Studio 3 Correlation: Climatology
18.05, Spring 2025

Overview of the studio

This studio is the first of the three that explore climate change through historical data. In this studio, we familiarize ourselves with the data, and as a warm-up, we analyze the temperature of all over Earth at 2° resolution with monthly averages from 1940 to 2024.

R introduced in this studio

In this studio we will use the functions

`var(), cor(), apply(), tapply()`

You can look up the documentation for each in the help window or by entering `?var`, etc., on the R console.

Download the zip file

You should have downloaded the studio3 zip file from our Canvas site, unzip it in your 18.05 studio folder, open `studio3.r`, and set the working directory to the folder. The folder also contains the data file `t2m_data.nc`.

Detailed instructions for the studio

Global temperature data is provided in a specific format, covering a grid of areas on Earth. These areas are determined by their latitude and longitude: latitude values range from $-90°$ to $90°$ and longitude values range from $0°$ to $360°$. Each cell measures $2° \times 2°$. For each cell, we the average temperature per month from January 1940 to November 2024. (The original data had 0.25° resolution, which is too large a file size for Studio.)

Problem 1. We examine the monthly and yearly correlation between the temperatures at two different points of Earth.

Problem 1a. Here you will finish the code for the function

`studio3_problem_1a(locations)`

This function should compute the *monthly* temperature averages for the places given by `locations`, plot them, and compute their correlation.

The argument of this function is

`locations` = a data frame containing the information of two (or more) locations.

- An example of locations (which is also in the test file) is as follows:

```
boston_and_sydney = data.frame(
  labels = c("Boston", "Sydney"),
  lats = c(42.36, -33.87),
  lons = c(288.94, 151.21),
  colors = c("blue", "red")
)
```

This data frame corresponds to the table

labels	lats	lons	colors
Boston	42.36	288.94	blue
Sydney	-33.87	151.21	red

You can use the `extract_data` function to extract the temperatures for the provided locations. This function searches for the closest location on the grid to the provided latitude and longitudes, and returns a matrix, where columns correspond to the locations and rows are available dates (January 1940 till November 2024). You can look at the variable `time` to see the corresponding time to a row (e.g., `time[10]` is the time corresponding to the tenth row of the matrix).

To compute the monthly averages (that is, average temperature for each month over all years), you have to group the dates by the month, and compute the mean of each group. You can do this using the `months` array, along with the `tapply()` function.

Problem 1b. Here you will finish the code for the function
`studio3_problem_1b(locations)`

This function should compute the *yearly* temperature averages for the places given by `locations`, plot them, and compute their correlation. All details are exactly the same as problem 1a. You can use the `years` array with `tapply()` to compute the yearly average.

Problem 2. In this problem, we compare the climatology of a city to the rest of the world, draw some nice plots on the surface of the Earth, and enjoy looking at them (and maybe check your intuition).

Problem 2a. Here you will finish the code for the function
`studio2_problem_2a(label, lat, lon)`

Arguments:
`label` = name of the city you are interested in, e.g., `"Boston"`
`lat` = the latitude of the location
`lon` = the longitude of the location

This function computes the *monthly* averages for all available locations on the grid, as well as a provided location in its arguments, and finds the correlation between that location and everywhere else.

Similar to Problem 1, we can use `tapply()` to compute the monthly averages. See the code for more instructions.

Problem 2b. Here you will finish the code for the function
`studio2_problem_2b(label, lat, lon)`

Arguments:
`label` = name of the city you are interested in, e.g., `"Boston"`

`lat` = the latitude of the location
`lon` = the longitude of the location

This function computes the *yearly* averages for all available locations on the grid, as well as a provided location in its arguments, and finds the correlation between that location and everywhere else.

Problem 2c. (Optional) Here, we compare Boston's monthly average to the places with the same latitude on Earth. You will finish the code for the function

`studio2_problem_2c()`

See the code for `extract_data()` as inspiration and find all cities on the same latitude as Boston. Using the same techniques as in Problem 2a, compute the correlation between monthly averages and plot them as a line plot.

Problem 3. Did the surface of Earth warm up uniformly between 1940 and "today"? Here, you will finish the code for the function

`studio2_problem_3()`

First, compute the yearly average of all earth between the years 1940 and 1949, as well as 2015 and 2024. Then, plot the difference in the average temperatures on the Earth as a heatmap. Observe that the temperature is not increasing uniformly everywhere. Are there places that are getting colder on average?

Testing your code

For each problem, we ran the problem function with certain parameters. You can see the function call and the output in studio3-test-answers.html. If you call the same function with the same parameters, you should get the same results as in studio3-test-answers.html – if there is randomness involved the answers should be close but not identical.

For your convenience, the file studio3-test.r contains all the function calls used to make studio3-test-answers.html.

Before uploading your code

1. Make sure all your code is in studio3.r. Also make sure it is all inside the functions for the problems.
2. Clean the environment and plots window.
3. Source the file.
4. Call each of the problem functions with the same parameters as the test file studio3-test-answers.html.
5. Make sure it runs without error and outputs just the answers asked for in the questions.
6. Compare the output to the answers given in studio3-test-answers.html.

Upload your code

Upload your code to Gradescope.

Leave the file name as studio3.r.

You can upload more than once. We will grade the last file you upload.

Due date

Due date: The goal is to upload your work by the end of class. If you need extra time, you can upload your work any time before 6 PM ET on the day of the studio (Friday).

Solutions uploaded: Solution code will be posted on Canvas at 4 AM the day after the studio.