R Base Data Visualization

Basim Alsaedi

2024-05-16

Contents

In this course, we will focus on the base R graphics system, which provides a solid foundation for understanding how plots are constructed in R. Once you have a good grasp of the basics, you can easily transition to more specialized packages like ggplot2 and plotly to create even more sophisticated visualizations.

1.3 Installing and Configuring R and RStudio

Before installing **RStudio** in your computer, first start with **R**. **RStudio** is a front end program that lets you write **R** code, view plots, and do many other useful things. The detailed steps below show how to install R and RStudio in your computer system on both Windows, Mac and Linux operating systems.

1.3.1 Step-by-step guide on installing R

1. Download the R installer from https://cran.r-project.org/.

a). Click on the link for your operating system. Make sure the installer is for the latest R version. For example, the latest version is 4.3.3.

b). Click install R for the first time.

c). Use the download link at the top and save the file.

2. Run the installer (double click), default settings are fine.

1.3.2 Step-by-step guide on installing RStudio and Set Up

1. Wait until the R installer has finished. 2.Download RStudio installer from the official website https://posit.co/download/rstudio-desktop/.
2. After the download is complete, double-click on the installer and follow the installation steps to install it in your computer.

After successful installation, you can launch RStudio by double-clicking the RStudio icon on your desktop or from the Start menu. You can install the packages using the install.packages() function. However, for the base R visualization, you don't need to install any package which will support in plotting of graphs except where we you will be required to use data that comes with R packages. The last set up may be setting your working directory to the folder where your R scripts and data files are located. This makes it easier to access your files but it is optional.

1.4 Basic R Concepts.

1.4.1 Introduction to R syntax and basic commands

R is designed for statistical computing and graphics. In this section, we will cover some basic syntax and commands to help you get started with R.

1. R as a Calculator:

You can use R as a simple calculator. Here are some basic arithmetic operations:

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIjIEFkZGl0aW9uXG4yICsgM1xuIyBTdWJ0cmFjdGlvblxuNSAtIDJcbiMgTXV

2. Assigning Values to Variables:

You can store values in variables using the assignment operator <- or =.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIjIEFzc2lnbmluZyBhIHZhbHVlIHRvIGEgdmFyaWFibGVcbnggPC0gNVxueSA

3. Basic Data Types:

R supports several basic data types, including numeric, character, logical, and complex.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIjIE51bWVyaWNcbm51bSA8LSAxMFxuIyBDaGFyYWN0ZXJcbmNoYXIgPC

4. Vectors:

A vector is a sequence of data elements of the same basic type. You can create a vector using the **c()** function.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIjIENyZWF0aW5nIGEgbnVtZXJpYyB2ZWN0b3Jcbm51bXMgPC0gYygxLCA

5. Indexing and Slicing:

You can access elements of a vector using square brackets [].

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIjIEFjY2Vzc2luZyBlbGVtZW50cyBvZiBhIHZlY3RvclxubnVtcyA8LSBjKDEsI

6. Functions:

R has a large number of built-in functions, and you can also create your own functions for some tasks that can not be achieved by built-in functions.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIjIEJ1aWx0LWluIGZ1bmN0aW9uXG5zcXJ0KDE2KSAgICMgU3F1YXJlIHJv

1.4.2 Overview of R's data types and structures essential for visualization

R provides several data types and structures that are essential for data visualization. Understanding these data types and structures is crucial for effectively analyzing and visualizing data.

1. Numeric:

Numeric data type is used to represent continuous numerical values.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiJudW0gPC0gNS42OyBudW0ifQ==

2. Integer:

Integer data type is used to represent integer values.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiJpbnQgPC0gMTBMIn0=

3. Character:

Character data type is used to represent text data.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiJjaGFyIDwtIFwiSGksIEpvaG4hXCIifQ==

4. Logical:

Logical data type is used to represent Boolean values (**TRUE** or **FALSE**).

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiJsb2dpYyA8LSBUUlVFIn0=

5. Vector:

A vector is a sequence of data elements of the same basic type. It is created using the concatenate **c()** command.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiJudW1zIDwtIGMoMSwgMiwgMywgNCwgNSkgIyBBIHZlY3RvciBvZiBudW1l

6. Matrix:

A matrix is a two-dimensional array with rows and columns in that order respectively, that is ($R \times C$). It is created using the **matrix()** command.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiJtYXQgPC0gbWF0cml4KDE6MTIsIG5yb3cgPSAzLCBuY29sID0gNCkgIyBBI

7. Data Frame:

A data frame is a two-dimensional data structure with rows and columns, similar to a spreadsheet. It can contain several data types. It is created using the command **data.frame()** command.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiJkZiA8LSBkYXRhLmZyYW1lKFxuICBOYW1lID0gYyhcIkpvaG5cIiwgXCJBb

