultado_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Orientacion_Resultado_pre", "Orientacion_Resultado_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Orientacion_Resultado_pre	877	0.22121	0.26889	0.25000	0.17496	0.37065	0	1	1
Orientacion_Resultado_post	877	0.29304	0.28740	0.33333	0.26458	0.49420	0	1	1

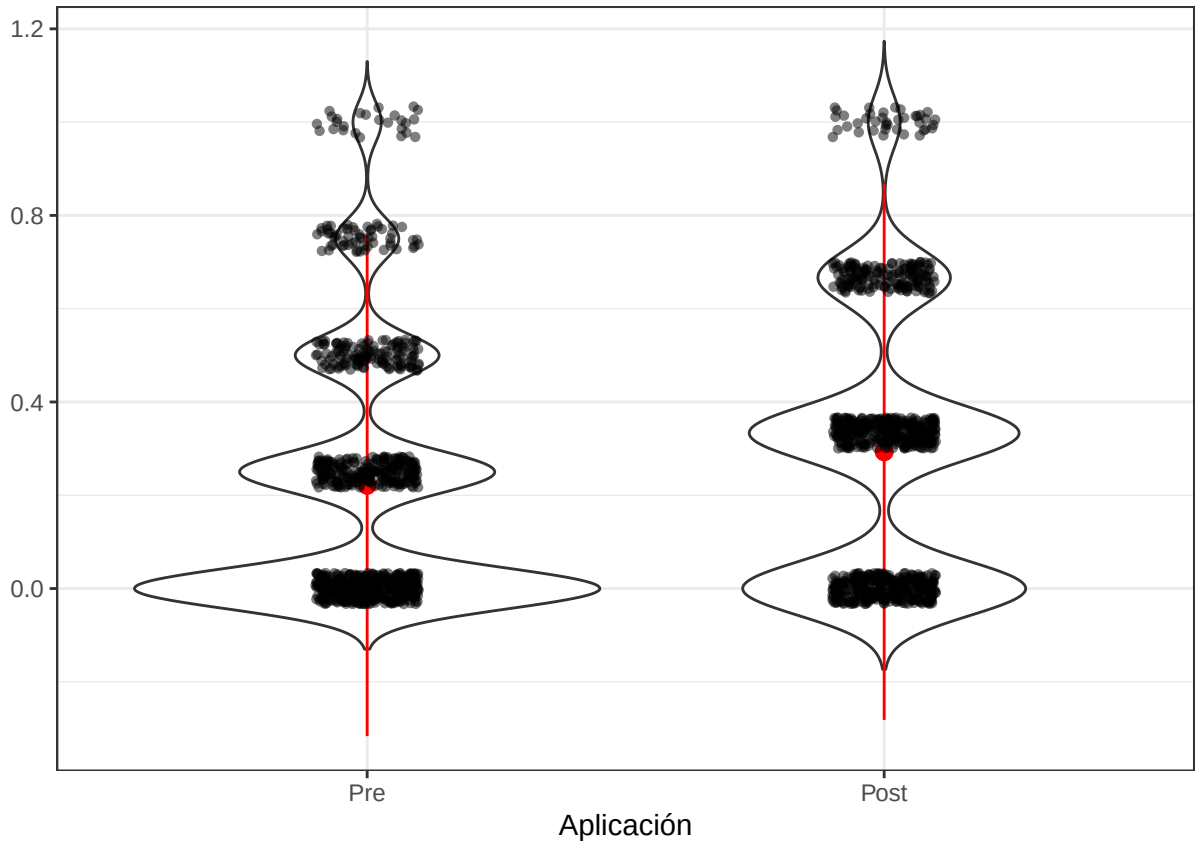
Comparación de medias

```
comparacion =
  wilcox.test(
    x           = pre_post$Orientacion_Resultado_pre,
    y           = pre_post$Orientacion_Resultado_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Orientacion_Resultado_pre and pre_post$Orientacion_Resultado_post
## V = 85650, p-value = 0.000000035
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Orientacion_Resultado_pre, Orientacion_Resultado_post) %>%
  pivot_longer(cols = c(Orientacion_Resultado_pre, Orientacion_Resultado_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/motivacion_metas_resultado.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Orientacion_Resultado_post,
          pre_post$Orientacion_Resultado_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: 0.25806 (small)
## 95 percent confidence interval:
##   lower   upper
## 0.17227 0.34385
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Motivación - Metas") %>%
          mutate(`Aplicación` = c("Pre", "Post"))

descriptivos = rbind(descriptivos, desc)
```

```
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Motivación - Metas",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

