# Reproducible Research - Notas de clase

*Daniel Czarnievicz*

*Curso 2018*

## Clase 1 (20/8/2018)

### Porqué Reproducible research?

reproducibilidad != replicabilidad Reproducibilidad = mismos datos -> mismo resultados replicabilidad = aplicar a otro experimento con mismo resultado global (la vacuna siempre es efectividad)

crisis de replicabilidad -> problema de los p valores. entender los p valores como variables aleatorias

packrat y docker guardan todo el trabajo en un "pack" de R

### R Markdown

`echo = FALSE` parameter prevent printing of the R code.
`eval = FALSE` no evalua el c?digo.
`cache = TRUE` guarda resultados del chunk en el cache y no vuelve a generarlo.

```
dat = data.frame(x=rnorm(100), y=rnorm(100))
ggplot(dat, aes(x,y)) +
      geom_point()
```

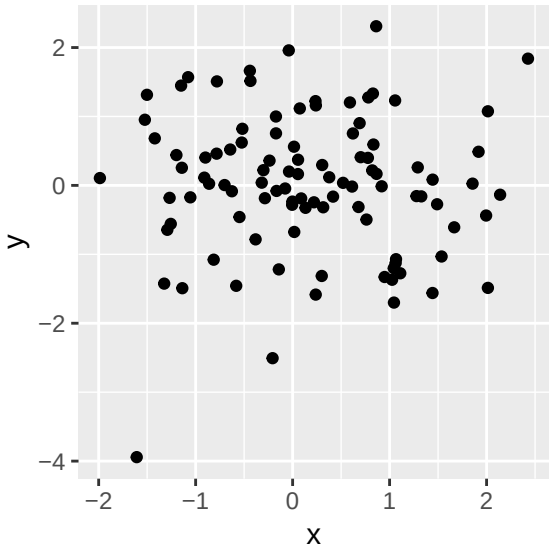



Figure 1: Puntos aleatorios

`xtable` para compilar pdf.

# Clase 2 (27/8/2018)

## Projectos en RStudio

- Organiza el trabajo en un mismo directorio.
- RStudio permite cambiar el directorio sin que se "rompa todo".
- El directorio de trabajo es el projecto mismo.
- Usar la librería "here".

```
# instalar librería here
library(here)
```

# Clase 3 (10/9/2018)

## Operador pipe %>%

$f(x, y, z)$ igual a $x$ %>% $f(y, z)$

```
x = c(1:10, 50)
mean(x, trim=.1)
```

```
## [1] 6
```

```
x %>% mean(trim=.1)
```

```
## [1] 6
```

```
0.1 %>% mean(x, .)
```

```
## [1] 6
```

```
res = transform(aggregate(. ~ cyl,
                          data=subset(mtcars, hp>100,
                                      select=c("mpg","cyl")),
                          FUN=function(x) round(mean(x), 2)),
                kpl=mpg*0.4251)
```

```
res <- mtcars %>%
      subset(hp > 100, select=c("mpg","cyl")) %>%
      aggregate(.~cyl, data=., FUN=function(x) round(mean(x), 2)) %>%
      transform(kpl = mpg*0.4251)
```

```
mtcars %>%
      filter(hp > 100) %>%
      dplyr::select(cyl, mpg) %>%
      group_by(cyl) %>%
      summarise(mpg = mean(mpg)) %>%
      mutate(kpl = mpg*0.4251)
```

```
## # A tibble: 3 x 3
##     cyl   mpg   kpl
##   <dbl> <dbl> <dbl>
## 1     4  25.9 11.0
## 2     6  19.7  8.39
## 3     8  15.1  6.42
```

# Clase 4 (17/9/2018)

## tidyverse

```
mtcars = as_tibble(mtcars)
# verbo filter: filtra seg{un condiciones de las filas}
mtcars %>% filter(mpg > 22)
```

```
## # A tibble: 9 x 11
##     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  22.8     4 108      93  3.85  2.32  18.6     1     1     4     1
## 2  24.4     4 147.     62  3.69  3.19  20       1     0     4     2
## 3  22.8     4 141.     95  3.92  3.15  22.9     1     0     4     2
## 4  32.4     4  78.7    66  4.08  2.2   19.5     1     1     4     1
## 5  30.4     4  75.7    52  4.93  1.62  18.5     1     1     4     2
## 6  33.9     4  71.1    65  4.22  1.84  19.9     1     1     4     1
## 7  27.3     4  79      66  4.08  1.94  18.9     1     1     4     1
## 8  26       4 120.     91  4.43  2.14  16.7     0     1     5     2
## 9  30.4     4  95.1   113  3.77  1.51  16.9     1     1     5     2
```

```
filter(mtcars, mpg == 24.4 & gear == 4)
```

```
## # A tibble: 1 x 11
##     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  24.4     4  147.    62  3.69  3.19    20     1     0     4     2
```

```
filter(mtcars, mpg == 24.4 | mpg == 22.8)
```

```
## # A tibble: 3 x 11
##     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  22.8     4  108     93  3.85  2.32  18.6     1     1     4     1
## 2  24.4     4  147.    62  3.69  3.19  20       1     0     4     2
## 3  22.8     4  141.    95  3.92  3.15  22.9     1     0     4     2
```

```
# arrange: ordena las filas (menor a mayor)
arrange(mtcars, mpg)
```

```
## # A tibble: 32 x 11
##      mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1  10.4     8  472    205  2.93  5.25  18.0     0     0     3     4
##  2  10.4     8  460    215  3     5.42  17.8     0     0     3     4
##  3  13.3     8  350    245  3.73  3.84  15.4     0     0     3     4
##  4  14.3     8  360    245  3.21  3.57  15.8     0     0     3     4
##  5  14.7     8  440    230  3.23  5.34  17.4     0     0     3     4
##  6  15       8  301    335  3.54  3.57  14.6     0     1     5     8
##  7  15.2     8  276.   180  3.07  3.78  18       0     0     3     3
##  8  15.2     8  304    150  3.15  3.44  17.3     0     0     3     2
##  9  15.5     8  318    150  2.76  3.52  16.9     0     0     3     2
## 10  15.8     8  351    264  4.22  3.17  14.5     0     1     5     4
## # ... with 22 more rows
```

```
# desc() ordena las filaes (mayor a menor)
arrange(mtcars, desc(mpg))
```

```
## # A tibble: 32 x 11
##      mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1  33.9     4  71.1    65  4.22  1.84  19.9     1     1     4     1
##  2  32.4     4  78.7    66  4.08  2.2   19.5     1     1     4     1
##  3  30.4     4  75.7    52  4.93  1.62  18.5     1     1     4     2
##  4  30.4     4  95.1   113  3.77  1.51  16.9     1     1     5     2
##  5  27.3     4  79      66  4.08  1.94  18.9     1     1     4     1
##  6  26       4 120.     91  4.43  2.14  16.7     0     1     5     2
##  7  24.4     4 147.     62  3.69  3.19  20       1     0     4     2
##  8  22.8     4 108      93  3.85  2.32  18.6     1     1     4     1
##  9  22.8     4 141.     95  3.92  3.15  22.9     1     0     4     2
## 10  21.5     4 120.     97  3.7   2.46  20.0     1     0     3     1
## # ... with 22 more rows
```

```
# select: selecciona variable
# puede utilizar funciones selectoras para matchear expresiones regulares
# starts_with(), ends_with(), etc...
dplyr::select(mtcars, ends_with("t"))
```

```
## # A tibble: 32 x 2
##     drat    wt
##    <dbl> <dbl>
##  1  3.9   2.62
##  2  3.9   2.88
##  3  3.85  2.32
##  4  3.08  3.22
##  5  3.15  3.44
##  6  2.76  3.46
##  7  3.21  3.57
##  8  3.69  3.19
##  9  3.92  3.15
## 10  3.92  3.44
## # ... with 22 more rows
```

```r
dplyr::select(mtcars, wt:gear)
```

```
## # A tibble: 32 x 5
##       wt  qsec    vs    am  gear
##    <dbl> <dbl> <dbl> <dbl> <dbl>
##  1  2.62  16.5     0     1     4
##  2  2.88  17.0     0     1     4
##  3  2.32  18.6     1     1     4
##  4  3.22  19.4     1     0     3
##  5  3.44  17.0     0     0     3
##  6  3.46  20.2     1     0     3
##  7  3.57  15.8     0     0     3
##  8  3.19  20       1     0     4
##  9  3.15  22.9     1     0     4
## 10  3.44  18.3     1     0     4
## # ... with 22 more rows
```

```r
dplyr::select(mtcars, -(wt:gear))
```

```
## # A tibble: 32 x 6
##      mpg   cyl  disp    hp  drat  carb
##    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1  21       6  160    110  3.9      4
##  2  21       6  160    110  3.9      4
##  3  22.8     4  108     93  3.85     1
##  4  21.4     6  258    110  3.08     1
##  5  18.7     8  360    175  3.15     2
##  6  18.1     6  225    105  2.76     1
##  7  14.3     8  360    245  3.21     4
##  8  24.4     4  147.    62  3.69     2
##  9  22.8     4  141.    95  3.92     2
## 10  19.2     6  168.   123  3.92     4
## # ... with 22 more rows
```

```r
# mutate: crea nueva variable
# transmute: me quedo solo con las variables nuevas
```

```r
# group_by: agrupa segun variables
#     una fila para cada grupo de la variable agrupadora
# summarize: crea resumenes
mtcars %>%
      group_by(cyl) %>%
      tally()
```

```
## # A tibble: 3 x 2
##     cyl     n
##   <dbl> <int>
## 1     4    11
## 2     6     7
## 3     8    14
```

