

Programación científica en R

R Markdown

Marcos Ehekatzin García Guzmán

Noviembre de 2024

- Instalaremos Rmarkdown con cualquier otra paquetería.

```
# Install from CRAN  
install.packages('rmarkdown')
```

- También puede instalarse desde github de la siguiente manera


```
# Install from GitHub  
if (!requireNamespace("devtools"))  
  install.packages('devtools')  
devtools::install_github('rstudio/rmarkdown')
```


- Adicionalmente, para generar documentos en formato pdf utilizaremos la paquetería tinytex.
- Tinitext nos permite utilizar **Latex** desde **R**, por lo que es posible utilizar comandos e instalar paqueterías de **Latex** y compilarlas desde **R**.


```
#Install Tinytext  
install.packages('tinytex')  
tinytex::install_tinytex()
```


- R Markdown está diseñado para facilitar la reproducción.
 - ① Podemos computar código
 - ② Podemos generar documentos
- Para iniciar un nuevo documento, crearemos un nuevo archivo Rmd desde el menú, presionando **File -> New File -> R Markdown**

New R Markdown

 Document

 Presentation

 Shiny

 From Template

Title:

Author:

Date:

☐ Use current date when rendering document

Default Output Format:

☒ HTML
Recommended format for authoring (you can switch to PDF or Word output anytime).

☐ PDF
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

☐ Word
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

Create Empty Document

OK Cancel

- En la pantalla anterior podemos seleccionar el tipo de documento que necesitamos.
 - Document: `html`, `pdf`, `Word`.
 - Presentation: `Slidy`, `Beamer`, `ppt`
- Si bien podemos convertir nuestros documentos de un formato a otro, hay que tener cuidado.
 - Algunos formatos no soportan algunos comandos.

- En un documento tendremos tres componentes principales:
 - ① El código
 - ② El texto
 - ③ Los metadatos (YAML)
- El cuerpo del documento seguirá al YAML.
 - Formato
 - Paqueterías
 - Información del autor
 - Títulos

- Un YAML sencillo se verá como en la siguiente imagen:

```
---  
title: "Ejemplo"  
author: "Marcos Ehekatzin García Guzmán"  
date: '2024-11-10'  
output: pdf_document  
---
```

- Un YAML avanzado se verá de la siguiente manera:

```

---
geometry: "left=2.54cm,right=2.54cm,top=2.54cm,bottom=2.54cm"
output:
  pdf_document:
    toc: no
    number_sections: true
    includes:
      before_body: PortadaTesis.tex
    toc_depth: 2
mainfont: Times New Roman
subtitle: Implicaciones salariales para México
fontsize: 12pt
bibliography: biblio.bib
metadobib: false
csl: apa.csl
tables: no
toc-title: Contenido
urlcolor: black
linkcolor: black
header-includes:
- \usepackage[nottoc]{tocbibind}
- \renewcommand{\listfigurename}{Lista de figuras}
- \renewcommand{\listtablename}{Lista de tablas}
- \usepackage[setspace]
- \setstretch{1.5}
- \renewcommand{\footnotesize}{\fontsize{10pt}{11pt}\selectfont}
- \usepackage[utf8]{inputenc}
- \usepackage[spanish]{babel}
- \usepackage{graphicx}
- \usepackage{float}
- \usepackage{multirow}
- \usepackage{booktabs}
- \usepackage{pdfscape}
- \usepackage{caption}
- \usepackage[subcaption}
- \bibliographystyle{apalike}
- \usepackage[para,online,flushleft]{threeparttable}
- \usepackage{longtable}
- \usepackage{longtable,booktabs,threeparttable}

```

- Para el código, utilizaremos **chunks**.
 - Un **chunk** empieza y termina con comillas invertidas ““
 - En la primera línea del chunk, entre corchetes, definiremos el lenguaje del código y las opciones para el chunk.
- Las opciones básicas de los chunks son:
 - ① **include**: Nos permite controlar si el código y los resultados aparecen en el documento final.
 - ② **echo**: Nos permite controlar si el código aparece en el documento final. Los resultados aparecerán.
 - ③ **message**: Nos permite controlar si los mensajes generados por el código aparecen en el documento final.
 - ④ **warning**: Igual que **message** pero para los warnings.

