Module 4. Automated reports for results communication

BARÇA INNOVATION HUB
Universitas

Module 4. Automated reports for results communication

In previous courses, we have reviewed the process of working in the analysis of physical performance data, how to delve into each of them and choose the best tools or functions to import, explore, transform, analyse and visualize data.

RStudio has been presented as a highly versatile tool to respond to each of the needs raised - it is a comprehensive tool that allows each of the steps to be executed. This tool also has great reproducibility. With this we mean that, if we intend to repeat the same analysis process multiple times, we will only have to make sure that the new data we analyse has the same structure, and if our code is well organized, the new analysis can be done in a matter of seconds.

We have already mentioned different strategies to communicate results, that we should be as efficient as possible in communicating results and correctly choosing what information to share depending on the recipient of our analyses. Once these questions have been

responded, we need to choose the tool that will allow us to communicate the objectives in the desired way.

Sometimes, it can be a single visualization aid, a chart, that represents what we want to assess and goes directly to the point we want to make. Other times, we may need to develop systems with greater user interaction (also known as dashboards). These elements allow for greater flexibility in the exploration of the results and will be highly useful if that is our goal. In previous modules, we have seen how to make simple visualizations and in later modules we will see how to create these dashboards with interaction for end users.

In this module, we will deal with that intermediate step - the reports or analyses that we want to repeat multiple times.

The head of the academy where we work as a Sport Scientist asks us for a certain report on specific players in the different categories, where we see the results of the physical tests throughout the season, where the player is compared to the rest of the players and we can see their evolution. Another instance could be when we want to see the results of the monthly tests we carry out and we want to share that information with the coaching staff. Finally, a more technical case in which we developed a statistical model to make estimates of the physical development of our athletes and we want to send the results and also the code used to our colleagues in the sports science department for them to review it and thus receive their feedback.

These types of documents usually display different types of information - there is text and description of each of the objectives we pursue. We show images, visualizations, charts, and sometimes, we are also interested in sharing the methods we use, such as the type of data we get and the code.

All in all, what we seek with these documents is to be able to convert them into a format that is useful for sharing - a PDF document that we can print or a Word document to add comments, etc.

Which of the following statements about RStudio is correct?

- [ ] A) RStudio is a comprehensive tool that allows you to execute each of the steps of the analysis.
- [ ] B) RStudio is not suitable for processes that require high reproducibility.
- [ ] C) To repeat the same analysis process in RStudio, the new data must have the same structure.
- [ ] D) RStudio does not allow for efficient code organization.

E) RStudio is not versatile and cannot respond to different analysis needs.

SUBMIT

As we have highlighted before, Excel can be an excellent tool for objectives with a certain level of complexity. If our database is organized and we have Excel sheets with certain filters or macros, we could automate this process to a certain extent. The same applies to tools focused on visualization such as Power BI or Tableau.

However, in this course we have shown the potential of using a tool like RStudio for data analysis: its great flexibility in all steps of the process; its superiority to handle large amounts of data, and different types and sources of tables compared to other tools; as well as its ability to analyse and visualize later.

Which of the following statements correctly describes situations where reporting is required in a sports academy?

- [ ] A) Create reports on the physical evolution of specific players throughout the season.
- [ ] B) Share the results of the physical tests with the players.
- [ ] C) View the results of the monthly tests and share the information with the coaching staff.
- [ ] D) Develop a statistical model to estimate the physical development of the players and send the results together with the code used.
- [ ] E) Create reports exclusively on the finances of the academy.