### Ejercicio de clase

```
# 1
mpg %>%
      dplyr::select(manufacturer, model, year, cyl, cty) %>%
      dplyr::filter(manufacturer == "toyota" & model == "camry")
```

```
## # A tibble: 7 x 5
##   manufacturer model  year   cyl   cty
##   <chr>        <chr> <int> <int> <int>
## 1 toyota       camry  1999     4    21
## 2 toyota       camry  1999     4    21
## 3 toyota       camry  2008     4    21
## 4 toyota       camry  2008     4    21
## 5 toyota       camry  1999     6    18
## 6 toyota       camry  1999     6    18
## 7 toyota       camry  2008     6    19
```

```
# 2
mpg %>%
      group_by(manufacturer) %>%
      summarise(rend.prom = mean(cty)) %>%
      filter(rend.prom == min(.$rend.prom) | rend.prom == max(.$rend.prom)) %>%
      dplyr::select(manufacturer)
```

```
## # A tibble: 2 x 1
##   manufacturer
##   <chr>
## 1 honda
## 2 lincoln
```

```
# 3
mpg %>%
      group_by(manufacturer) %>%
      summarise(cty.mean = mean(cty),
                sd.mean = sd(cty)/sqrt(n()),
                rend.rg = max(cty) - min(cty))
```

```
## # A tibble: 15 x 4
##    manufacturer cty.mean sd.mean rend.rg
##    <chr>           <dbl>   <dbl>   <int>
##  1 audi             17.6   0.465       6
##  2 chevrolet        15     0.671      11
##  3 dodge            13.1   0.409       9
##  4 ford             14     0.383       7
##  5 honda            24.4   0.648       7
##  6 hyundai          18.6   0.401       5
##  7 jeep             13.5   0.886       8
##  8 land rover       11.5   0.289       1
##  9 lincoln          11.3   0.333       1
## 10 mercury          13.2   0.25        1
```

```
## 11 nissan           18.1   0.950      11
## 12 pontiac          17     0.447       2
## 13 subaru           19.3   0.244       3
## 14 toyota           18.5   0.694      17
## 15 volkswagen       20.9   0.877      19
```

```r
# 4
left_join(
      mpg %>%
      group_by(manufacturer) %>%
      filter(year < 2004) %>%
      summarise(cty.mean.antes = mean(cty)),
      mpg %>%
      group_by(manufacturer) %>%
      filter(year > 2004) %>%
      summarise(cty.mean.desp = mean(cty)),
      by = "manufacturer"
)
```

```
## # A tibble: 15 x 3
##    manufacturer cty.mean.antes cty.mean.desp
##    <chr>                 <dbl>         <dbl>
##  1 audi                   17.1          18.1
##  2 chevrolet              15.1          14.9
##  3 dodge                  13.4          13.0
##  4 ford                   13.9          14.1
##  5 honda                  24.8          24
##  6 hyundai                18.3          18.9
##  7 jeep                   14.5          13.2
##  8 land rover             11            12
##  9 lincoln                11            12
## 10 mercury                13.5          13
## 11 nissan                 17.7          18.4
## 12 pontiac                17            17
## 13 subaru                 19            19.5
## 14 toyota                 18.2          19.1
## 15 volkswagen             21.2          20.5
```

```r
# mpg %>%
#       group_by(manufacturer, year) %>%
#       summarise(cty.mean = mean(cty))
#
# mpg %>%
#       filter()
```

