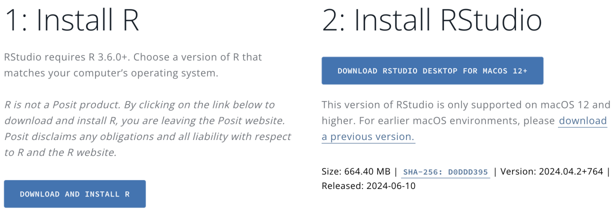
Introduction to R Programming

Haodong Ling

What is R, Rstudio, Rmd?

- R is programming language, like python, but specifically designed for statistical computing. It is widely used among statisticians, data analysts, and researchers.
- RStudio is an integrated development environment (IDE) for R. It provides a user-friendly interface that makes it easier to write, debug, and visualize R code.
- R Markdown is a file format used to create dynamic documents with R. An R Markdown file (.Rmd) allows you to combine text, R code, and the output of the code (such as plots and tables) in a single document.
- Simply install R and Rstudio on `https://posit.co/download/rstudio-desktop/`!



1: Install R

RStudio requires R 3.6.0+. Choose a version of R that matches your computer's operating system.

R is not a Posit product. By clicking on the link below to download and install R, you are leaving the Posit website. Posit disclaims any obligations and all liability with respect to R and the R website.

2: Install RStudio

This version of RStudio is only supported on macOS 12 and higher. For earlier macOS environments, please download a previous version.

Size: 664.40 MB | SHA-256: D0DDD395 | Version: 2024.04.2+764 | Released: 2024-06-10

Figure 1: installation

How to run code?

- In command line, simply press enter
- For your script (.R file, .Rmd file) you can run a chunk of code by select and
  - command + Enter on Mac
  - Ctrl + Enter on Windows
- For Rmd chunk, you can also click the green triangle button.

There are other shortcuts in RStudio:

- Tab completion
- Command history: up/down arrows
- RStudio: select a line or block for execution
- For keyboard shortcuts in RStudio see Tools -> Keyboard Shortcuts Help

Working directory

To read and write from R, you need to have a firm grasp of where in the computer's filesystem you are reading and writing from.

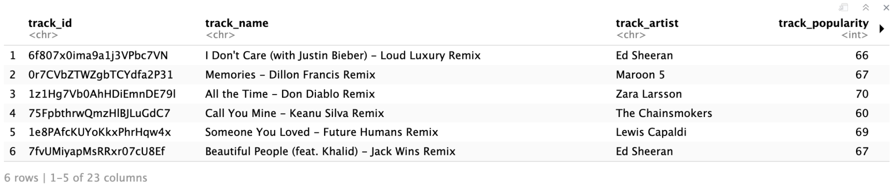
```r
## What directory does R look for files in (working directory)?
getwd()

## Changing the working directory (Linux/Mac specific)
setwd('~/Desktop/courses/stat151a/lab1') # change the working directory

## Changing the working directory (Windows specific)
## Windows - use either \\ or / to indicate directories
# setwd('C:\\Users\\Your_username\\Desktop\\courses\\stat151a\\lab1')
```

dataset operations

```r
data_location <- "../datasets"
df <- read.csv(file.path(data_location, "spotify_songs.csv"))
head(df)
```



	track_id <chr>	track_name <chr>	track_artist <chr>	track_popularity <int>
1	6f807x0ima9a1j3VPbc7VN	I Don't Care (with Justin Bieber) – Loud Luxury Remix	Ed Sheeran	66
2	0r7CVbZTWZgbTCYdfa2P31	Memories – Dillon Francis Remix	Maroon 5	67
3	1z1Hg7Vb0AhHDiEmnDE79l	All the Time – Don Diablo Remix	Zara Larsson	70
4	75FpbthrwQmzHlBJLuGdC7	Call You Mine – Keanu Silva Remix	The Chainsmokers	60
5	1e8PAfcKUYoKkxPhrHqw4x	Someone You Loved – Future Humans Remix	Lewis Capaldi	69
6	7fvUMiyapMsRRxr07cU8Ef	Beautiful People (feat. Khalid) – Jack Wins Remix	Ed Sheeran	67

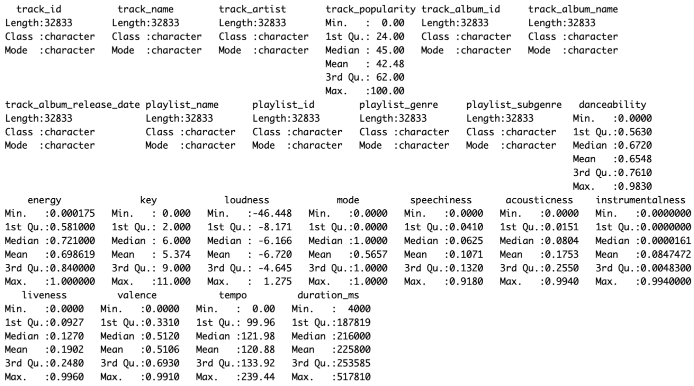
6 rows | 1-5 of 23 columns

Figure 2: spotify data

The **head()** function in R is used to display the first few rows of a data frame, matrix, or vector. By default, it shows the first 6 rows, but you can specify a different number of rows if needed.
There are a lot of embedded functions useful for dataframe operations in R, like **nrow**, **ncol**, **colnames**, etc.