SUBMIT

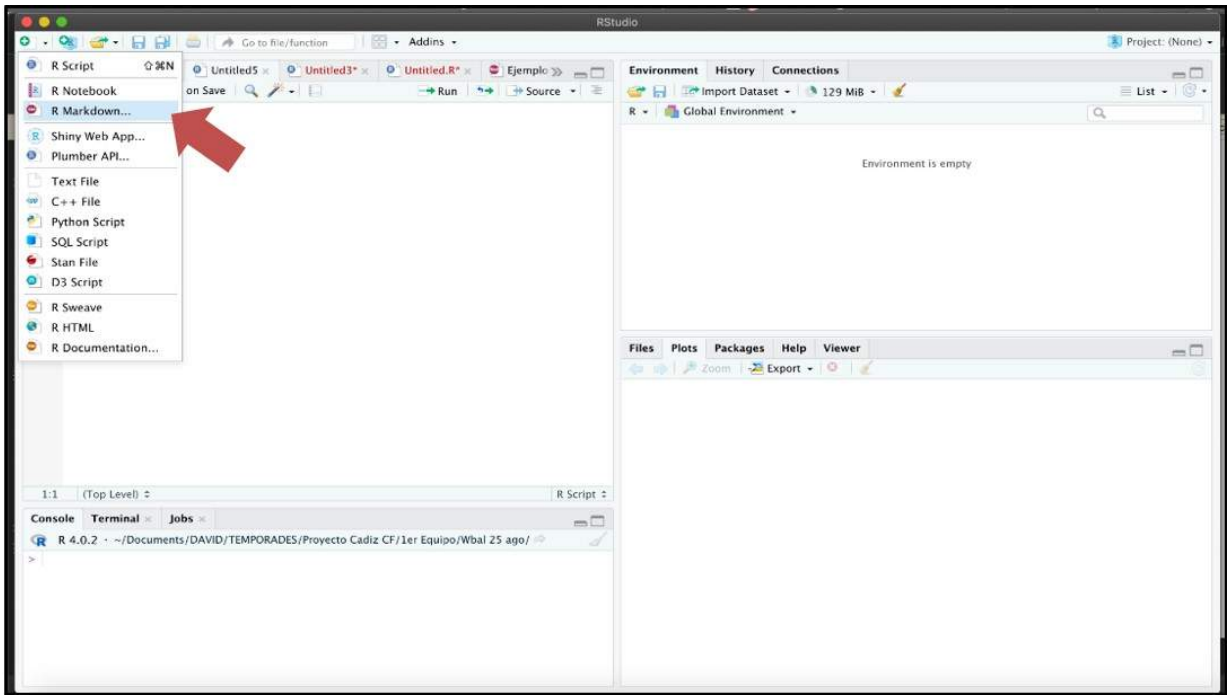
If the analysis we want to communicate could only be carried out with RStudio; How could we share those reports?

So far, we have only shown how to work with simple R files, the so-called R Script. If we were to use these files for the examples described above, the process of sharing a report would be very expensive. The analysis would be automated and the results would be collected efficiently, but what do we do with the information? We should probably write the text in a Word document, download the visualizations we have produced in RStudio and add them manually to the Word document and the same with the tables we have produce. If it is a process that we repeat on a weekly basis, we would be facing a very inefficient process.

To address this issue, RStudio has another type of file, called RMarkdown.

RMarkdown allows us to create files combining text, image, code, and table visualization, among others.

Figure 1: RMarkdown



Source: RStudio screenshot.

The main feature of the file when we use it in RStudio is that there are certain areas of the file intended for each purpose.

- Header: (YAML) within - - -.
    - It allows you to set the title of the document and the format you want the final file to be converted to.
- Code block: within ```.
    - All the code we write must be within these blocks, also called "chucks". If it is not, it will not

run.

- Written text:
    - outside the header and code blocks. In this document, we will look at how to change the formatting of the text so that it's the shape we want.

```
---
title: "Estructura RMarkdown"
author: "David"
date: '2024-04-27'
output:
  word_document: default
  html_document: default
---

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

Cargar Librerias

```{r,results='hide' }

library(tidyverse)
library(readxl)
```

Importar Datos y Visualizar

```{r,results='hide' }

data_rpe <- read_excel("WELLNESS.xlsx",sheet = 2)
data_rpe %>%
  ggplot(aes(RPE))+
  geom_density(fill="lightblue")+
  theme_bw()

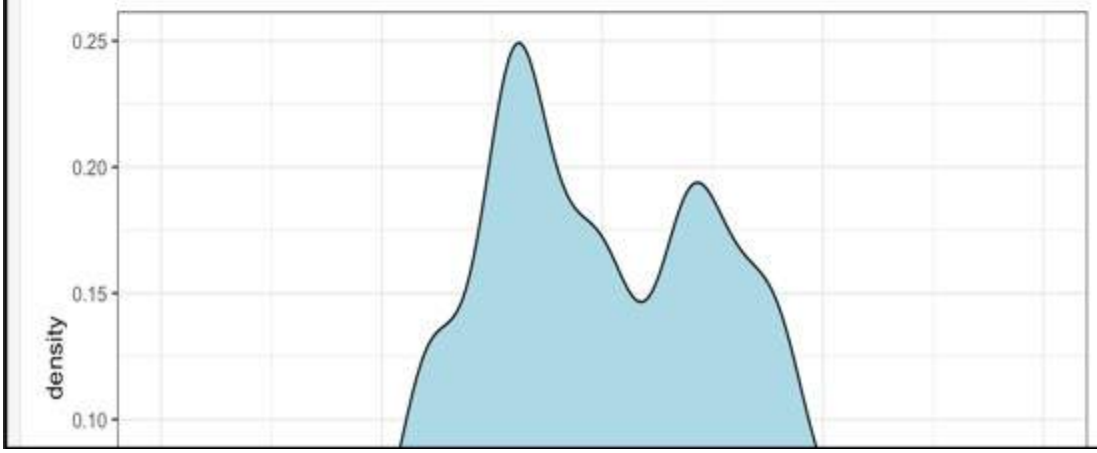
```
```

Heading

Text

Code block

Code block



Source: prepared by the author.

