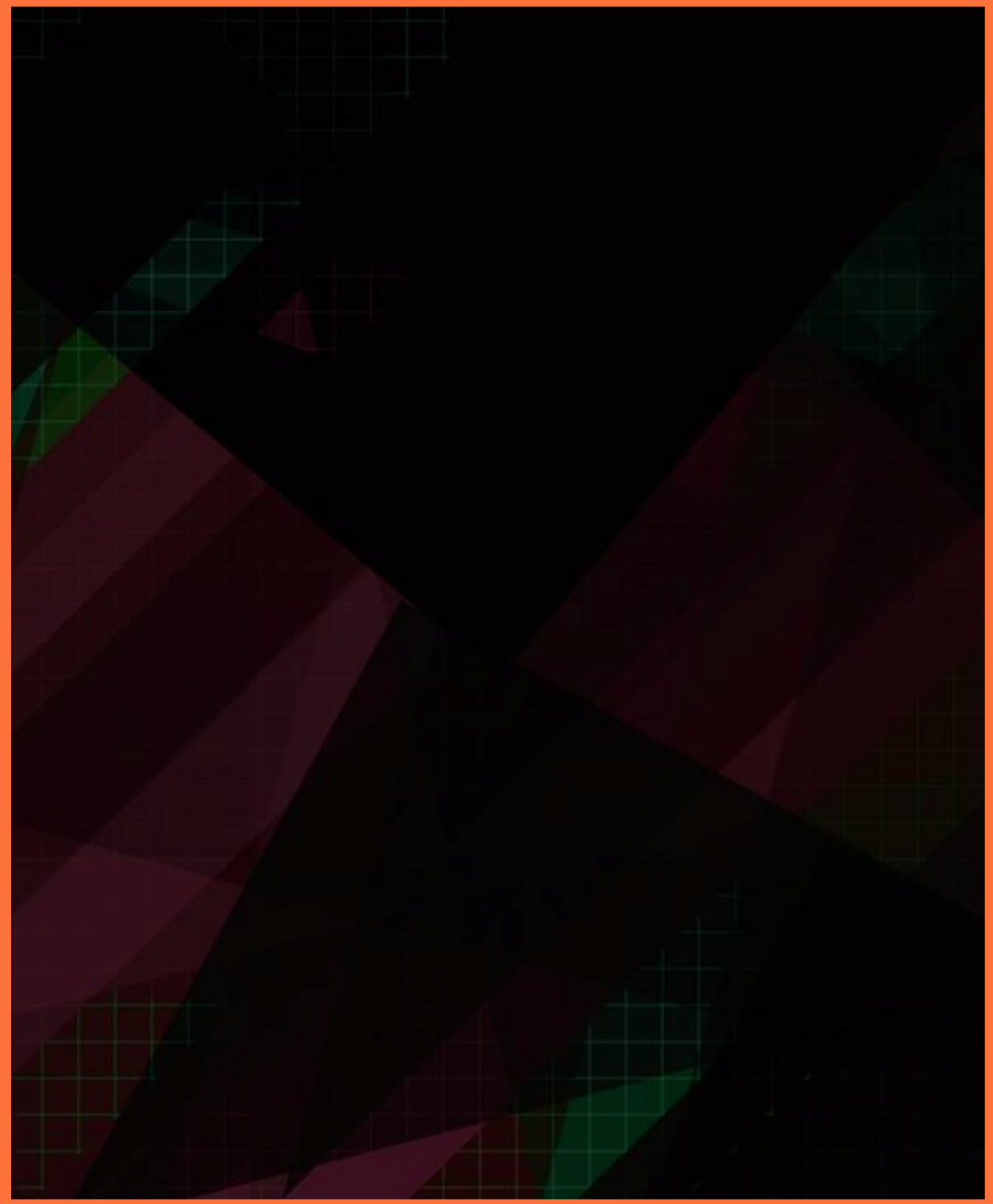
Practice IV. Work in groups. Functions to compute sectorial indices



DRA. ANNA BOQUÉ CIURANA

DR. JON XAVIER OLANO POZO

Objectives of Practice IV

Apply R programming to compute sectorial indices.

Work collaboratively in small groups.

Learn how to use both package-based and custom R functions.

Why Use R for This Analysis?