# Clase 5 (24/9/2018)

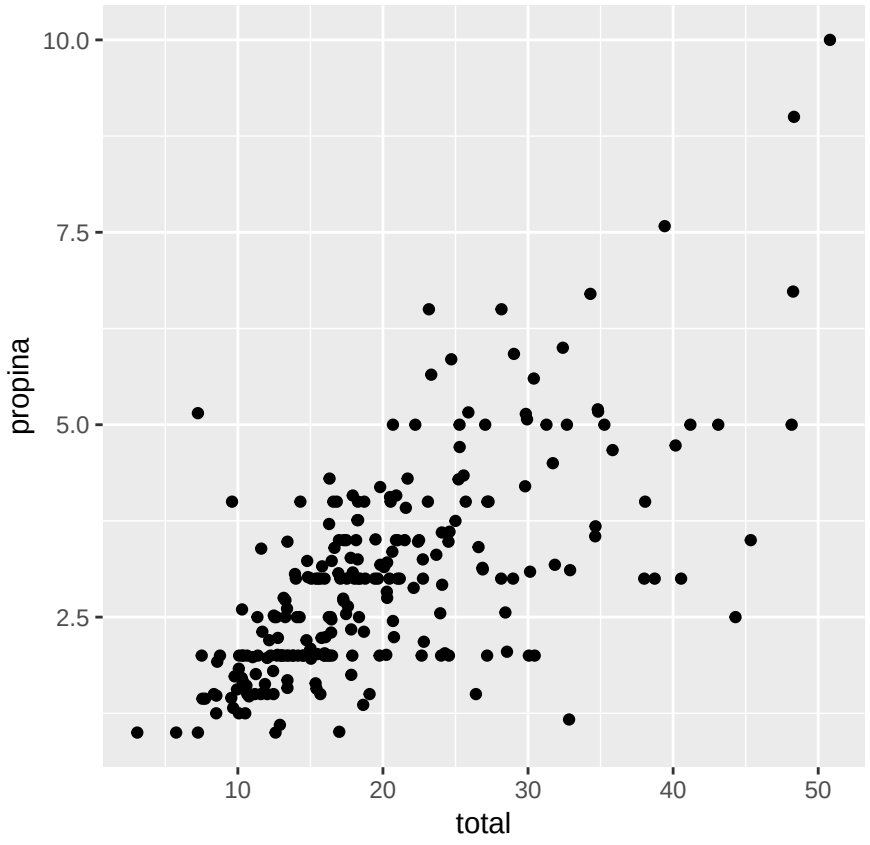
## Básicas de ggplot2

. ggplot2
. basado en la grámatica gráfica de Wilkinson (2006)
. gráfico: mapeo dedatos a atributos estéticos de objetos geométricos

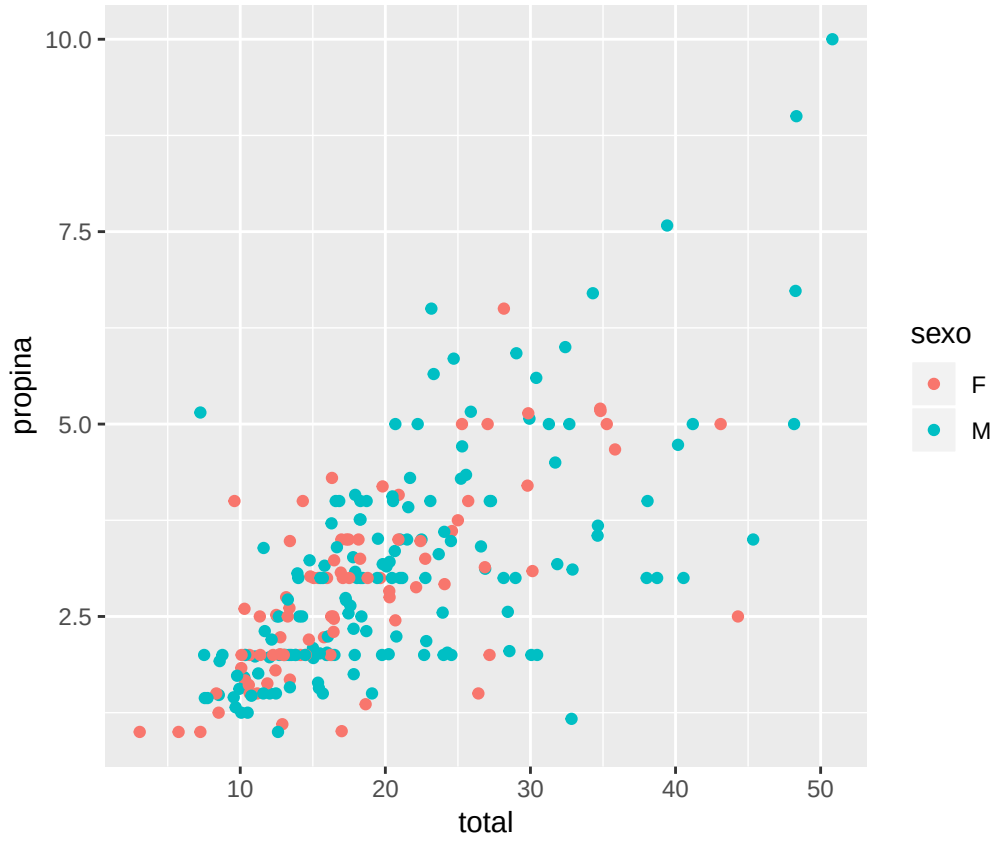
```
propinas = read_csv("propina.csv")
```

```
## Parsed with column specification:
## cols(
##   total = col_double(),
##   propina = col_double(),
##   sexo = col_character(),
##   fuma = col_character(),
##   dia = col_character(),
##   momento = col_character(),
##   cantidad = col_double()
## )
```

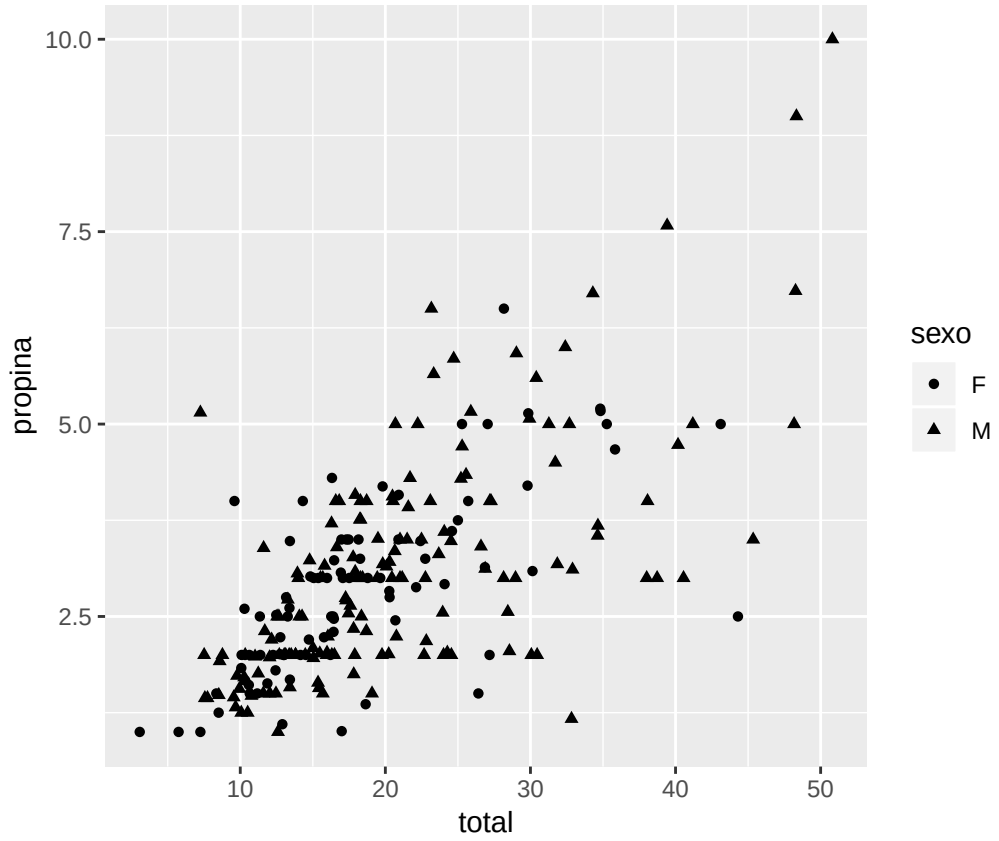
```
ggplot() +
      geom_point(data=propinas, aes(x=total, y=propina)) +
      theme(aspect.ratio=1)
```