```
pre_post$Atribucion_externa_pre = 1 - pre_post$Atribucion_interna_pre
pre_post$Atribucion_externa_post = 1 - pre_post$Atribucion_interna_post
```

Atribución externa

```
print("Estadístico de normalidad pre")
```

Estadísticos de normalidad

```
## [1] "Estadístico de normalidad pre"
```

```
lillie.test(pre_post$Atribucion_externa_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Atribucion_externa_pre
## D = 0.417, p-value <0.0000000000000002
```

```
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```
lillie.test(pre_post$Atribucion_externa_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Atribucion_externa_post
## D = 0.308, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```
summ = data.frame(unclass(psych::describe(pre_post[,c("Atribucion_externa_pre",
                                                      "Atribucion_externa_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Atribucion_externa_pre", "Atribucion_externa_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Atribucion_externa_pre	877	0.20125	0.31915	0.0	0.13727	0.0000	0	1	1
Atribucion_externa_post	877	0.31186	0.33610	0.5	0.26529	0.7413	0	1	1

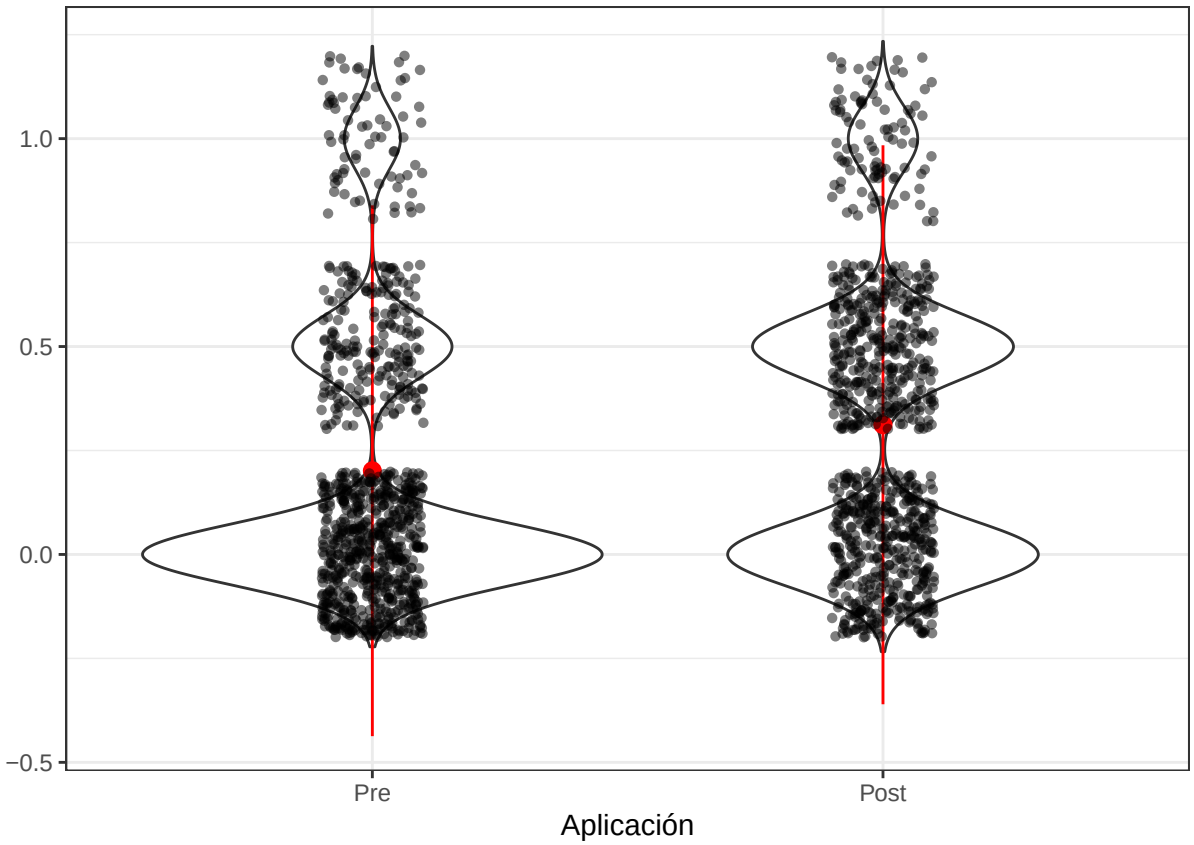
Comparación de medias

```
comparacion =
  wilcox.test(
    x           = pre_post$Atribucion_externa_pre,
    y           = pre_post$Atribucion_externa_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Atribucion_externa_pre and pre_post$Atribucion_externa_post
## V = 33132, p-value = 0.00000000000032
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Atribucion_externa_pre, Atribucion_externa_post) %>%
  pivot_longer(cols = c(Atribucion_externa_pre, Atribucion_externa_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/motivacion_atribucion_externa.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Atribucion_externa_post,
          pre_post$Atribucion_externa_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: 0.33744 (small)
## 95 percent confidence interval:
##   lower   upper
## 0.24859 0.42629
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Motivación - Atribución externa") %>%
          mutate(`Aplicación` = c("Pre", "Post"))

descriptivos = rbind(descriptivos, desc)
```

```r
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Motivación - Atribución externa",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```

Expectativas positivas