R IS OPEN-SOURCE, REPRODUCIBLE, AND COMMUNITY-SUPPORTED.

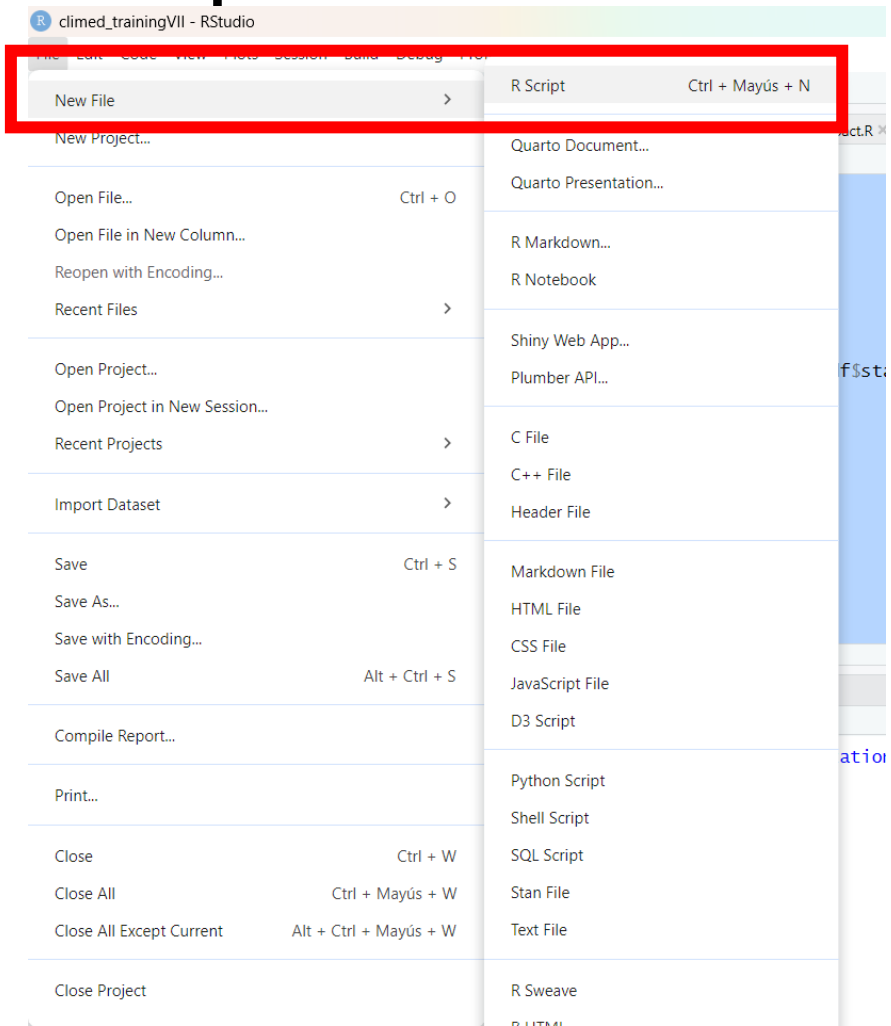
IDEAL FOR WORKING WITH TIME SERIES, SPATIAL DATA, AND CLIMATE INDICATORS.

