CSC113 Final Project: Climate Change and Inferencing

London Gibson-Purcell

Fall 2024 | Data Science for the World

This project investigates data on climate change, i.e., long-term shifts in temperatures and weather patterns in the United States. By the end of the project, you should know how to:

1. Test whether observed data appear to be random samples from the same underlying distribution
2. Tidy a dataset using R and the `tidyverse` to prepare data for analysis
3. Apply inferencing in a case study where data were not randomly generated
4. Implement and interpret a hypothesis test
5. Generate and analyze visualizations, and then draw conclusions from them

Housekeeping

Rules. While collaboration is encouraged, sharing answers is never okay. In particular, posting code or other assignment answers publicly on Ed (or elsewhere) is academic dishonesty. It will result in a reduced project grade at a minimum. If you wish to ask a question that involves code, you *must* reach out to a TA or the instructor for help, either on Ed, during office hours, or by email.

All of the concepts necessary for this project can be found in the textbook or were discussed in lecture. **You may not use any coding/statistical techniques or conventions that have not been covered by the course**. We reserve the right to penalize projects that are not within scope. Please reach out to us if you have any doubts.

Grading & Due Date. Parts 1 through 3 of the project are required and contribute to the total project grade. Parts 4 and 5 can be used for extra credit; it does **NOT** need to be completed to earn full credit on the project. The entire project (parts 1, 2, 3, 4, and 5) is due on Gradescope by the end of the final exam period **Monday, December 9 at 10:30AM**.

Advice. Develop your answers incrementally. To perform a complicated task, break it up into steps, perform each step on a different line, give a new name to each result, and check that each intermediate result is what you expect. You can add as many additional names or functions as you need in the provided cells. **Start this project early and seek help early (either from the instructor, the TAs, etc.).**

On to the project!

Run the cell below to prepare the notebook. You may need to install additional packages (use `install.packages()`). The automated tests for this project **definitely do not** catch all possible errors; they're designed to help you avoid some common mistakes. Merely passing the tests does not guarantee full credit on any question.

In this project, we will investigate one of the 21st century's most prominent issues: climate change. While the details of climate science are beyond the scope of this course, we can start to learn about climate change just by analyzing public records of different cities' temperature and precipitation over time.

We will analyze a collection of historical daily temperature and precipitation measurements from weather stations in 210 U.S. cities. The dataset was compiled by Yuchuan Lai and David Dzombak [1]; a description of the data from the original authors and the data itself is also available.

[1] Lai, Yuchuan; Dzombak, David (2019): Compiled historical daily temperature and precipitation data for selected 210 U.S. cities. Carnegie Mellon University. Dataset.

Part I: Cities

Let us examine the information about the `cities`.

```
cities
```

```
## # A tibble: 461 × 7
##    Name      ID            Lat    Lon Stn.Name             Stn.stDate
Stn.edDate
##    <chr>     <chr>       <dbl>  <dbl> <chr>                <date>
<date>
##  1 Lander    USW00024021  42.8 -109.  LANDER WBO           1892-01-01
1946-05-28
##  2 Lander    USW00024021  42.8 -109.  LANDER HUNT FIELD    1946-05-29
2021-12-31
##  3 Cheyenne  USW00024018  41.2 -105.  CHEYENNE WBO         1871-01-01
1935-08-31
##  4 Cheyenne  USW00024018  41.2 -105.  CHEYENNE MUNICIPAL … 1935-09-01
2021-12-31
##  5 Wausau    USW00014897  44.9  -89.6 Wausau Record Herald 1896-01-01
1941-12-31
##  6 Wausau    USW00014897  44.9  -89.6 Wausau Airport       1942-01-01
2021-12-31
##  7 Hayward   USW00094973  46.0  -91.4 Hayward RS           1894-01-01
1998-03-31
##  8 Hayward   USW00094973  46.0  -91.4 Hayward Muni AP      1998-04-01
2021-12-31
##  9 EauClaire USW00014991  44.9  -91.5 Eau Claire City      1893-01-01
1949-09-30
## 10 EauClaire USW00014991  44.9  -91.5 Eau Claire Chippewa… 1949-10-01
2021-12-31
## # ℹ 451 more rows
```

The `cities` tibble has one row per weather station and the following columns:

1. `"Name"`: The name of the US city
2. `"ID"`: The unique identifier for the US city