```r
print("Estadístico de normalidad pre")
```

Estadísticos de normalidad

```
## [1] "Estadístico de normalidad pre"
```

```r
lillie.test(pre_post$Expectativa_pre)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Expectativa_pre
## D = 0.477, p-value <0.0000000000000002
```

```r
print("Estadístico de normalidad post")
```

```
## [1] "Estadístico de normalidad post"
```

```r
lillie.test(pre_post$Expectativa_post)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  pre_post$Expectativa_post
## D = 0.455, p-value <0.0000000000000002
```

Debido a que no existe normalidad en las puntuaciones, se usará una prueba no paramétrica

Descriptivos

```r
summ = data.frame(unclass(psych::describe(pre_post[,c("Expectativa_pre",
                                                      "Expectativa_post")])),
                  check.names = FALSE, stringsAsFactors = FALSE)

summ$vars = c("Expectativa_pre", "Expectativa_post")

summ[,1:10] %>%
  gt()
```

vars	n	mean	sd	median	trimmed	mad	min	max	range
Expectativa_pre	877	0.87001	0.27486	1	0.93385	0	0	1	1
Expectativa_post	877	0.83580	0.30940	1	0.90754	0	0	1	1

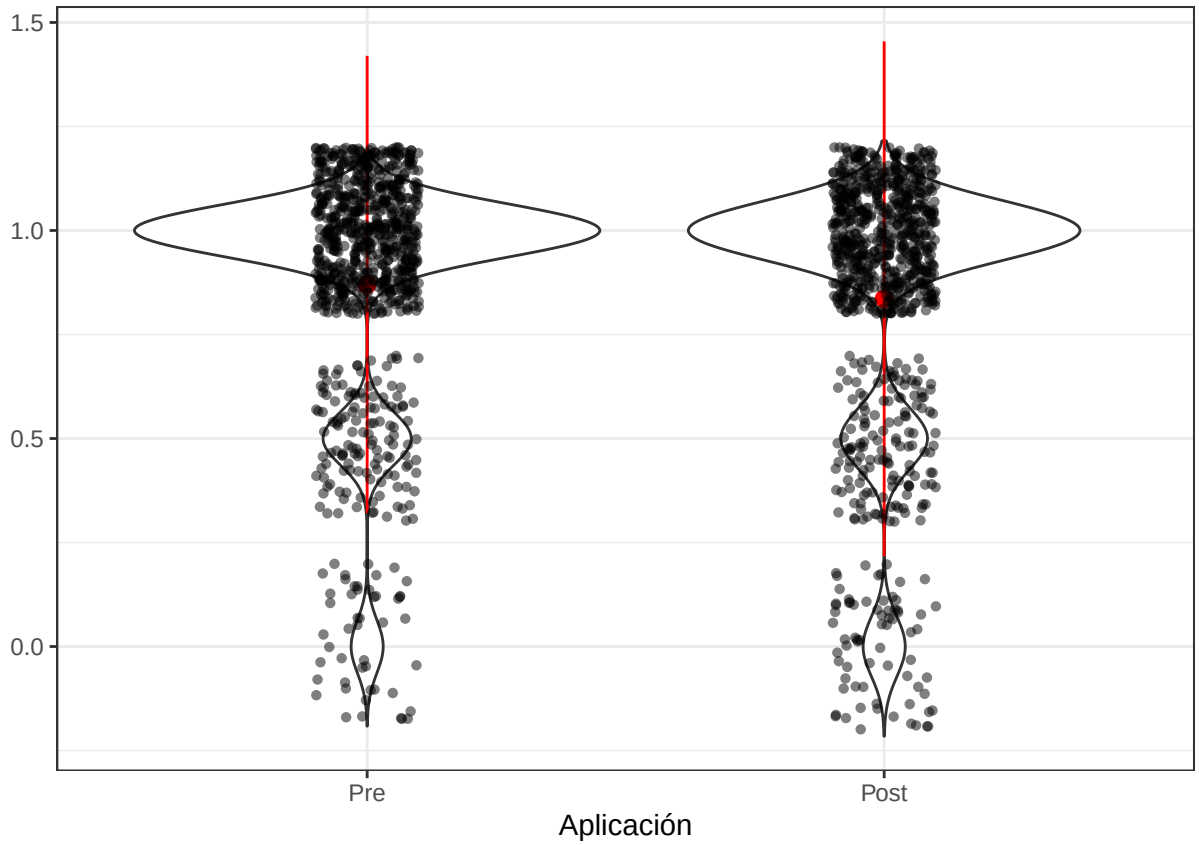
Comparación de medias

```
comparacion =
  wilcox.test(
    x           = pre_post$Expectativa_pre,
    y           = pre_post$Expectativa_post,
    alternative = "two.sided",
    mu          = 0,
    var.equal   = TRUE,
    paired      = TRUE,
    conf.level  = 0.95
  )

comparacion
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pre_post$Expectativa_pre and pre_post$Expectativa_post
## V = 22736, p-value = 0.0086
## alternative hypothesis: true location shift is not equal to 0
```