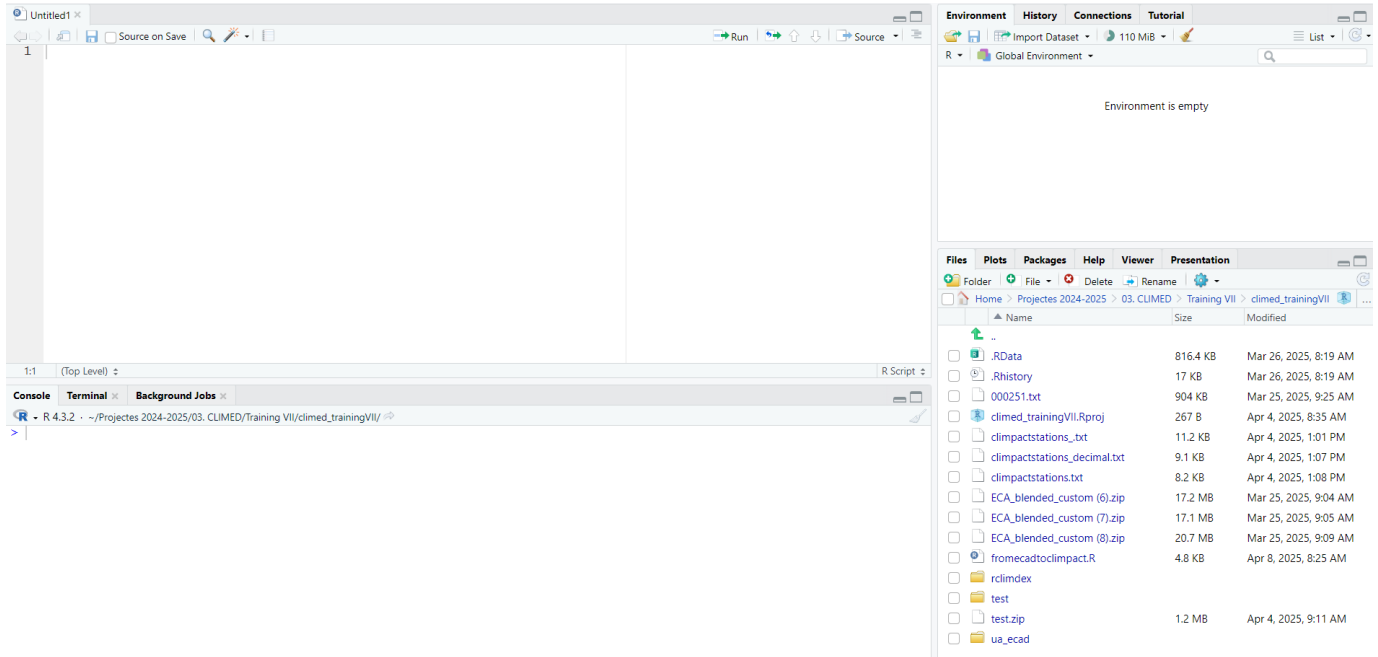
COMPATIBLE WITH CLIMATE-SPECIFIC PACKAGES AND GENERAL DATA ANALYSIS TOOLS.

Hands-on practice

1. Install and load packages
2. Load and prepare data
3. Apply a predefined function
4. Create a custom function
5. Visualization of the results