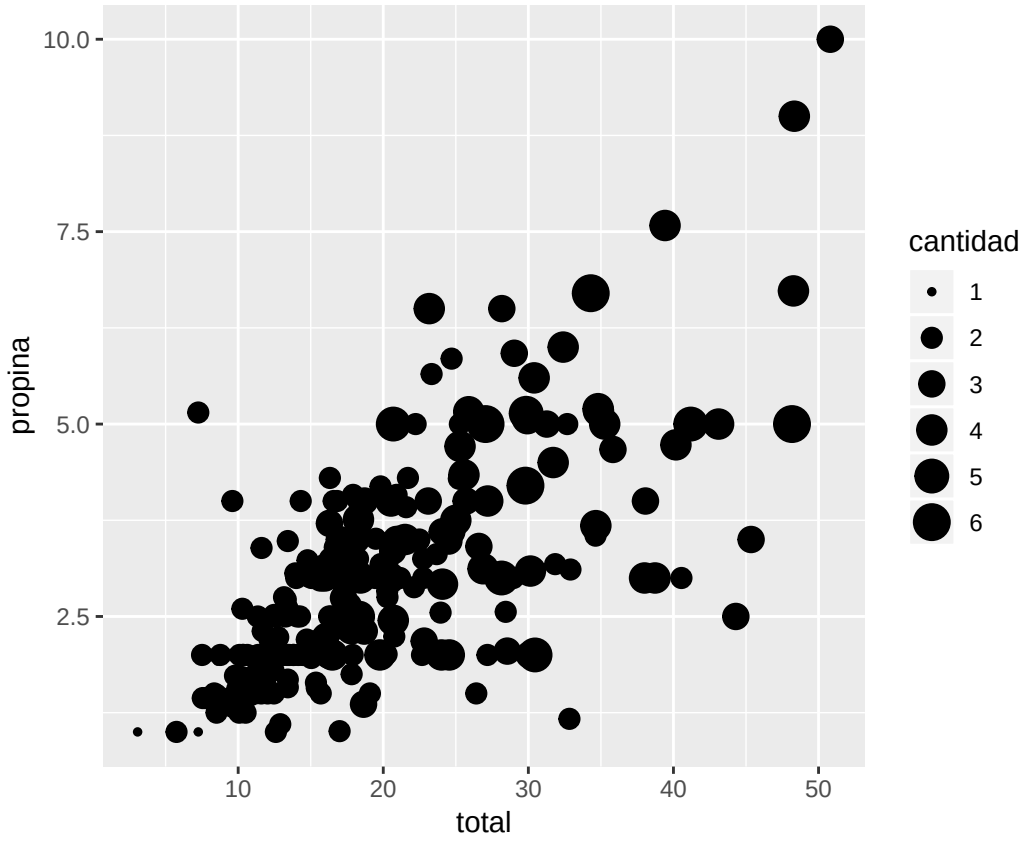

```
ggplot() +
      geom_point(data=propinas, aes(x=total, y=propina, colour=sexo)) +
      theme(aspect.ratio=1)
```



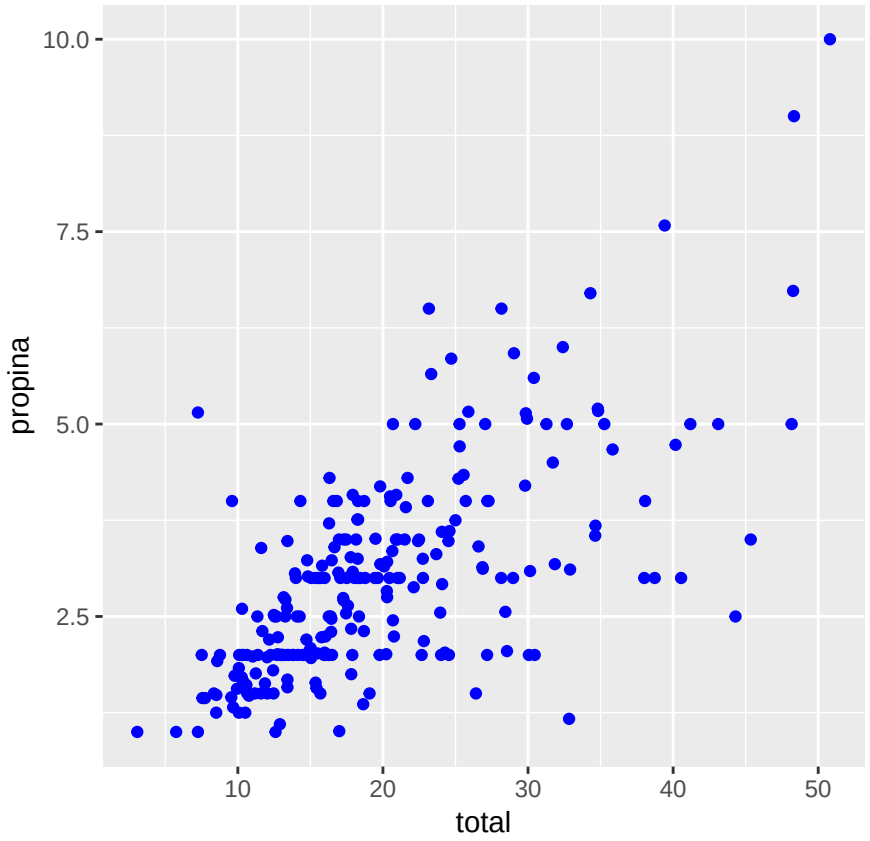

```
ggplot() +
      geom_point(data=propinas, aes(x=total, y=propina, shape=sexo)) +
      theme(aspect.ratio=1)
```