Summary() function

```r
summary(df)
```



```
  track_id           track_name         track_artist       track_popularity track_album_id     track_album_name  
 Length:32833       Length:32833       Length:32833       Min.   :  0.00   Length:32833       Length:32833      
 Class :character   Class :character   Class :character   1st Qu.: 24.00   Class :character   Class :character  
 Mode  :character   Mode  :character   Mode  :character   Median : 45.00   Mode  :character   Mode  :character  
                                                          Mean   : 42.48                                        
                                                          3rd Qu.: 62.00                                        
                                                          Max.   :100.00                                        
 track_album_release_date playlist_name      playlist_id        playlist_genre     playlist_subgenre   danceability   
 Length:32833             Length:32833       Length:32833       Length:32833       Length:32833       Min.   :0.0000  
 Class :character         Class :character   Class :character   Class :character   Class :character   1st Qu.:0.5630  
 Mode  :character         Mode  :character   Mode  :character   Mode  :character   Mode  :character   Median :0.6720  
                                                                                                      Mean   :0.6548  
                                                                                                      3rd Qu.:0.7610  
                                                                                                      Max.   :0.9830  
     energy              key            loudness            mode         speechiness      acousticness    instrumentalness  
 Min.   :0.000175   Min.   : 0.000   Min.   :-46.448   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000000  
 1st Qu.:0.581000   1st Qu.: 2.000   1st Qu.: -8.171   1st Qu.:0.0000   1st Qu.:0.0410   1st Qu.:0.0151   1st Qu.:0.0000000  
 Median :0.721000   Median : 6.000   Median : -6.166   Median :1.0000   Median :0.0625   Median :0.0804   Median :0.0000161  
 Mean   :0.698619   Mean   : 5.374   Mean   : -6.720   Mean   :0.5657   Mean   :0.1071   Mean   :0.1753   Mean   :0.0847472  
 3rd Qu.:0.840000   3rd Qu.: 9.000   3rd Qu.: -4.645   3rd Qu.:1.0000   3rd Qu.:0.1320   3rd Qu.:0.2550   3rd Qu.:0.0048300  
 Max.   :1.000000   Max.   :11.000   Max.   :  1.275   Max.   :1.0000   Max.   :0.9180   Max.   :0.9940   Max.   :0.9940000  
    liveness          valence           tempo         duration_ms    
 Min.   :0.0000   Min.   :0.0000   Min.   :  0.00   Min.   :  4000  
 1st Qu.:0.0927   1st Qu.:0.3310   1st Qu.: 99.96   1st Qu.:187819  
 Median :0.1270   Median :0.5120   Median :121.98   Median :216000  
 Mean   :0.1902   Mean   :0.5106   Mean   :120.88   Mean   :225800  
 3rd Qu.:0.2480   3rd Qu.:0.6930   3rd Qu.:133.92   3rd Qu.:253585  
 Max.   :0.9960   Max.   :0.9910   Max.   :239.44   Max.   :517810  
```

Figure 3: spotify data summary

Retrieve values from data frame

In R, you can use square brackets [] to retrieve values from a data frame. The general syntax for using square brackets with data frames is `dataframe[row, column]`.

- Retrieve a single value

```
df[2,3] # Retrieves the value in the 2nd row and 3rd column
```

```
## [1] "Maroon 5"
```

```
df[1, 'track_name'] # Retrieves the value using column name
```

```
## [1] "I Don't Care (with Justin Bieber) - Loud Luxury Remix"
```

- Retrieve entire row or column

```
df[2,]
```

- Retrieve multiple columns or columns

```
rows <- df[c(1, 3, 5), ]  # Retrieves the 1st, 3rd, and 5th rows
```

- Using logical vectors

```
df[df$track_popularity==100,'track_name']
```

```
## [1] "Dance Monkey" "Dance Monkey"
```

Broadcasting in R

Calculation in R use a Broadcasting method. Broadcasting allows R to perform element-wise operations between vectors and scalars (single numbers) by implicitly expanding the scalar to match the length of the vector. (like **numpy** in python)