```
pre_post %>%
  dplyr::select(Expectativa_pre, Expectativa_post) %>%
  pivot_longer(cols = c(Expectativa_pre, Expectativa_post),
               values_to = "value",
               names_to = "Aplicación") %>%
  ggplot(aes(x = reorder(`Aplicación`, desc(`Aplicación`)), y = value)) +
  geom_violin(trim=FALSE) +
  theme_bw() +
  theme(legend.position = "none") +
  xlab("Aplicación") + ylab("")+
  stat_summary(fun.data=mean_sdl, geom="pointrange", color="red") +
  geom_jitter(shape=16, position=position_jitter(0.1), alpha=0.5) +
  scale_x_discrete(labels = c("Pre","Post"))
```




```
# Guardamos la imagen

ggsave("../Plots/motivacion_expectativas.png")
```

Tamaño del efecto

```
size_effect =
  cohen.d(pre_post$Expectativa_post,
          pre_post$Expectativa_pre, paired = TRUE)

size_effect
```

```
##
## Cohen's d
##
## d estimate: -0.11681 (negligible)
## 95 percent confidence interval:
##     lower     upper
## -0.202698 -0.030928
```

```
# Separamos los descriptivos importantes
desc = summ[,1:10] %>%
        dplyr::select(c("n", "mean", "sd", "min", "max")) %>%
          mutate(Prueba = "Motivación - Expectativas") %>%
          mutate(`Aplicación` = c("Pre", "Post"))

descriptivos = rbind(descriptivos, desc)
```

```r
# Guardamos los resultados de esta comparación

data_temp =
  data.frame(Prueba = "Motivación - Expectativas",
             `Media pre` = summ$mean[1],
             `Media pro` = summ$mean[2],
             `p value` = comparacion$p.value,
             `D de cohen` = size_effect$magnitude)

comparaciones = rbind(comparaciones, data_temp)
```