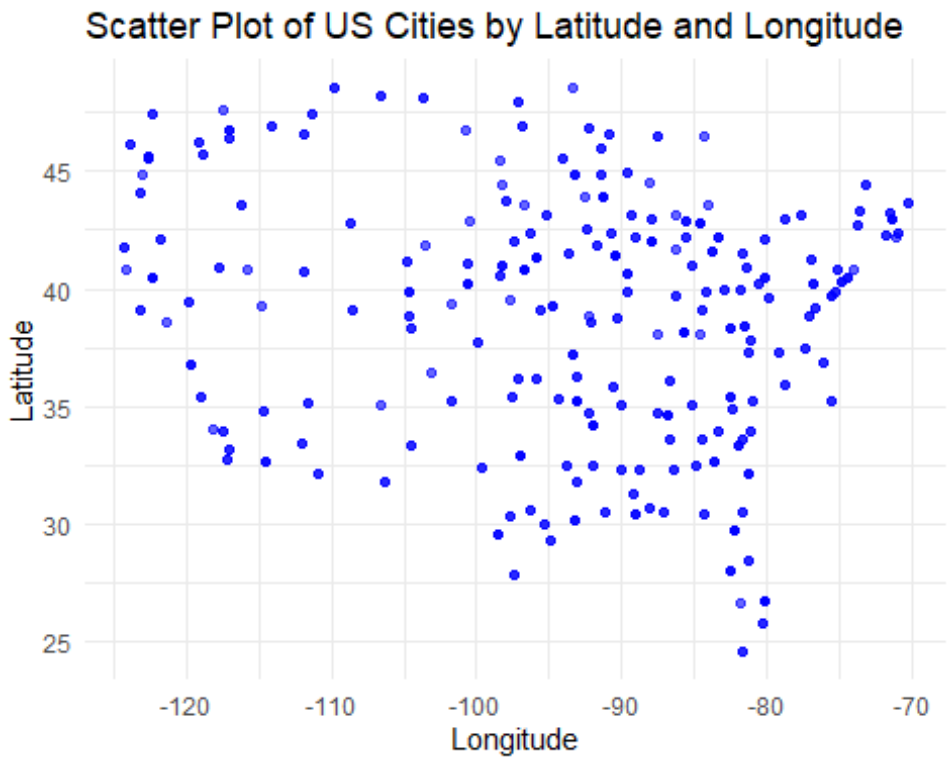
3. `"Lat"`: The latitude of the US city (measured in degrees of latitude)
4. `"Lon"`: The longitude of the US city (measured in degrees of longitude)
5. `"Stn.Name"`: The name of the weather station in which the data was collected
6. `"Stn.stDate"`: A string representing the date of the first recording at that particular station
7. `"Stn.edDate"`: A string representing the date of the last recording at that particular station

The data lists the weather stations at which temperature and precipitation data were collected. Note that although some cities have multiple weather stations, only one is collecting data for that city at any given point in time. Thus, we are able to just focus on the cities themselves.

Question 1.1: Generate a scatter plot that plots the latitude and longitude of every city in the `cities` tibble so that the result places northern cities at the top and western cities at the left. Note that the same point can be plotted multiple times.

Hint: A latitude is the set of horizontal lines that measures distances north or south of the equator. A longitude is the set of vertical lines that measures distances east or west of the prime meridian.

```
ggplot(cities, aes(x = Lon, y = Lat)) +
  geom_point(alpha = 0.6, color = "blue") +
  labs(
    title = "Scatter Plot of US Cities by Latitude and Longitude",
    x = "Longitude",
    y = "Latitude"
  ) +
  theme_minimal()
```



These cities are all within the continental U.S., and so the general shape of the U.S. should be visible in your plot. The shape will appear distorted compared to most maps for two reasons: the scatter plot is square even though the U.S. is wider than it is tall, and this scatter plot is an equirectangular projection of the spherical Earth. A geographical map of the same data uses the common Pseudo-Mercator projection.

Run the following cell to see this in action:

```
leaflet(states) |>
  addTiles() |>
  addMarkers(data=cities, ~Lon, ~Lat, popup = ~as.character(Name))
```

Question 1.2: Do these city locations appear to be sampled uniformly at random from all the locations in the United States? Briefly explain your answer.

No, there appears to be more weather stations on the eastern part of the country which has a higher populations density. The stations to the west seemed to be less frequent compared to the amount of the space available.