0. Prepare R-Studio

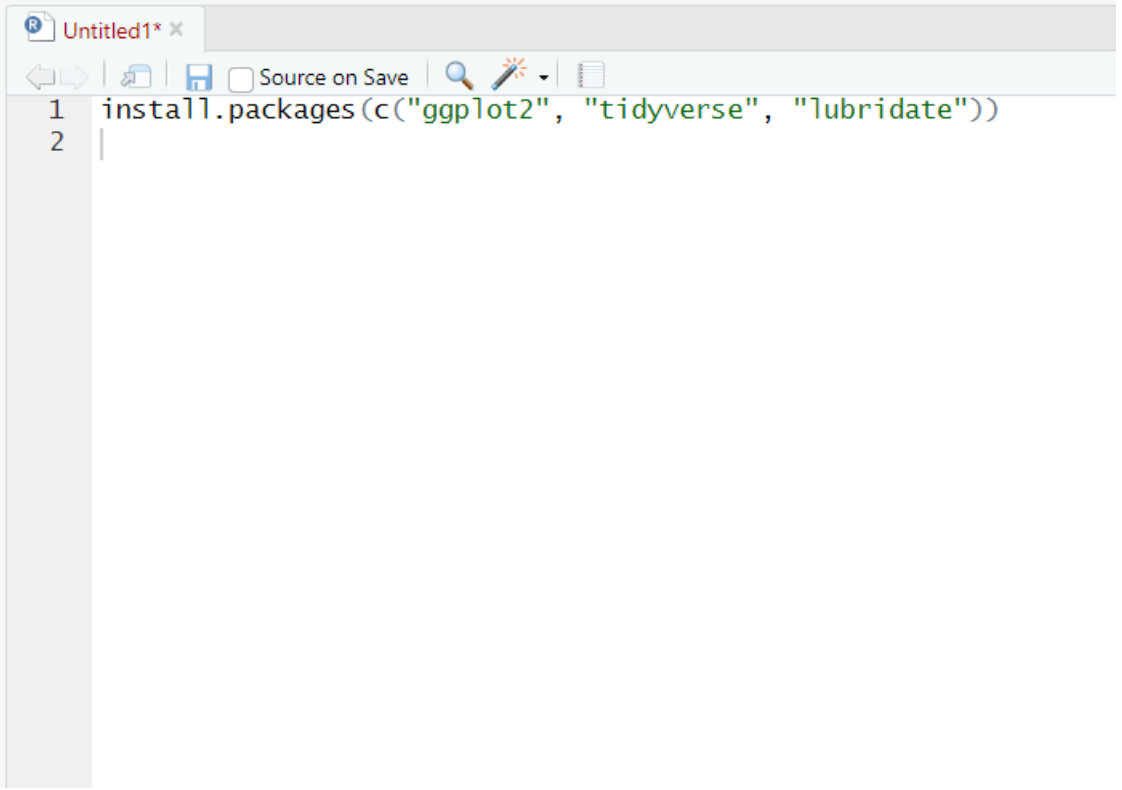




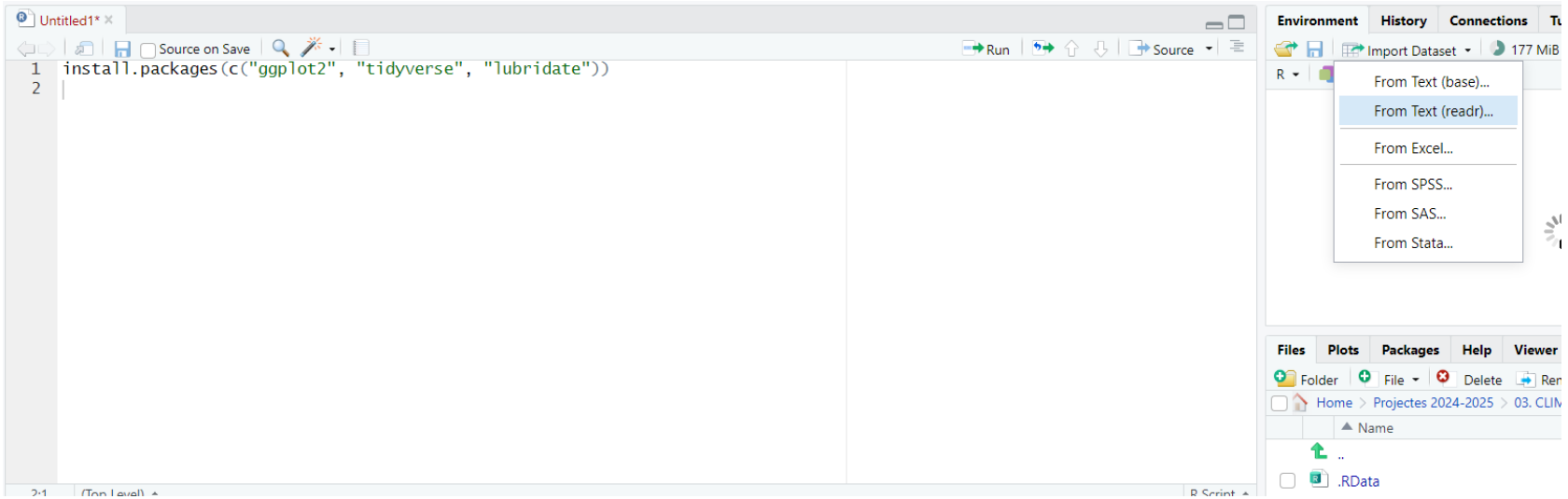
1. Install and load packages

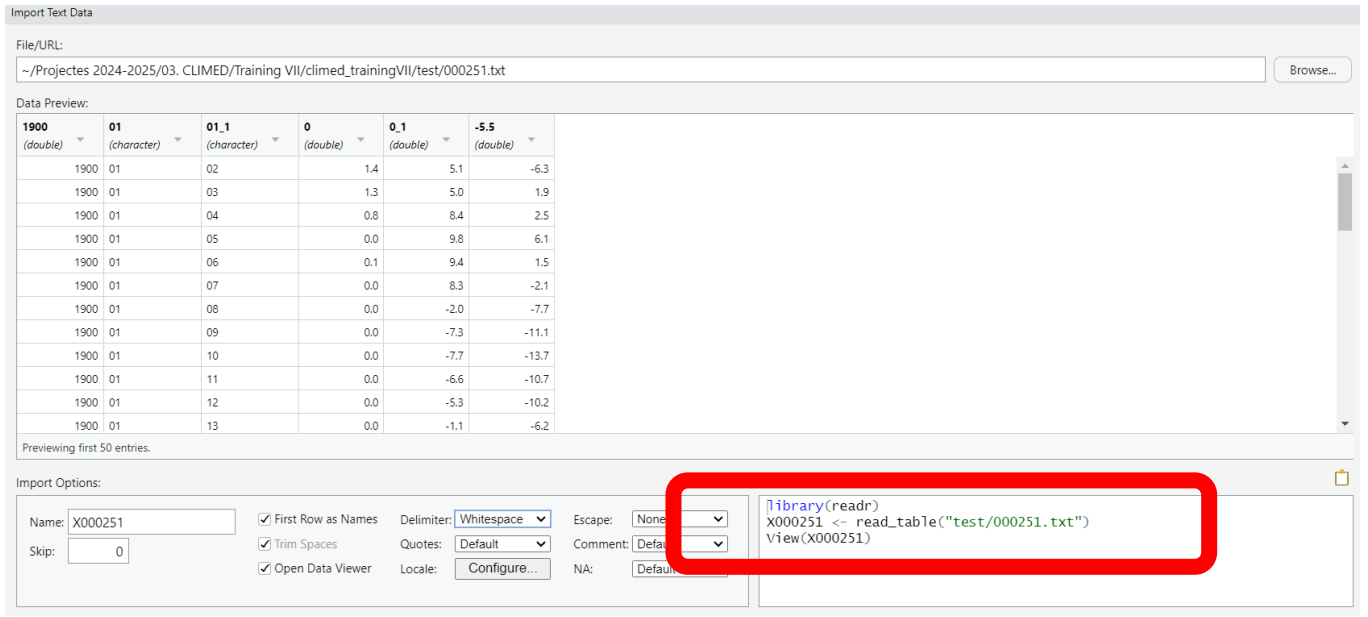
- # Install packages
- install.packages(c("tidyverse", "lubridate"))

- # Load librarys
- library(tidyverse)
- library(lubridate)