8. List:

A list is an ordered collection of objects (which may be of different types: numeric, character, etc.). It is created using the **list()** command.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiJsc3QgPC0gbGlzdChcbiAgTmFtZSA9IGMoXCJKb2huXCIsIFwiQWxpY2VcIi

9. Factors:

Factors are used to represent categorical data. It is created using the **factor()** command.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiJnZW5kZXIgPC0gZmFjdG9yKGMoXCJNYWxlXCIsIFwiRmVtYWxlXCIsIFw

1.5 Practical Examples: Exercises

Exercise 1: Basic arithmetic

Create a variable called x and give it the value 15.Take the exponent of the variable and add 5 to the final result. Print the final result of x.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIjIFlvdXIgY29kZSJ9

Exercise 2: Vectors:

The weights of five people before and after a diet programme are given in the table below. Read the 'before' and 'after' values into two different vectors called **before** and **after**. Use **R** to evaluate the amount of weight lost for each participant. What is the average amount of weight lost?

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIgIGRmPWRhdGEuZnJhbWUoXG4gICAgWDEgPSBjKDc4LCA2NyksXG4gI

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIjWW91ciBjb2RlIn0=

Exercise 3: Matrices

Create two matrices called **A** and **X** defined below.

$$A = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 10 \end{pmatrix}$$

$$X = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$$

Find:

i) the product of **A** and **X**.

ii) the transpose of **A**.

iii) the determinant of **A**.

iv) the diagonal of **A**.

v) the inverse of **A**.

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIjWW91ciBjb2RlIn0=

Exercise 4: Creating Data Frames

(a) Create a small data frame representing a database of films. It should contain the fields **title**, **director**, **year**, **country**, and at least three films.

(b) Create a second data frame of the same format as above, but containing just one new film.

(c) Merge the two data frames using rbind().

(d) Try sorting the titles using sort(): what happens?

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIjWW91ciBjb2RlIn0=

Exercise 5: Factors and Simple plot

Suppose we have the heights of 100 individuals, where the first are 50 male and the rest female. Generate 100 fixed random numbers from a normal distribution where the mean height of male is 170 while that of female is 160 with an equal standard deviation of 10 and call that vector as 'height'. Create another vector called 'sex' with two entries 'M' and 'F' each replicated 50 times. Tell R to treat 'sex' as a categorical variable and name it as 'Sex'. Plot **Sex** against **height** using **plot()** function. Which type of plot have you obtained? What happens if you try to plot **sex** against **height** instead?

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiIjWW91ciBjb2RlIn0=

2 R BASE GRAPHICS - STARTING WITH THE BASICS

2.1 Exploring Base Graphics in R

2.1.1 Overview of the philosophy behind R's base graphics system, including its stateful nature

2.1.1.1 Stateful Nature R's base graphics system is stateful, meaning that plots are built up incrementally. You start with an empty plot and add elements to it one by one. This is in contrast to systems like ggplot2, which use a declarative approach where you specify the plot all at once. Stateful nature means that every new plotting command modifies the existing plot or creates a new one if none exists. This allows for a high degree of flexibility but can sometimes lead to complex and intuitive behavior.

R's base graphics system provides a set of low-level graphics primitives for creating plots. These primitives include functions for drawing points, lines, polygons, text, and more. By combining these primitives, you can create a wide variety of plots, from simple scatter plots to complex multi-panel layouts.

2.1.1.2 Philosophy The philosophy behind R's base graphics system is to provide a flexible and powerful tool for creating a wide range of plots. The emphasis is on simplicity and ease of use, making it easy for users to quickly create informative visualizations.

However, despite that base graphics are powerful, they do have some limitations compared to more modern plotting systems like ggplot2. For instance, they lack some of the advanced features of ggplot2, such as automatic faceting and easy customization of plot themes.

2.1.2 Introduction to Core Plotting Functions in R

R provides a variety of core plotting functions that are useful for creating basic visualizations. In this section, we will explore some of the most commonly used plotting functions. The commonly used core plotting functions are **plot()**, **hist()**, **boxplot()** and **barplot()**.

- The **plot()** function is used to create scatter plots, line plots, and other types of plots. It is a versatile function that can be used to visualize relationships between two or more variables.
- The **hist()** function is used to create histograms, which are used to visualize the distribution of a single numeric variable. Note that the optional argument breaks chooses (approximately) how many bins the histogram should have, and col alters the colour of the bars.
- The **boxplot()** function is used to create box plots, which are used to visualize the distribution of a numeric variable, optionally broken down by a categorical variable.
- The **barplot()** function is used to create bar plots, which are used to visualize the distribution of a categorical variable.