Question 1.3: Assign `num_unique_cities` to the number of unique cities that appear in the `cities` tibble. You should use a `dplyr` verb to help answer this.

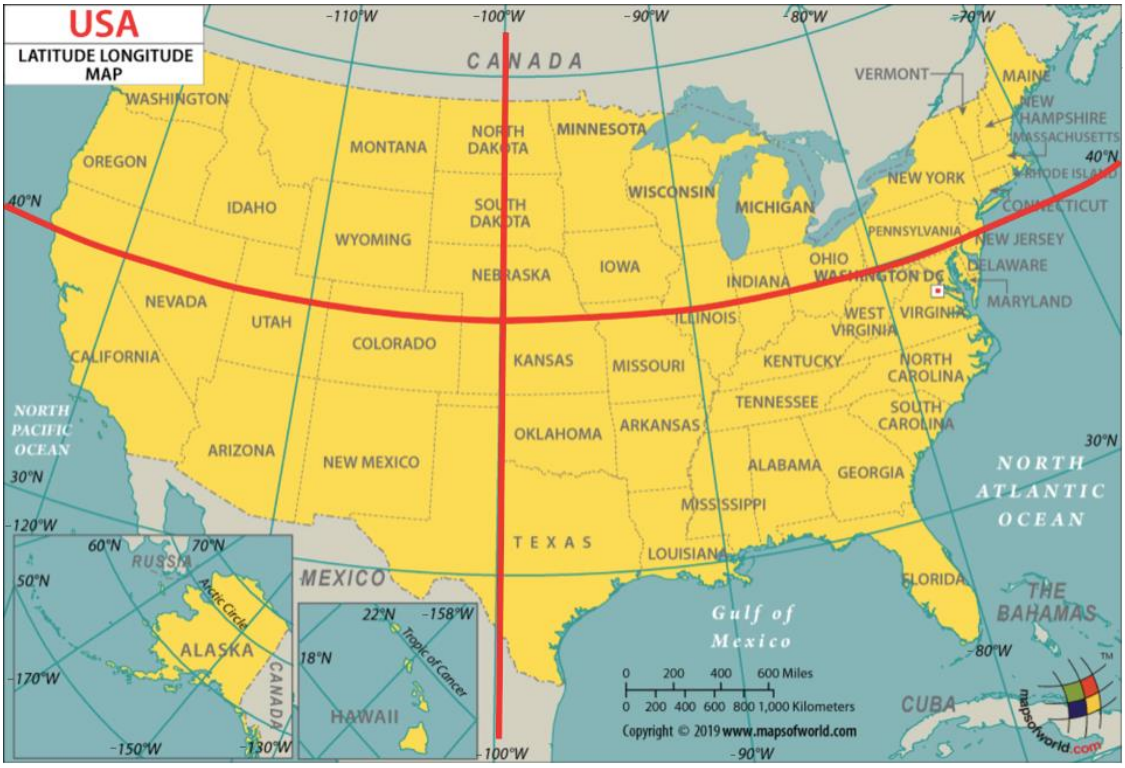
```
num_unique_cities <- cities |>
  summarise(unique_cities = n_distinct(Name)) |>
  pull(unique_cities)
num_unique_cities
```

```
## [1] 210

. = ottr::check("tests/part1_q3.R")

## All tests passed!
```

In order to investigate further, it will be helpful to determine what region of the United States each city was located in: Northeast, Northwest, Southeast, or Southwest. Let us use the following geographical boundaries:



- A station is located in the `"Northeast"` region if its latitude is above or equal to 40 degrees and its longitude is greater than or equal to -100 degrees.
- A station is located in the `"Northwest"` region if its latitude is above or equal to 40 degrees and its longitude is less than -100 degrees.
- A station is located in the `"Southeast"` region if its latitude is below 40 degrees and its longitude is greater than or equal to -100 degrees.
- A station is located in the `"Southwest"` region if its latitude is below 40 degrees and its longitude is less than -100 degrees.

