# R para contextos humanitarios de emergencia

## Reportes paramétricos y más

Violeta Roizman

# Personalizando reportes

Ya sabemos lo basico sobre generar reportes.

- RMarkdown y sus elementos
    - bloques de codigo (chunks)
    - texto
    - encabezado (hasta ahora no lo tocamos)
- Tipo de output
- Knit para generar

Pero ya vimos que las salidas por ahora no son muy lindas

En este modulo vamos a ver como personalizarlos para que sean publicables y mas eficientes

# Agregar tabla de contenidos

Por ejemplo, si quiero agregar una tabla de contenidos con los titulos y subtitulos puedo agregar en el encabezado el termino "toc: TRUE"

```
---
output:
  html_document:
    toc: TRUE
---
```

Es importante que respetemos la estructura tal como se indica! (los espacios y la alineacion de toc con respecto a html_document)

```
---
output:
  word_document:
    toc: TRUE
---
```

# Automatizando reportes

Es muy común tener que hacer un reporte cuyo resultado dependa de ciertos parámetros.

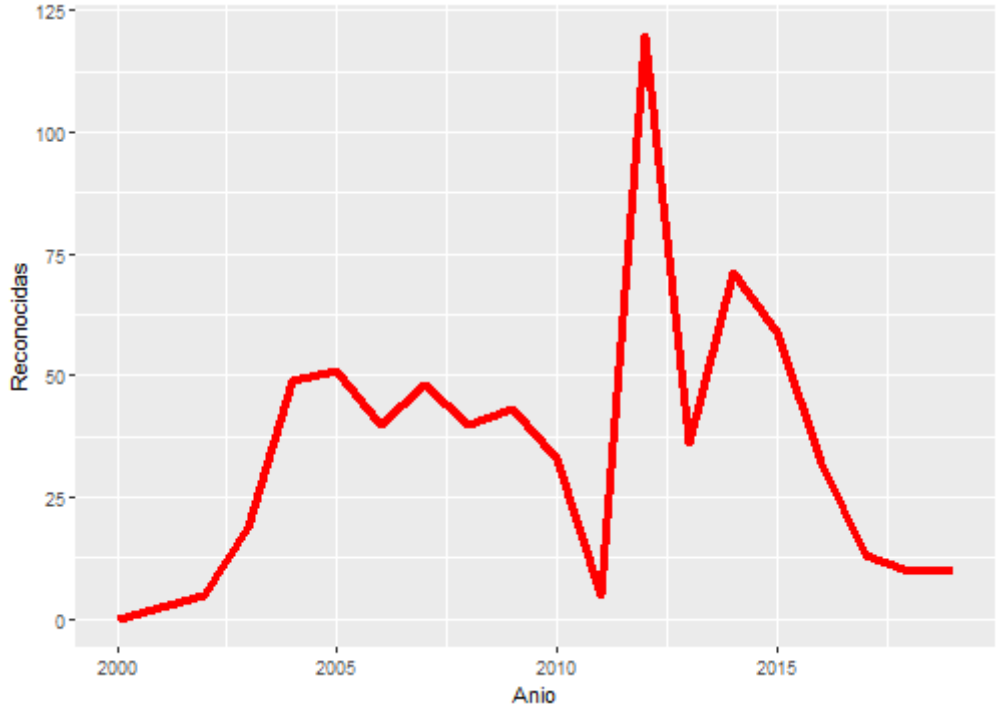
Por ejemplo, podrías tener un reporte que analiza las decisiones a los pedidos de asilo de personas colombianas en Perú.

```
library(tidyverse)

decisiones <- read_csv("data/decisiones_asilo_peru.csv")

decisiones_filtrado <- decisiones %>%
  filter(`Codigo Pais Origen` == "COL")

decisiones_filtrado %>%
  ggplot(aes(Anio, Reconocidas)) +
  geom_line(color = "red", size = 1.5)
```

# Automatizando reportes

Es muy común tener que hacer un reporte cuyo resultado dependa de ciertos parámetros.

Por ejemplo, podrías tener un reporte que analiza las decisiones a los pedidos de asilo de personas colombianas en Perú.