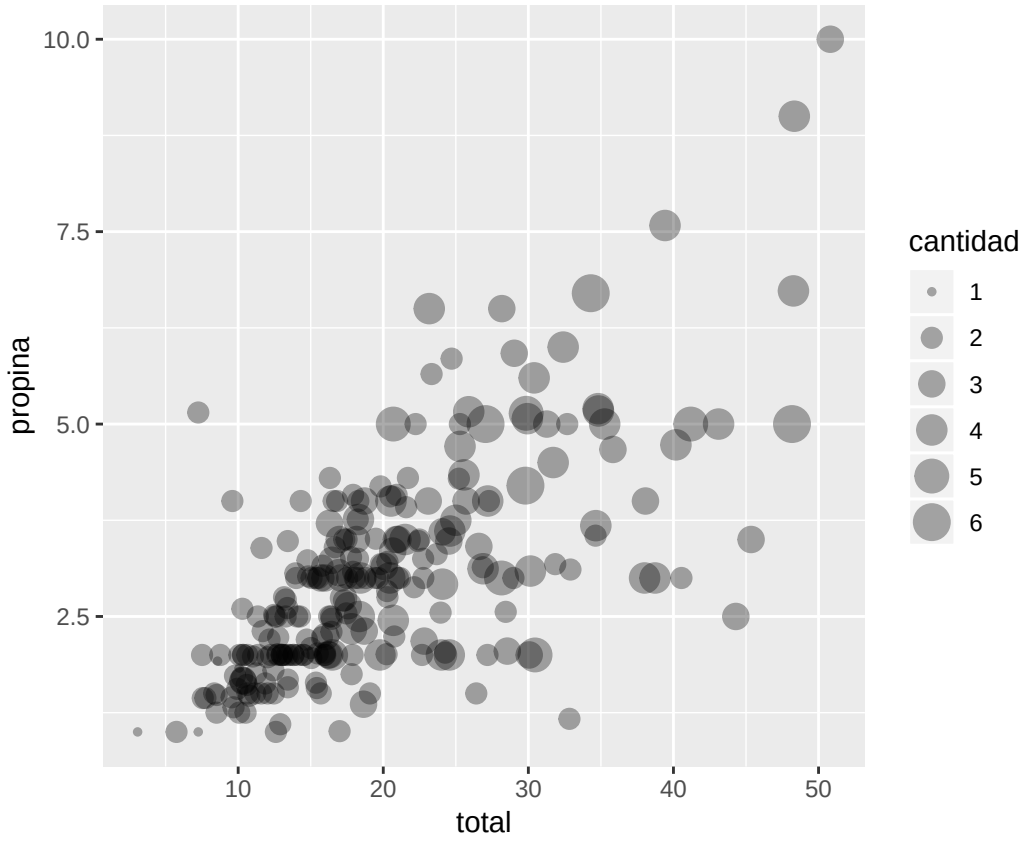

```
ggplot() +
      geom_point(data=propinas, aes(x=total, y=propina, size=cantidad)) +
      theme(aspect.ratio=1)
```




```
ggplot(data=propinas, aes(x=total, y=propina) ) +
      geom_point(colour="blue") +
      theme(aspect.ratio=1)
```

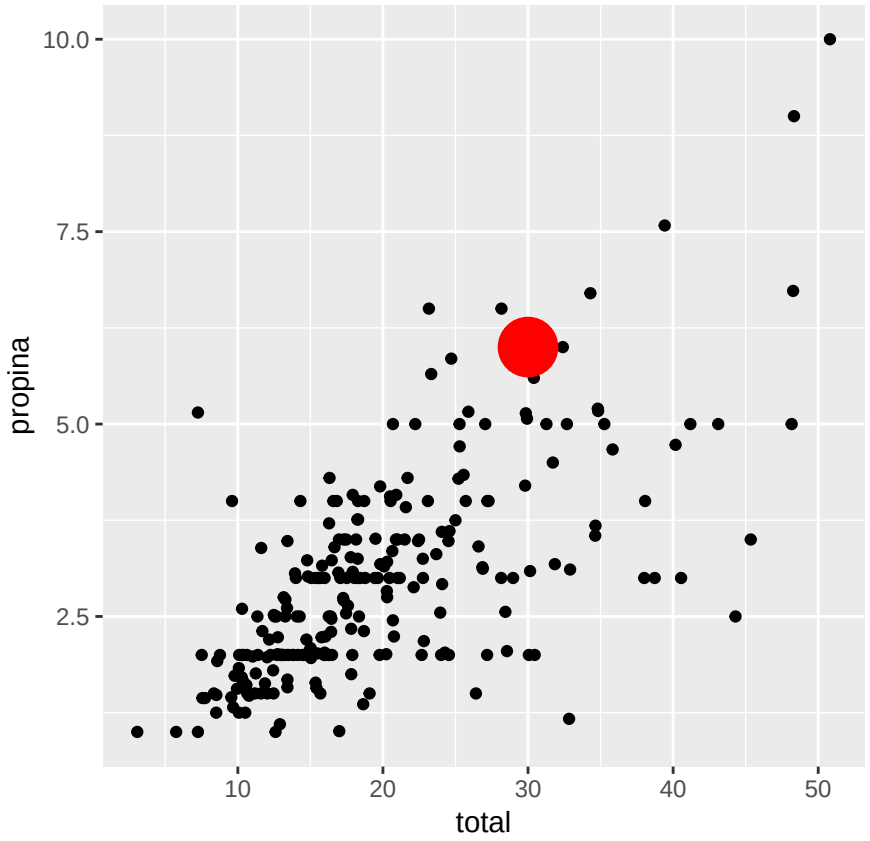



```
ggplot(data=propinas, aes(x=total, y=propina, size=cantidad)) +
      geom_point(alpha=1/3) +
      theme(aspect.ratio=1)
```