Question 1.4: Define the function `convert_coordinates` below. It should receive two arguments, a city's latitude (`lat`) and longitude (`lon`) coordinates, and return a string representing the region it is located in.

```
convert_coordinates <- function(lat, lon) {
  ifelse(lat >= 40 & lon >= -100, "Northeast",
         ifelse(lat >= 40 & lon < -100, "Northwest",
                ifelse(lat < 40 & lon >= -100, "Southeast",
```

```
                        "Southwest")))
}

. = ottr::check("tests/part1_q4.R")
```

```
## All tests passed!
```

Question 1.5: Using the function you created (`convert_coordinates`), add a new column in `cities` named `Region` that contains the region in which the city is located. Assign the resulting tibble back to the name `cities`.

```
cities <- cities |>
  mutate(Region = convert_coordinates(Lat, Lon))
cities
```

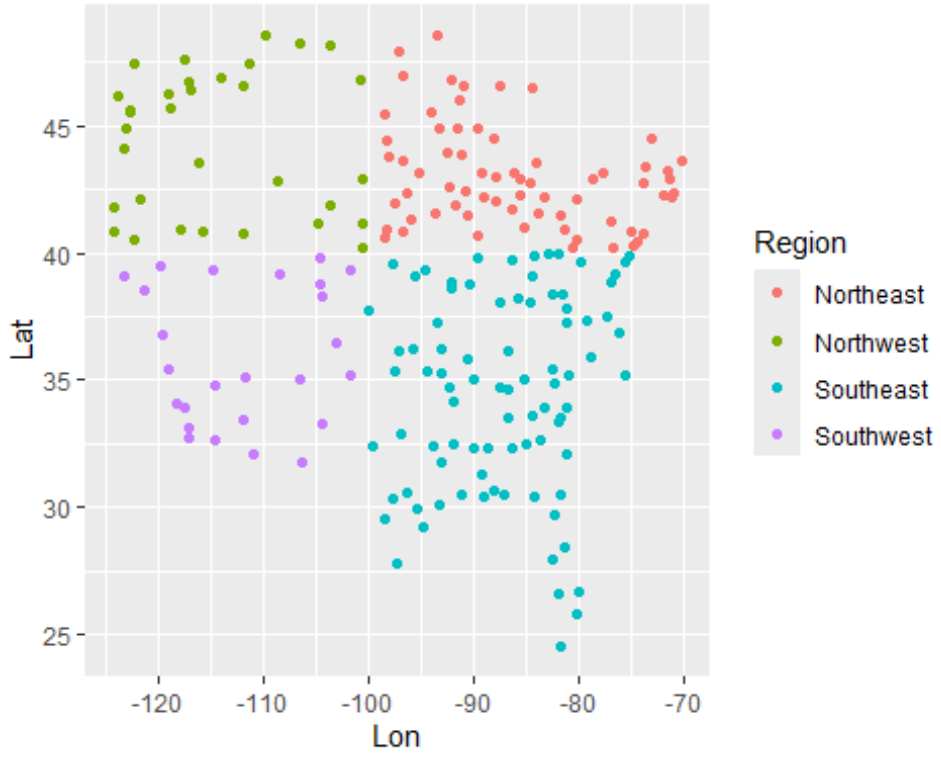
```
## # A tibble: 461 × 8
##    Name      ID            Lat    Lon Stn.Name      Stn.stDate Stn.edDate
Region
##    <chr>     <chr>       <dbl>  <dbl> <chr>         <date>     <date>
<chr>
##  1 Lander    USW00024021  42.8 -109.  LANDER WBO    1892-01-01 1946-05-28
North…
##  2 Lander    USW00024021  42.8 -109.  LANDER HUNT … 1946-05-29 2021-12-31
North…
##  3 Cheyenne  USW00024018  41.2 -105.  CHEYENNE WBO  1871-01-01 1935-08-31
North…
##  4 Cheyenne  USW00024018  41.2 -105.  CHEYENNE MUN… 1935-09-01 2021-12-31
North…
##  5 Wausau    USW00014897  44.9  -89.6 Wausau Recor… 1896-01-01 1941-12-31
North…
##  6 Wausau    USW00014897  44.9  -89.6 Wausau Airpo… 1942-01-01 2021-12-31
North…
##  7 Hayward   USW00094973  46.0  -91.4 Hayward RS    1894-01-01 1998-03-31
North…
##  8 Hayward   USW00094973  46.0  -91.4 Hayward Muni… 1998-04-01 2021-12-31
North…
##  9 EauClaire USW00014991  44.9  -91.5 Eau Claire C… 1893-01-01 1949-09-30
North…
## 10 EauClaire USW00014991  44.9  -91.5 Eau Claire C… 1949-10-01 2021-12-31
North…
## # ℹ 451 more rows
```

```
. = ottr::check("tests/part1_q5.R")
```

```
## All tests passed!
```

To confirm that you've defined your `convert_coordinates` function correctly and successfully added the `Region` column to the cities table, run the following cell. Each region should have a different color in the result.

```
ggplot(cities) +
  geom_point(aes(x = Lon, y = Lat, color = Region))
```



Part II: Welcome to Needles, California

Each city has a different CSV file of daily temperature and precipitation measurements. The file for Needles, California is included with this project as `needles.csv`. The files for other cities can be downloaded here by matching them to the ID of the city in the `cities` tibble.