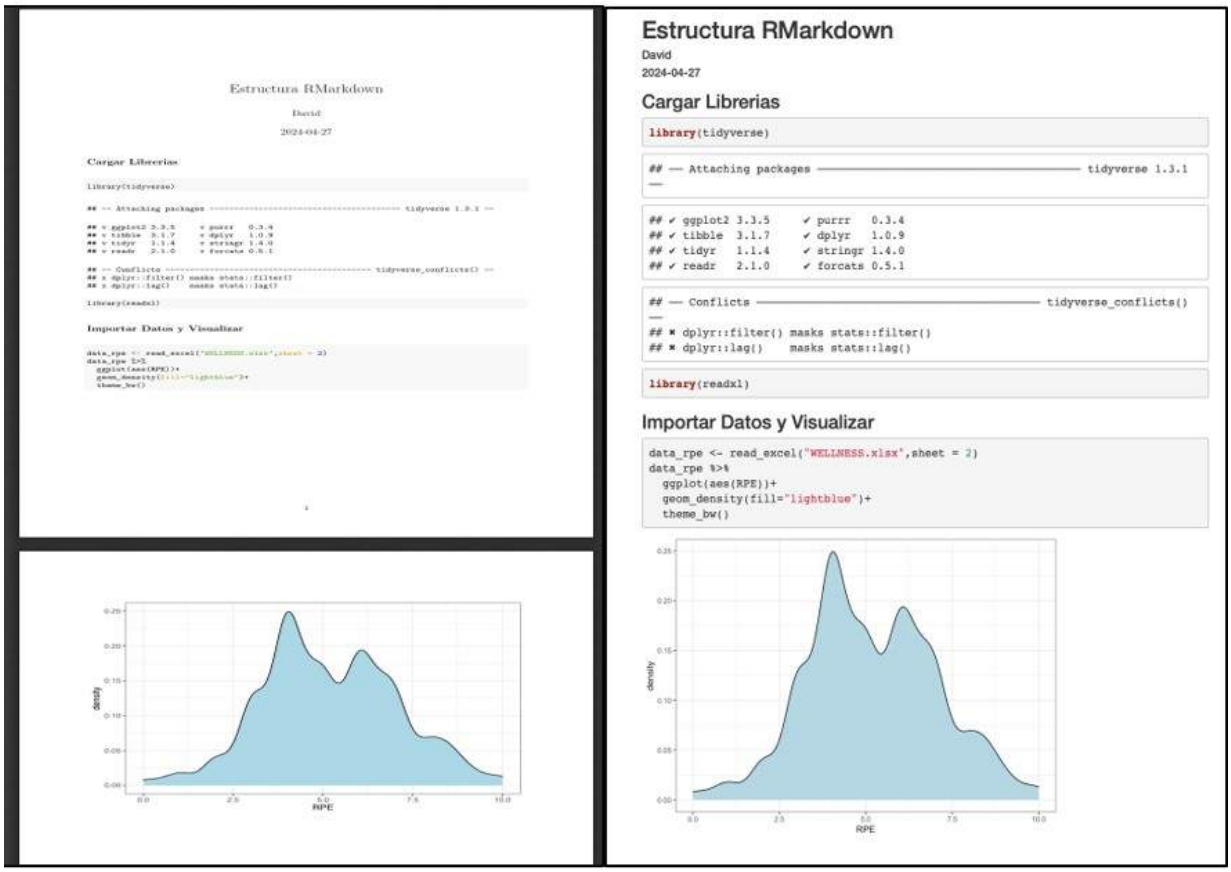
As we can see in the image above, in the case of RMarkdown documents, we see the results of each line of code in the same window that we have the file in, instead of seeing them in the graphics or console panels. This element can be very useful for each of the steps we carry out to be reflected in an organized way.

Figure 3: ***Knit***

Source: prepared by the author.

Although we will highlight the characteristics of each of the areas of the file below, the final step is to convert the file to the desired format. Once we have written all the code and structure that we want to communicate, we only have to click on the "knit" button. **This button will execute the entire code and return the file in the desired format**. Below there are the examples in PDF and HTML (web file).

Figure 4: PDF and HTML examples



Source: prepared by the author.

Using an RMarkdown file, we have converted a process that was inefficient to the "click" of a button that performs all the steps automatically and produces a document ready to be shared.