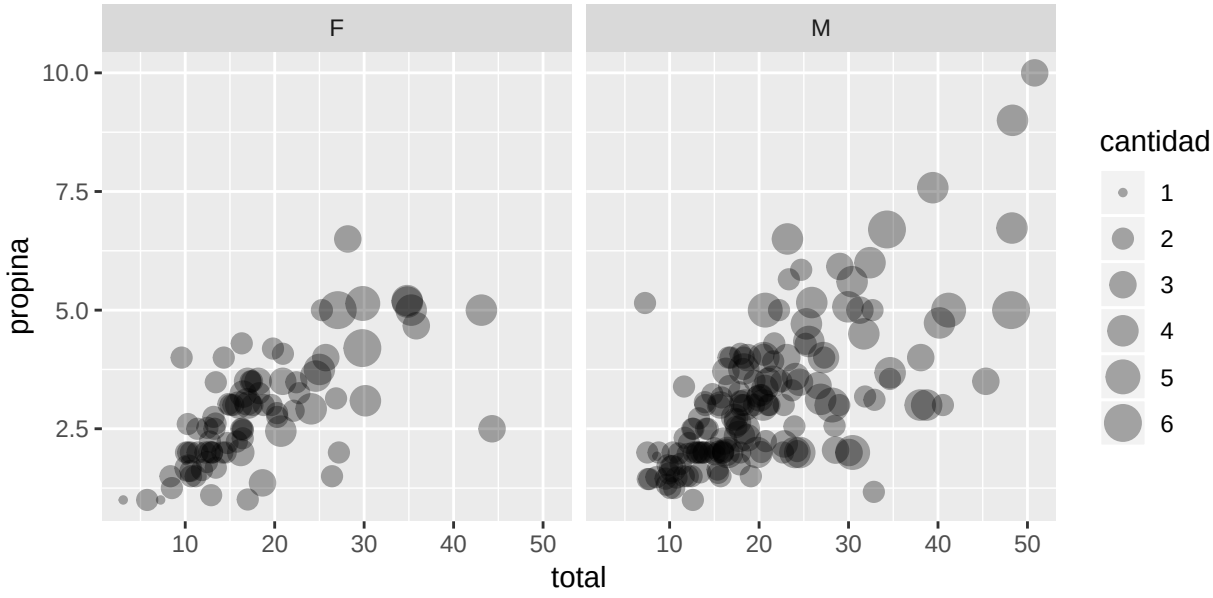

Las scales se modifican con una serie de funciones con el siguiente esquema de nombrado `scale_<aesthetic>_<type>`. Mirar `scale_<tab>` ver la lista de las funciones de scale.

```
ggplot() +
      geom_point(data=propinas, aes(x=total, y=propina)) +
      geom_point(data=data.frame(x=30, y=6), aes(x, y), color="red", size=10) +
      theme(aspect.ratio=1)
```

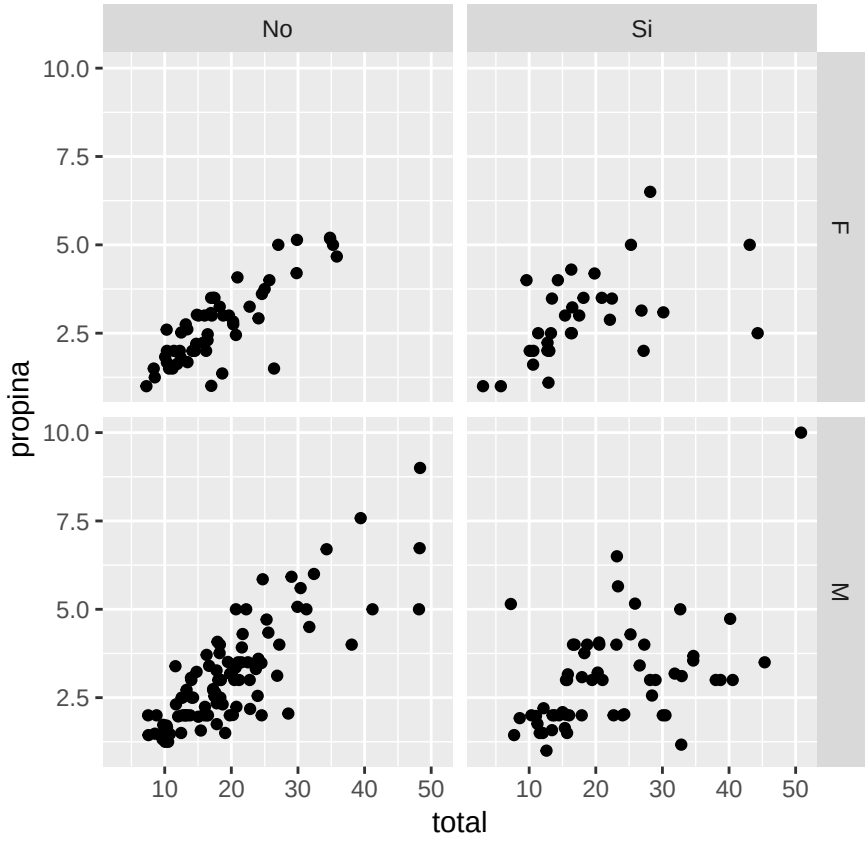



### Facets

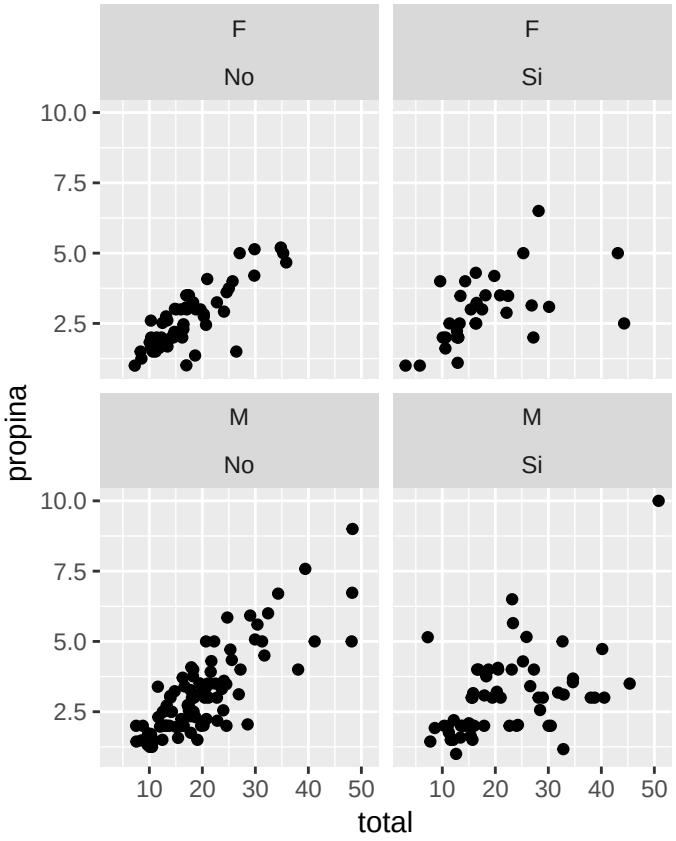
```
ggplot(data=propinas, aes(x=total, y=propina, size=cantidad)) +
      geom_point(alpha=1/3) +
      theme(aspect.ratio=1) +
      facet_wrap(~sexo)
```




```
ggplot(data=propinas, aes(x=total, y=propina)) +
      geom_point() +
      theme(aspect.ratio=1) +
      facet_grid(sexo~fuma)
```