Needles is located in in the Mojave Desert region of Southern California and is known for its impressive temperatures.

Route 66 Sign in Needles, California

Run the following cell to view the `needles` tibble. It has one row per day and the following columns:

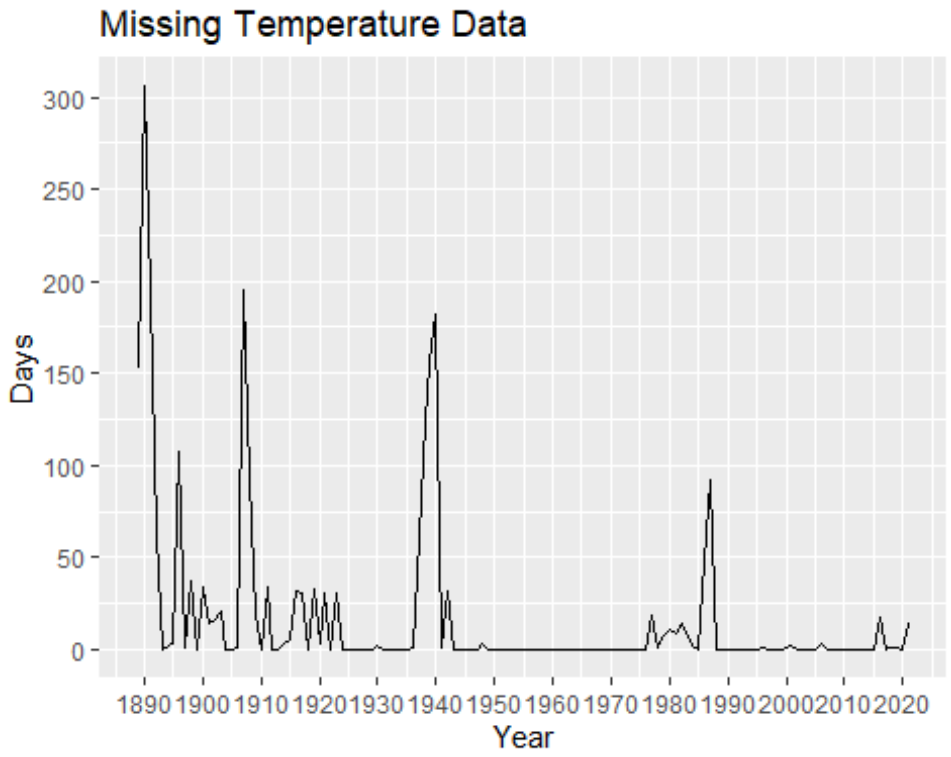
1. `"Date"`: The date (a string) representing the date of the recording in YYYY-MM-DD format
2. `"tmax"`: The maximum temperature for the day (°F)
3. `"tmin"`: The minimum temperature for the day (°F)
4. `"prcp"`: The recorded precipitation for the day (inches)

```
needles
```

```
## # A tibble: 48,577 × 4
##    Date        tmax  tmin  prcp
##    <date>     <dbl> <dbl> <dbl>
##  1 1889-01-01    55    42  0.08
##  2 1889-01-02    56    32  0
##  3 1889-01-03    56    36  0
##  4 1889-01-04    54    40  0.13
##  5 1889-01-05    62    44  0.2
##  6 1889-01-06    67    47  0.09
##  7 1889-01-07    66    50  0
##  8 1889-01-08    62    48  0
##  9 1889-01-09    63    46  0
## 10 1889-01-10    63    43  0
## # ℹ 48,567 more rows
```

According to the documentation, cities may miss substantial amounts of data during their periods of record. Let's check this for the Needles data. The following cell generates a line plot of the number of days with missing temperature data by year.

```
needles |>
  group_by(Year=year(Date)) |>
  summarize(Days=sum(is.na(tmax) | is.na(tmin))) |>
  ggplot() +
  geom_line(aes(x = Year, y = Days)) +
  scale_x_continuous(breaks = seq(1800, 2020, 10))+
  scale_y_continuous(breaks = seq(0, 365, 50)) +
  labs(title = "Missing Temperature Data")
```



Question 2.1: Does the data appear to have a missing value problem? Do the missing values appear to be concentrated around certain years or randomly distributed over time? Briefly describe your observations.