# Automatizando reportes

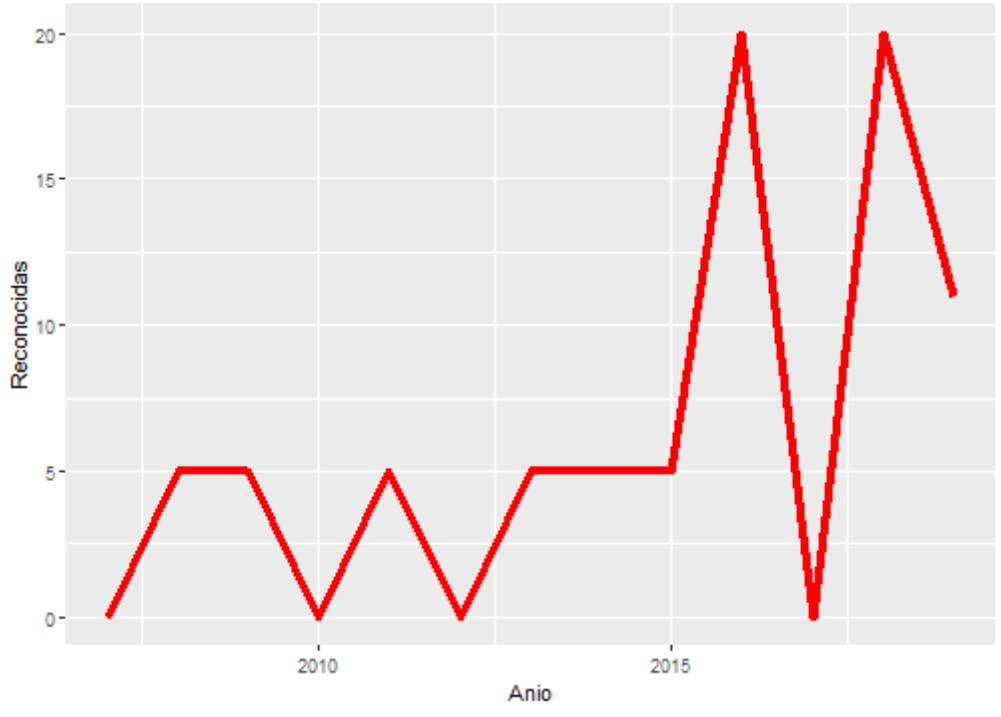
Si ahora querés hacer el mismo reporte pero para Haiti, tienes que abrir el archivo y modificar la llamada a `filter` para quedarte sólo con ese país:

```
library(tidyverse)

decisiones <- read_csv("data/decisiones_asilo_peru.csv")

decisiones_filtrado <- decisiones %>%
  filter(`Codigo Pais Origen` == "HTI")

decisiones_filtrado %>%
  ggplot(aes(Anio, Reconocidas)) +
  geom_line(color = "red", size = 1.5)
```

Si el reporte es largo y usa el nombre del país en múltiples lugares cambiar "COL" por "HTI" puede ser tedioso y propenso a error, ya que te obliga a modificar muchas partes del código.

# Automatizando reportes

Si ahora querés hacer el mismo reporte pero para Haiti, tienes que abrir el archivo y modificar la llamada a `filter` para quedarte sólo con ese país:




Si el reporte es largo y usa el nombre del país en múltiples lugares cambiar "COL" por "HTI" puede ser tedioso y propenso a error, ya que te obliga a modificar muchas partes del código.

# Parametrizando reportes

En estas situaciones podés crear un reporte parametrizado. La idea es que el reporte tiene una serie de parámetros que puede modificar la salida. Es como si el archivo de R Markdown fuera una gran función con sus argumentos!

Para generar un reporte parametrizado hay que agregar un elemento llamado params al encabezado con la lista de parámetros y sus valores por default.

```
params:
  pais: COL
```

# Parametrizando reportes

```
library(tidyverse)

decisiones <- read_csv("data/decisiones_asilo_peru.csv")

decisiones_filtrado <- decisiones %>%
  filter(`Codigo Pais Origen` == params$pais)

decisiones_filtrado %>%
  ggplot(aes(Anio, Reconocidas)) +
  geom_line(color = "red", size = 1.5)
```

# Automatizando reportes

Posibilidades

```
for (pais_actual in c("HTI", "VEN", "COL")) {
  rmarkdown::render("reporte_final.Rmd",
                    output_file = paste0("reporte-",
                                         pais_actual,
                                         ".docx"),
                    params = list(pais = pais_actual))
}
```

Este codigo crea 3 reportes!

# Control de chunks

Hay una serie de opciones que controlan si el código se ejecuta y si el resultado del código va a quedar en el reporte o no:

- `eval = FALSE` evita que se corra el código del chunk, de manera que tampoco va a mostrar resultados. Es útil para mostrar códigos de ejemplo si estás escribiendo, por ejemplo un documento para enseñar R.

- `echo = FALSE` corre el código del chunk y muestra los resultados, pero oculta el código en el reporte. Esto es útil para escribir reportes para personas que no necesitan ver el código de R que generó el gráfico o tabla.

- `include = FALSE` corre el código pero oculta tanto el código como los resultados. Es útil para usar en chunks de configuración general donde cargas las librerías.