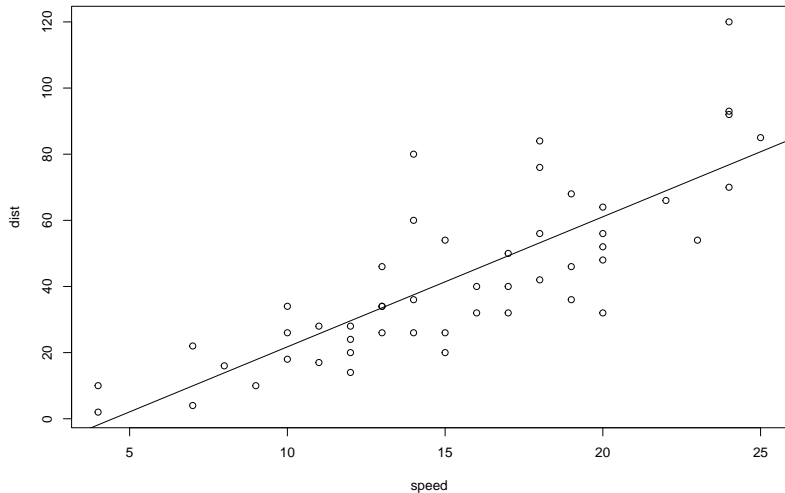
- En el primer chunk del documento, podremos establecer los valores predeterminados con `opts_chunk$set`

```
'''{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE, include = TRUE,
                      message = FALSE, warning = FALSE)
'''
```

- Los siguientes chunks seguirán la configuración a menos que indiquemos lo contrario.

- Con la configuración podemos incluir gráficas y tablas utilizando código directamente en R.

```
```{r, fig.cap= "Ejemplo", fig.==1, echo = F}  
fit = lm(dist ~ speed, data = cars)
b = coef(fit)
plot(cars)
abline(fit)
```
```



- Formato del texto:
 - Negritas: **`**text**`**
 - Itálicas: *`*text*`*
 - Código: ````code````
 - Enlaces: `[text](link)`, por ejemplo
`[RStudio](https://www.rstudio.com)`

- Texto en bloques:
 - ① Títulos: Los títulos se indican con #, los subtítulos con ## y así sucesivamente.
 - ② Para bullets podemos utilizar -, *, o +. Podemos anidar listas dentro de otras listas según el tabulado.
 - item 1
 - item 1.1
 - item 1.2
 - item 2
 - ③ Para listas enumeradas simplemente utilizamos los números y un punto (1., 2., ...)
 - ④ Podemos combinar bullets y listas enumeradas.

- Las expresiones matemáticas en el texto van entre $\$$ (*mathexpresion*)
- Para las expresiones matemáticas se utiliza la sintaxis de latex, por ejemplo: $f(k) = \binom{n}{k} p^k (1-p)^{n-k}$
 - (No veremos Latex, pero en el repositorio de la clase habrá una presentación)
- Para que las expresiones aparezcan enumeradas utilizaremos la sintaxis de \LaTeX

```
\begin{equation}
f(k) = \{n \text{ \choose k} \} p^{\{k\}} (1-p)^{\{n-k\}}
\end{equation}
```

$$f(k) = \binom{n}{k} p^k (1-p)^{n-k} \quad (1)$$

- Para las figuras podemos establecer el tamaño de la figura con `fig.width` o `fig.height`.
 - También podemos hacerlo con `out.width = "XX %"` o con `out.height = "XX %"`.
 - Podemos alinear la figura con `fig.align = X`, donde las alineaciones pueden ser `left`, `center` o `right`.
 - Podemos añadir un título a la figura con `fig.cao = "Título"`
- Si queremos añadir una imagen en lugar de una gráfica hecha en R podemos utilizar la sintaxis de Latex o `knitr::include_graphics("path")`

- Para las tablas, lo más conveniente es utilizar la paquetería `kableExtra`
- Para instalarlo: `install.packages("kableExtra")`

```
library(kableExtra)
dt <- mtcars[1:3, 1:6]
kable(head(dt), format = "latex", booktabs = T,
      caption="Ejemplo")
```

Cuadro 1: Ejemplo

| | mpg | cyl | disp | hp | drat | wt |
|---------------|------|-----|------|-----|------|-------|
| Mazda RX4 | 21.0 | 6 | 160 | 110 | 3.90 | 2.620 |
| Mazda RX4 Wag | 21.0 | 6 | 160 | 110 | 3.90 | 2.875 |
| Datsun 710 | 22.8 | 4 | 108 | 93 | 3.85 | 2.320 |