2. Load Data






```
library(ggplot2)
library(tidyverse)
library(lubridate)
library(readr)
climate_data<- read_table("test/000251.txt")
View(climate_data)
```

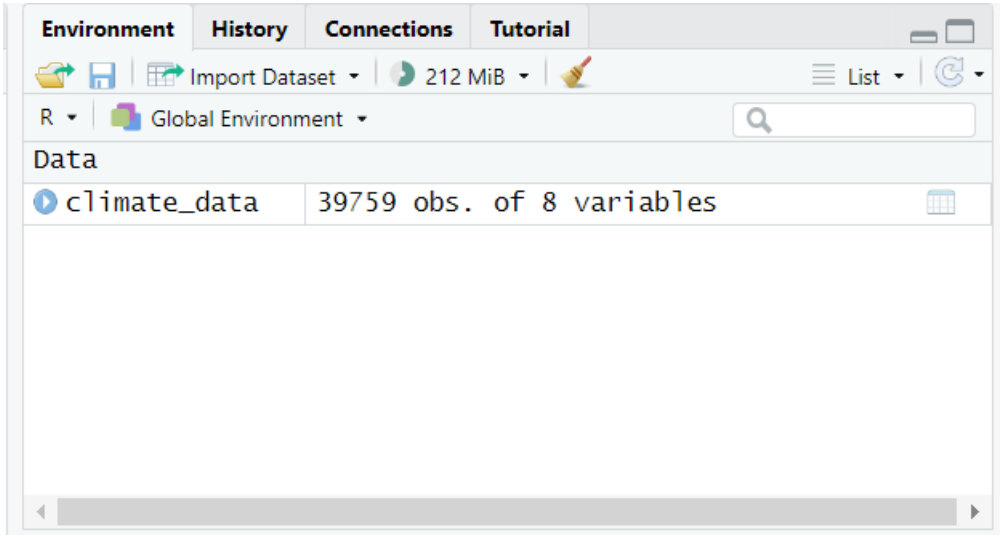

```
4 library(readr)
5 climate_data<- read_table("test/000251.txt")
6 View(climate_data)
```