There are published books written and designed exclusively using RMarkdown, including doctoral theses or websites. In many selection processes where technical skills such as knowing how to use programming language are required, job application evaluators are required to submit an RMarkdown file to assess the candidate's ability to structure and write code to achieve a desired result, as well as their

understanding of it by sharing it with other professionals. Therefore, this is a very useful tool in our professional development in the field of physical performance data analysis.

Characteristics of the code block

As described above, this section must always be in "' in order to be executed. However, we can determine parameters that change the final result according to our objective, as we have highlighted in the example. Sometimes, we could decide to share the code we use (for example, with colleagues in our department) but on other occasions, the objective will be to present documents without any type of code, since the final recipient does not generally need that information. That format can be chosen by modifying certain elements within {}.

- include= FALSE. This will cause the code to not show up in the final document, although it will be run, but the results won't be displayed.
- echo=FALSE. It will display the results, but it will not show the code.
- warning=FALSE. Sometimes R displays warnings in code execution. If we don't want it to be shown in the final document, we must add it.

- results='hide'. It does not show the results.
- fig.show='hide'. It does not show the charts.

Characteristics of text formatting

In the same way that we change the format in Word or Excel files to highlight or structure our document (titles, indexes, highlighted words, etc.), we can do so in the RMarkdown document. We need to write the text in a certain way in order to achieve the desired objective in the document produced. In the following images, there are examples of the way to write the text in our RMarkdown document (gray shaded area) and the way it will be displayed in the final document (blank area below).

Figure 5: How to write the text in our RMarkdown document

Source: prepared by the author.

For titles, we have to add # and a space before the name of our title; the more #, the smaller the size of the title.

Figure 6: Example of text structure

Source: prepared by the author.

To separate lines we add \. Otherwise, even if the text appears on two lines in the RMarkdown file, it will be displayed on the same line in the created document.

Figure 7: Example of text formatting

Source: prepared by the author.

For indexed lists in the way shown in the image we will use a * or 2 depending on whether we want to highlight the text in italics or bold. To display the text in code format we will use ".

Figure 8: Example of links

Source: prepared by the author.

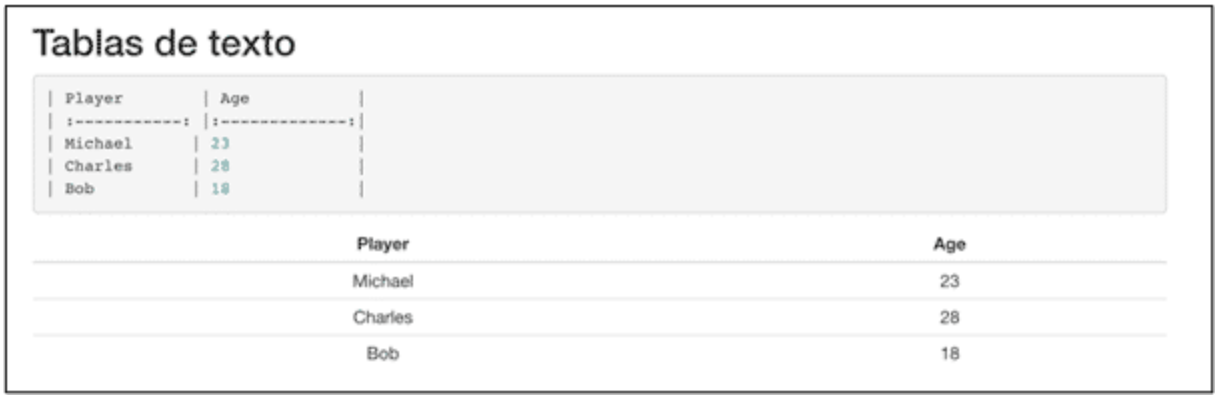
To add links, we need to write the text we want to be displayed between [] and the desired link in parentheses.

Figure 9: Example of adding Images

Source: prepared by the author.

If we want to add images from a link, we should copy this format.

Figure 10: Example of displaying text tables



Source: prepared by the author.

If we want to show text tables (there are other options to show tables that we get through the analysed code, as we will see in the videos), the structure to follow is the one indicated in the example.

Which of the following statements about editing documents in RMarkdown is correct?

- [ ] 1) To create titles in RMarkdown, you need to add # and a space before the title name.
- [ ] 2) The more #s that are added before the title name, the larger the title size.
- [ ] 3) To separate lines in RMarkdown, you should add "...".
- [ ] 4) If \ is not added between the lines, the text will appear on the same line in the created document.
- [ ] 5) RMarkdown documents do not allow formatting editing like in Word or Excel.

SUBMIT

CONTINUE