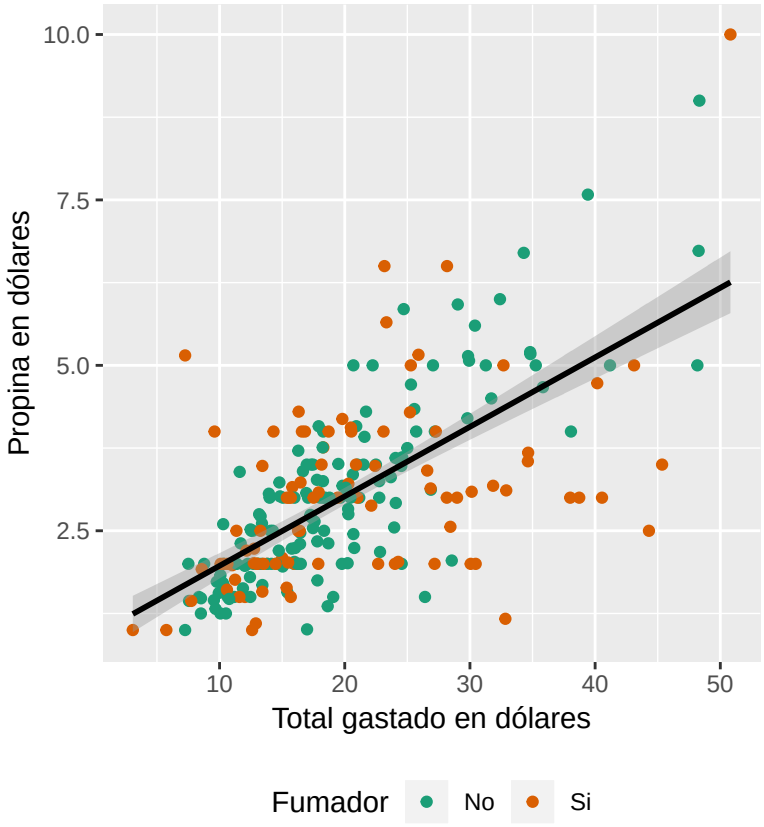



```
ggplot(data=propinas, aes(x=total, y=propina)) +
      geom_point() +
      theme(aspect.ratio=1) +
      facet_wrap(sexo~fuma)
```