3. Apply predefined function

- 3.1 Prepare data

```
 9  View(climate_data)
10
11
12  climate_data$year <- climate_data[[1]]
13  climate_data$tx   <- climate_data[[5]]
14
```

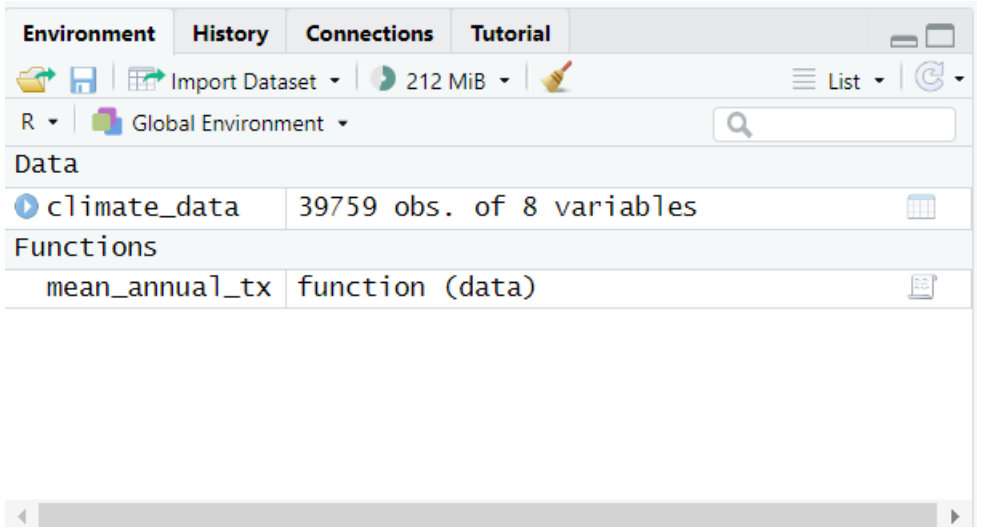
- # Extract year and TX
- climate_data$year <- climate_data[[1]]
  # Column 1 = year
- climate_data$tx <- climate_data[[5]]
  # Column 5 = TX



3. Apply predefined function

- 3.2 Create function for yearly mean and apply
- mean_annual_tx <- function(data) {
- data %>%
- group_by(year) %>%
- summarise(mean_tx = mean(tx, na.rm = TRUE))
- }
- #Apply function
- annual_avg_tx <- mean_annual_tx(climate_data)
- # See result
- print(annual_avg_tx)

```
16 mean_annual_tx <- function(data) {
17   data %>%
18     group_by(year) %>%
19     summarise(mean_tx = mean(tx, na.rm = TRUE))
20 }
```



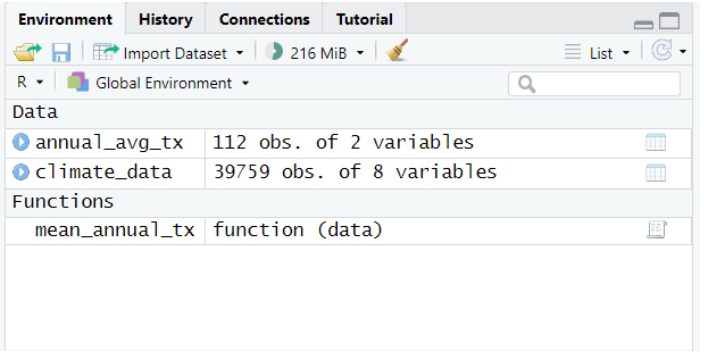

```
22 # Apply
23 annual_avg_tx <- mean_annual_tx(climate_data)
24
25 # See
26 print(annual_avg_tx)
```

Console | Terminal | Background Jobs

R 4.3.2 · ~/Projectes 2024-2025/03. CLIMED/Training VII/climed_trainingVII/

A tibble: 112 × 2

	year	mean_tx
	<dbl>	<dbl>
1	1900	15.6
2	1901	15.8
3	1902	14.8
4	1903	15.7
5	1904	15.0
6	1905	15.6
7	1906	15.8
8	1907	14.2
9	1908	14.7
10	1909	16.0