# Control de chunks

- Si estás escribiendo un informe en el que no querés que se muestre ningún código, agregarle echo = FALSE a cada chunk nuevo se vuelve tedioso.
- Solución: cambiar la opción de forma global de manera que aplique a todos los chunks. Esto se hace mediante la función knitr::opts_chunk$set() , que setea las opciones globales de los chunks que le siguen. (Ya esta agregado en el primer bloque del documento reporte_final.Rmd)

# Control de chunks

- Vimos que a veces algunas funciones imprimen mensajes sobre lo que hacen. Por ejemplo, cuando read_csv lee un archivo describe el tipo de dato de cada columna
- Esto es útil pero en general no quiere que quede en el reporte. Para que no muestre estos mensajes basta con poner la opción message = FALSE
- Si queremos evitar una advertencia (warning) podemos indicar tamben warning = FALSE, separado por coma de los otros mensajes

# Tu turno

En el archivo reporte_final.Rmd elige las opciones adedcuadas para cada chunk

# Mejores reportes

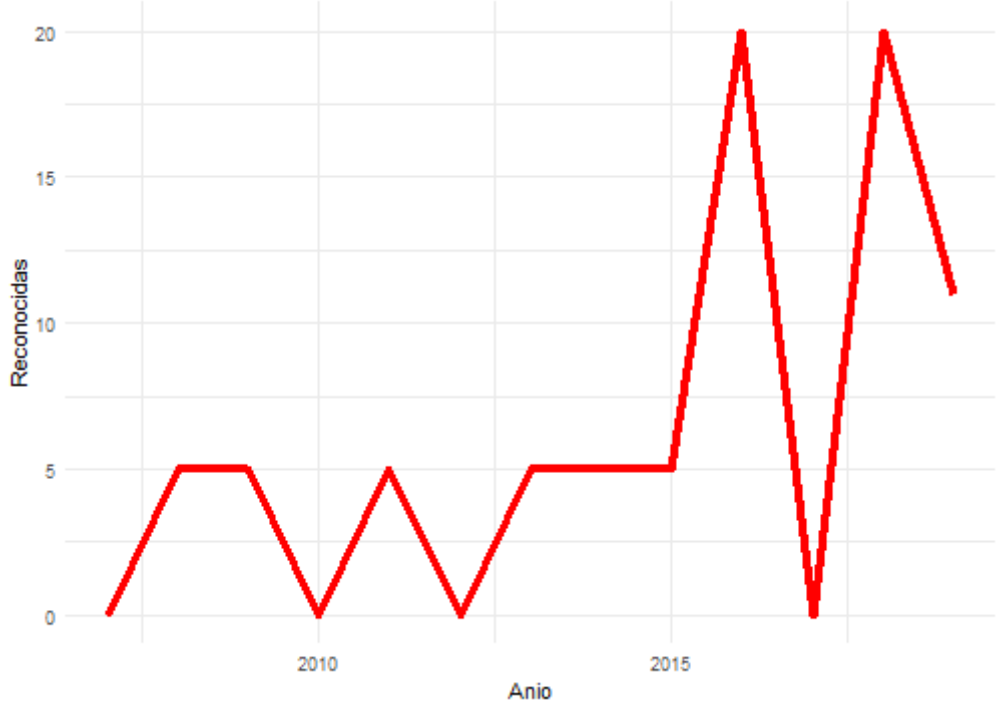
- Mejores graficos
- Mejores tablas

# Temas en `ggplot2`

- https://ggplot2.tidyverse.org/reference/ggtheme.html
- https://mran.microsoft.com/snapshot/2017-02-04/web/packages/ggthemes/vignettes/ggthemes.html

# Temas en ggplot2

```
decisiones_filtrado %>%
  ggplot(aes(Anio, Reconocidas)) +
  geom_line(color = "red", size = 1.5) +
  theme_minimal()
```



# Temas en ggplot2

```
# install.packages("ggthemes")
library(ggthemes)
decisiones_filtrado %>%
  ggplot(aes(Anio, Reconocidas)) +
  geom_line(color = "red", size = 1.5) +
  theme_fivethirtyeight()
```

# Tablas simples con kable

```
library(knitr)
kable(decisiones)
```

| Anio | Codigo Pais Origen | Codigo Pais Asilo | Nombre Pais de Origen | Nombre Pais Asilo | Tipo de procedimiento | Nombre del Procedimiento | Codigo de Tipo de Decision | T D |
|---|---|---|---|---|---|---|---|---|
| 2000 | COL | PER | Colombia | Peru | U | UNHCR | FI | F |
| 2000 | CUB | PER | Cuba | Peru | U | UNHCR | FI | F |
| 2001 | RUS | PER | Russian Federation | Peru | G | Government | FI | F |
| 2002 | COL | PER | Colombia | Peru | G | Government | FI | F |
| 2002 | CUB | PER | Cuba | Peru | G | Government | FI | F |
| 2003 | ARG | PER | Argentina | Peru | G | Government | FI | F |
| 2003 | COL | PER | Colombia | Peru | G | Government | FI | F |

# Tablas lindas con `kableExtra`

El paquete `kableExtra`, como su nombre lo indica, nació para extender el poder de la función kable.

Descargalo y prueba sus funcionalidades

# Licencia y material usado



Este material está inspirado y utiliza explicaciones de:

- R para Clima de Paola Corrales y Elio Campitelli
- Master the Tidyverse de Garrett Grolemund