Yes, there are prolonged periods of time and large spikes where data is missing. This indicates that there was a consistent and concentrated issue with recording temperatures. Most of the missing data is from the 1800s which could be due to lack of proficient equipment resulting in lost data or errors. There are also several spikes which could be from troubling times in history like war periods and political instability where temperature recording wasn't funded or important at the time.

Question 2.2: One way to deal with missingness is to eliminate rows containing missing temperature data. If we can assume the missing entries are due to faulty equipment that fails randomly or another source of error that is unrelated to the data, this can be a plausible approach. Assign the name `needles_complete` to a copy of the data in `needles`, but omits daily entries that have missing temperature data (in either `tmax` or `tmin`).

```
needles_complete <- needles |>
  filter(!is.na(tmax) & !is.na(tmin))

needles_complete
```

```
## # A tibble: 46,298 × 4
##    Date        tmax  tmin  prcp
##    <date>     <dbl> <dbl> <dbl>
##  1 1889-01-01    55    42  0.08
```

```
##  2 1889-01-02    56    32  0
##  3 1889-01-03    56    36  0
##  4 1889-01-04    54    40  0.13
##  5 1889-01-05    62    44  0.2
##  6 1889-01-06    67    47  0.09
##  7 1889-01-07    66    50  0
##  8 1889-01-08    62    48  0
##  9 1889-01-09    63    46  0
## 10 1889-01-10    63    43  0
## # ℹ 46,288 more rows
```

```
. = ottr::check("tests/part2_q2.R")
```

```
## All tests passed!
```

Question 2.3: Assign the name `largest_2017_range_date` to the date of the largest temperature range in Needles, California for any day between January 1st, 2017 and December 31st, 2017. To accomplish this, first create a new tibble, say named `needles_with_ranges_2017`, that is a copy of `needles_complete` but contains only days in 2017 and has an additional column corresponding to the temperature range for a given day.

Note: Your answer should be a string in the "YYYY-MM-DD" format. Feel free to use as many lines as needed. A temperature range is calculated as the difference between the max and min temperatures for the day.

```
needles_with_ranges_2017 <- needles_complete |>
  filter(year(Date) == 2017) |>
  mutate(temperature_range = tmax - tmin)


largest_2017_range_date <- needles_with_ranges_2017 |>
  slice_max(order_by = temperature_range, n = 1) |>
  pull(Date) |>
  as.character()

largest_2017_range_date
```

```
## [1] "2017-07-08"
```

```
. = ottr::check("tests/part2_q3.R")
```

```
## All tests passed!
```

Let's take a look at the maximum temperature for that day.

```
needles_complete |>
  filter(Date == largest_2017_range_date)
```

```
## # A tibble: 1 × 4
##   Date        tmax  tmin  prcp
```

```
##   <date>     <dbl> <dbl> <dbl>
## 1 2017-07-08   120    81     0
```

YOWZA – that's hot!

The following function `get_year_from_date` takes a date string in the `"YYYY-MM-DD"` format and returns a double representing the year. The function `get_month_from_date` takes a date string and returns a string describing the month. Run this cell, but you do not need to understand how this code works or edit it.

```
get_year_from_date <- function(date) {
  year(date)
}

get_month_from_date <- function(date) {
  ymd(date) |> format("%m (%b)")
}

# Examples
str_c("2024-10-04 has year ", get_year_from_date("2024-10-04"))
```

```
## [1] "2024-10-04 has year 2024"
```

```
str_c("2024-10-04 has month ", get_month_from_date("2024-10-04"))
```

```
## [1] "2024-10-04 has month 10 (Oct)"
```