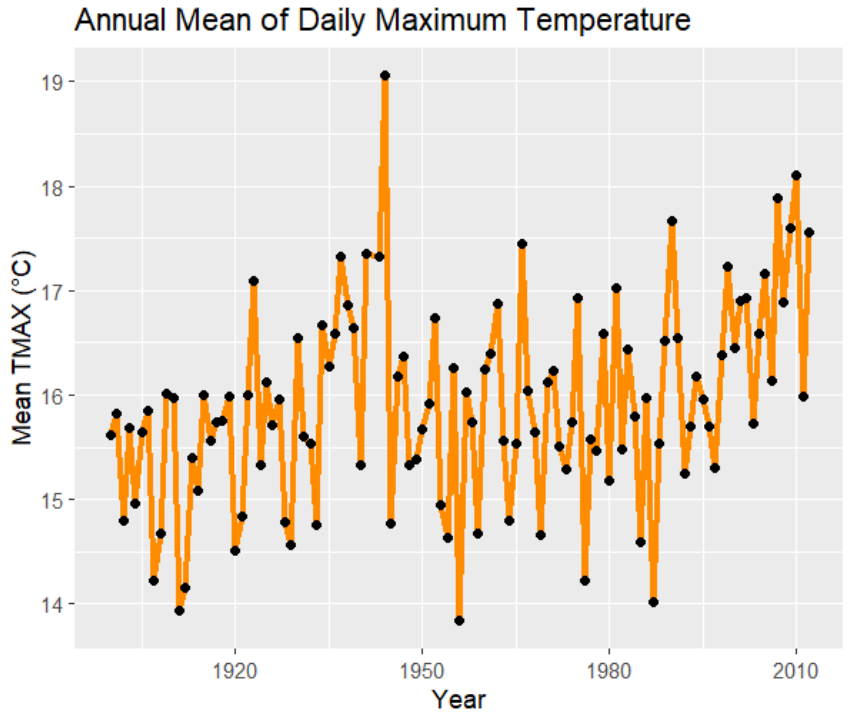


3. Apply predefined function

- 3.2 Visualize
- library(ggplot2)
- ggplot(annual_avg_tx, aes(x = year, y = mean_tx)) +
- geom_line(color = "darkorange", size = 1.2) +
- geom_point() +
- labs(title = "Annual Mean of Daily Maximum Temperature",
- x = "Year", y = "Mean TMAX (°C)")

```
30  ggplot(annual_avg_tx, aes(x = year, y = mean_tx)) +
31    geom_line(color = "darkorange", size = 1.2) +
32    geom_point() +
33    labs(title = "Annual Mean of Daily Maximum Temperature",
34         x = "Year", y = "Mean TMAX (°C)")
35
```

24:1 (Top Level)



4. Sectorial functions

- 1. Agriculture: Dry Spells

```
climate_data$prec <- climate_data[[4]]


count_dry_spells <- function(prec, threshold = 1, min_length = 5) {
  rle_obj <- rle(prec < threshold)  # TRUE si precipitación < 1 mm
  sum(rle_obj$values & rle_obj$lengths >= min_length)
}
dry_spells <- count_dry_spells(climate_data$prec)

# Mostrar resultado
print(paste("Number of dry spells:", dry_spells))
```

4. Sectorial functions

- 1. Agriculture: Dry Spells

```
climate_data$tmax <- climate_data[[5]]

count_hot_days <- function(temp, threshold = 35) {
  sum(temp > threshold, na.rm = TRUE)
}

hot_days <- count_hot_days(climate_data$tmax)


print(paste("Number of hot days:", hot_days))
```

4. Custom functions

- rain_mm <- function(rain_mm) { if (rain_mm > 50) {
  return(5) } else if (rain_mm >= 31 && rain_mm <= 50) {
  return(4) } else if (rain_mm >= 21 && rain_mm <= 30) {
  return(3) } else if (rain_mm >= 11 && rain_mm <= 20) {
  return(2) } else if (rain_mm >= 0 && rain_mm <= 10) {
  return(1) } else {   return(0) }}
- climate_data$prec <- climate_data[[4]]

- mm<-sapply(climate_data$prec ,FUN=cp_rain_mm)