- Addition and Subtraction

```r
vec <- c(1, 2, 3, 4)
num <- 5
vec + num # Output: 6 7 8 9
```

- Multiplication and Division

```r
vec * num # Output: 5 10 15 20
```

- Exponentiation

```r
vec ^ num # Output: 0.2 0.4 0.6 0.8
```

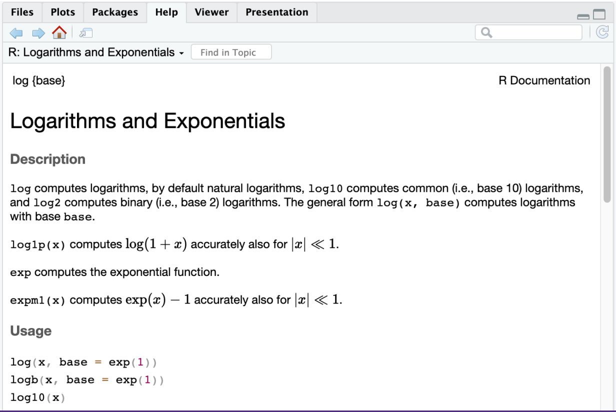
- Logical operation

```r
vec > num # Output: FALSE FALSE FALSE FALSE
```

Functions in R

Functions in R are blocks of code designed to perform a specific task. They take inputs, process them, and return an output.Functions allow you to modularize your code.
To get information about a function you know exists, use **help** or **?** or simply search in Rstudio help panel!

```r
?(log)
help(log)
```



log {base} R Documentation

Logarithms and Exponentials

Description

`log` computes logarithms, by default natural logarithms, `log10` computes common (i.e., base 10) logarithms, and `log2` computes binary (i.e., base 2) logarithms. The general form `log(x, base)` computes logarithms with base `base`.

`log1p(x)` computes $\log(1 + x)$ accurately also for $|x| \ll 1$.

`exp` computes the exponential function.

`expm1(x)` computes $\exp(x) - 1$ accurately also for $|x| \ll 1$.

Usage

```r
log(x, base = exp(1))
logb(x, base = exp(1))
log10(x)
```

Write your own function

```
# A simple function to add two numbers
add_numbers <- function(a, b) {
  a <- a + b
  a
}

# Using the function
a <- 5
b <- 3
sum <- add_numbers(a, b)
print(sum)  # Output: 8
print(a) # Output: 5
```

Key Points:

- Functions are defined using the `function` keyword.
- Inputs (arguments) are passed within the parentheses.
- Functions in R are (roughly) pass-by-value and not pass-by-reference. This means that if you modify an argument inside the function it will not change the original value outside the function.
- The last `a` is the output of the function. You can also use `return(a)`.

Conditional Statements in R

```r
# A function to check if a number is positive, negative, or zero
check_number <- function(num) {
  if (num > 0) {
    return("Positive")
  } else if (num < 0) {
    return("Negative")
  } else {
    return("Zero")
  }
}

# Using the function
result <- check_number(-5)
print(result)  # Output: "Negative"
```

For loop

```r
# A function to print numbers from 1 to n
print_numbers <- function(n) {
  for (i in 1:n) {
    print(i)
  }
}

# Using the function
print_numbers(3)
```

```
## [1] 1
## [1] 2
## [1] 3
```

While loop

```r
# A function to print numbers while a condition is true
print_numbers_while <- function(n) {
  i <- 1
  while (i <= n) {
    print(i)
    i <- i + 1
  }
}

# Using the function
print_numbers_while(3)
```

```
## [1] 1
## [1] 2
## [1] 3
```

Packages in R

R packages are collections of R functions, data, and compiled code that are stored in a well-defined format. Think of a package as a toolbox, where each tool is designed to help you accomplish a specific task. Whether you're performing data analysis, creating visualizations, or developing machine learning models, there's likely an R package that can help you get the job done more efficiently.
To use a package, you first need to install it. The simplest way to do this is through the Comprehensive R Archive Network, or CRAN, which is the official repository for R packages.

```r
# Install the package (you only need to do this once)
install.packages('ggplot2')

# Load the package into your R session
library(ggplot2)
```

After load the package, freely use any objects in the library!

Additional learning materials for R

I have only covered basic parts of R. There are a lot more to explore! (ggplot2, tidyverse). There are a lot of useful learning materials for you to boost your R skills and, more importantly, be prepared for the final project!

- R bootcamp held usually held before each fall semester `https://berkeley-scf.github.io/r-bootcamp-fall-2023/schedule`
- Swirl - an excellent interactive learning way! `https://swirlstats.com/students.html`
- R for Data Science: free online book `https://r4ds.hadley.nz/`
- TidyTuesday: real-world data scenarios `https://github.com/rfordatascience/tidytuesday`

Table of Contents