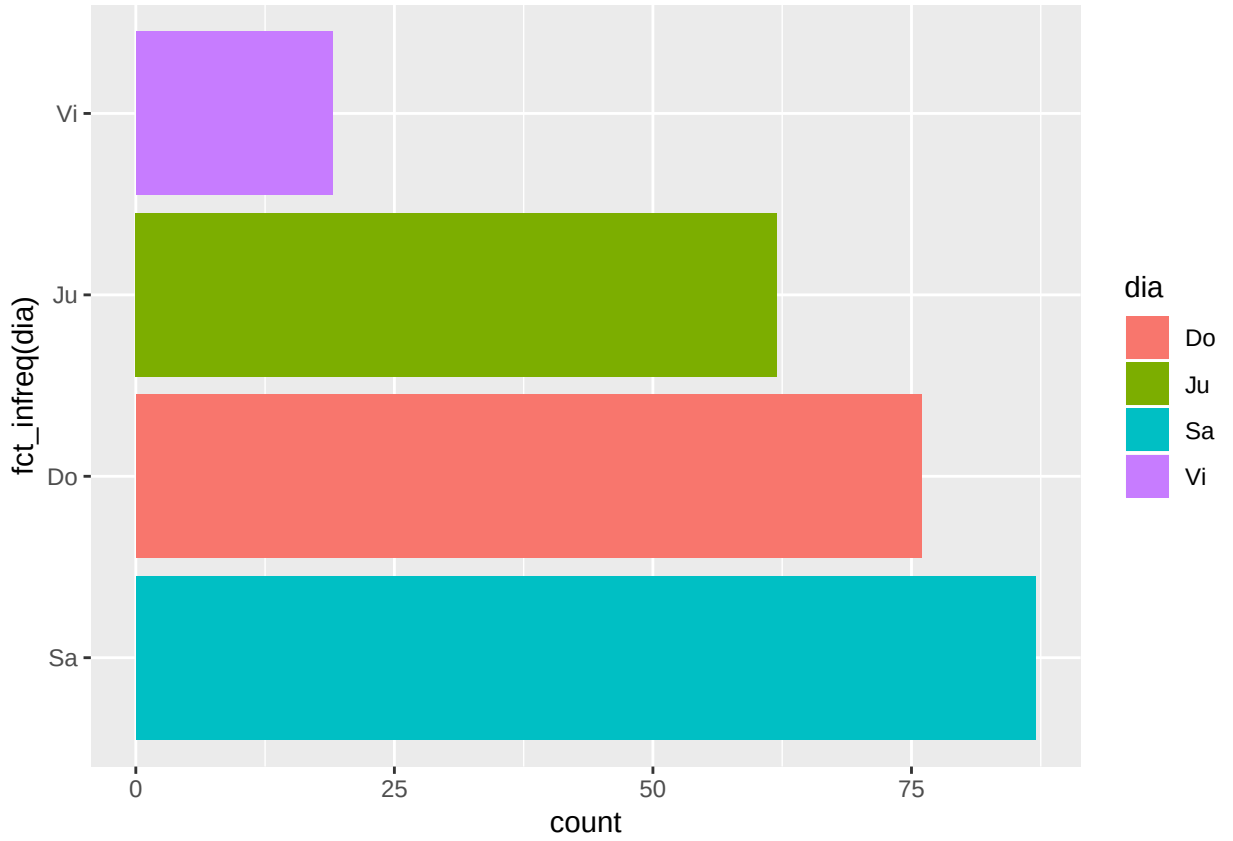



### Ejercicios de clase

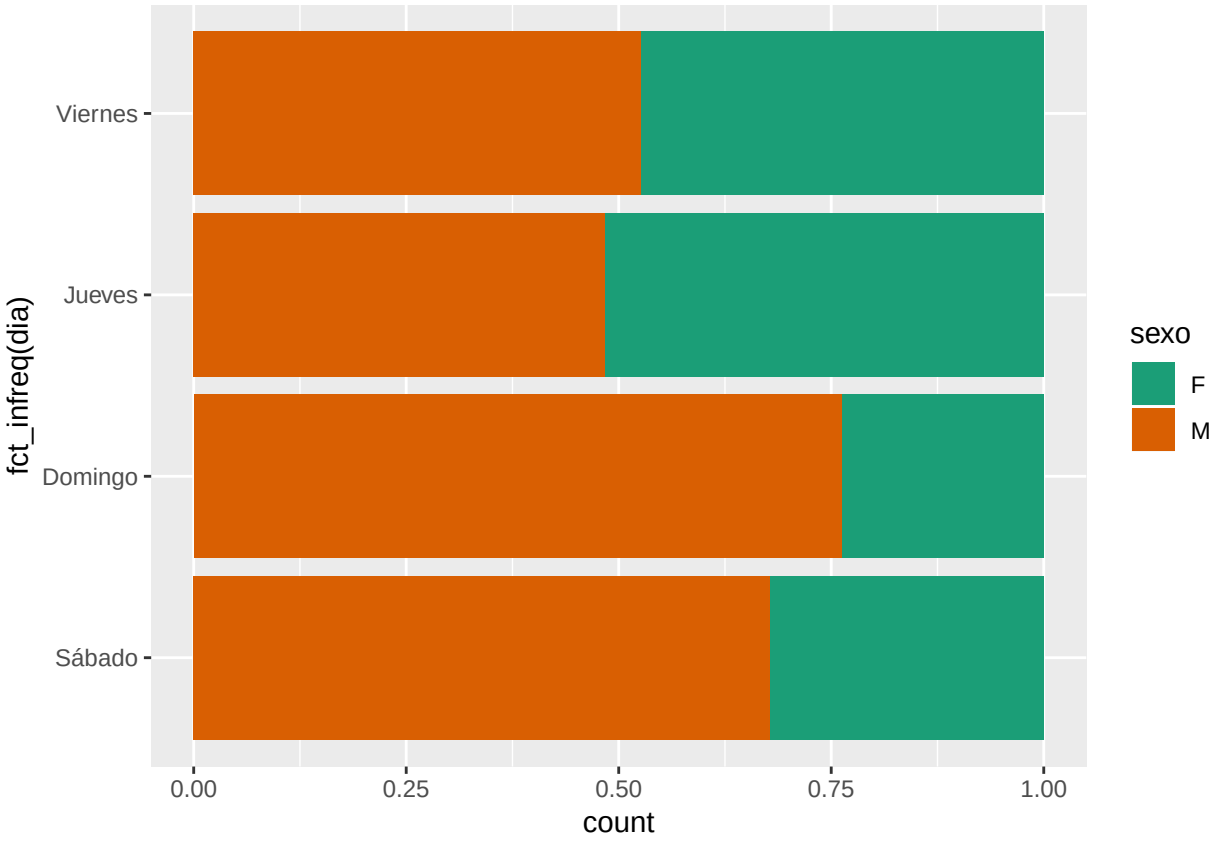
```
ggplot(data=propinas, mapping=aes(total, propina)) +
      geom_point(aes(color=fuma)) +
      geom_smooth(color="black", method="lm") +
      scale_color_brewer(palette="Dark2") +
      labs(x="Total gastado en dólares", y="Propina en dólares", color="Fumador") +
      theme(aspect.ratio=1, legend.position="bottom")
```




```
ggplot(propinas) +
      geom_bar(aes(fct_infreq(dia), fill=dia)) +
      coord_flip()
```




```
ggplot(propinas) +
      geom_bar(aes(fct_infreq(dia), fill=sexo), position="fill") +
      scale_x_discrete(labels=c("Vi"="Viernes", "Ju"="Jueves",
                                "Sa"="Sábado", "Do"="Domingo" )) +
      scale_fill_brewer(palette="Dark2") +
      coord_flip()
```




# Clase 6 (10/1/2018)

## Datos ordenados

- una observación por fila
- una variable por columna
- un valor por celda

## Lectura de datos

usar readr

- delim: permite especificar el delimitador
- csv: comas
- csv2: punto y coma
- tsv: tabulador
- table: espacios

librería `haven` - sas sas
- sav: spss
- dta: stata

leer excel `readxl`

```
# mirar ejemplo en pdf de clase
```

## Exportar datos

todo igual con `write_`

Usar `saveRDS` y `readRDS` y no `save` y `load` dado que las primeras no guardan el nombre del objeto, por lo tanto no se corre el riesgo de sobreescribir luego. Guarda de a un objeto. Si necesito guardar varios objetos, ponerlos en una lista.

## Ordenar datos

Parto de datos no ordenados y los necesito como tidy data.

key = categorias a colpasar value = valores de las observaciones (realizaciones de las variables)

verbos: - gather: mueve columnas a filas
- spread: mueve filas a columnas (gather a la menos uno)
- spearate: una columna a múltiples
- unite: une columnas

## Ejercicio de clase

```
tmi = readRDS("tmi.rds")

tmi_1 = tmi %>%
      gather(key="key", value="value", -Depto) %>%
      separate(key, into=c("tipo", "year"))

tmi_1 %>%
      filter(Depto == "MONTEVIDEO" | Depto == "INTERIOR", tipo == "Tasa") %>%
      spread(Depto, value) %>%
      mutate(ratio = MONTEVIDEO / INTERIOR)
```

```
## # A tibble: 19 x 5
##    tipo  year  INTERIOR MONTEVIDEO ratio
##    <chr> <chr>    <dbl>      <dbl> <dbl>
##  1 Tasa  1997      16.1       18.3  1.13
##  2 Tasa  1998      16.0       17.3  1.08
##  3 Tasa  1999      13.9       15.2  1.10
##  4 Tasa  2000      13.7       14.6  1.06
##  5 Tasa  2001      13.0       15.3  1.17
##  6 Tasa  2002      14.3       12.7  0.891
##  7 Tasa  2003      15.1       15.0  0.992
##  8 Tasa  2004      13.0       13.5  1.03
##  9 Tasa  2005      12.9       12.5  0.973
```

```
## 10 Tasa  2006     10.5       10.8  1.03
## 11 Tasa  2007     11.8       12.6  1.07
## 12 Tasa  2008     11.0       10.2  0.934
## 13 Tasa  2009      9.75       9.40 0.964
## 14 Tasa  2010      7.83       7.64 0.976
## 15 Tasa  2011      8.08       8.23 1.02
## 16 Tasa  2012     10.4        6.11 0.585
## 17 Tasa  2013      8.85       8.33 0.942
## 18 Tasa  2014      8.15       7.23 0.887
## 19 Tasa  2015      7.52       7.36 0.978
```

# Clase 7 (8/10/2018)

## Benchmarking and profiling

. Usar microbenchmark para testear tiempo de la libraria microbenchmark
. usar profvis

## Rcpp

. Conecta R con C++

```
# funcionará esto??
Rcpp::sourceCpp("primer_funcion.cpp")
```

# Clase 8 (22/10/2018)

```
f <- function(n){
      if(n < 2){
            return(n)
      } else {
            return(f(n-1) + f(n-2))
      }
      }

f(5)
```

```
## [1] 5
```

```
f(10)
```

```
## [1] 55
```

```
f(30)
```

```
## [1] 832040
```

```r
# funcionará esto??
Rcpp::sourceCpp("fibonacci.cpp")
g(30)
```

```
## Unit: relative
##   expr      min       lq     mean   median       uq      max neval cld
##  f(20) 342.8532 344.3467 205.8761 296.5601 265.2018 11.99765   100   b
##  g(20)   1.0000   1.0000   1.0000   1.0000   1.0000  1.00000   100  a
```