Question 2.4: Add two new columns called `Year` and `Month` to the `needles_complete` tibble that contain the year as a **double** and the month as a **string** (e.g., `"10 (Oct)"` for October) for each day, respectively.

```
needles_complete <- needles_complete |>
  mutate(
    Year = get_year_from_date(Date),
    Month = get_month_from_date(Date)
  )
needles_complete
```

```
## # A tibble: 46,298 × 6
##    Date        tmax  tmin  prcp  Year Month
##    <date>     <dbl> <dbl> <dbl> <dbl> <chr>
##  1 1889-01-01    55    42  0.08  1889 01 (Jan)
##  2 1889-01-02    56    32  0     1889 01 (Jan)
##  3 1889-01-03    56    36  0     1889 01 (Jan)
##  4 1889-01-04    54    40  0.13  1889 01 (Jan)
##  5 1889-01-05    62    44  0.2   1889 01 (Jan)
##  6 1889-01-06    67    47  0.09  1889 01 (Jan)
##  7 1889-01-07    66    50  0     1889 01 (Jan)
##  8 1889-01-08    62    48  0     1889 01 (Jan)
##  9 1889-01-09    63    46  0     1889 01 (Jan)
```

```
## 10 1889-01-10    63    43  0     1889 01 (Jan)
## # ℹ 46,288 more rows
```

```
. = ottr::check("tests/part2_q4.R")
```

```
## All tests passed!
```

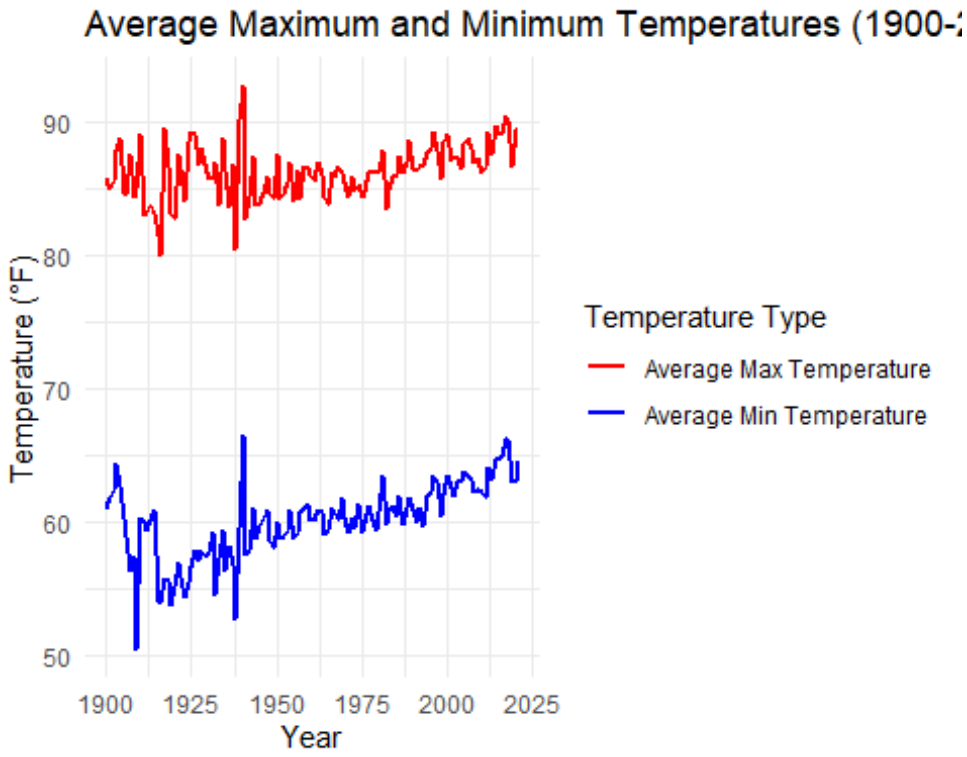
Question 2.5: Using the `needles_complete` tibble, generate an overlaid line plot of the average maximum temperature and average minimum temperature for each year between 1900 and 2021 (inclusive). The lines should be **colored differently to distinguish between minimum and maximum temperatures** and **a corresponding legend should be shown**. As for all visualizations, text should be legible, axes labeled appropriately, have a title, etc.

Hint: Before applying `ggplot`, `dplyr`/`tidyr` verbs should be used to transform the data into the appropriate shape and summarize the relevant information.

```
temperature_summary <- needles_complete |>
  filter(between(Year, 1900, 2021)) |>
  group_by(Year) |>
  summarize(
    avg_tmax = mean(tmax, na.rm = TRUE),
    avg_tmin = mean(tmin, na.rm = TRUE)
  )

ggplot(temperature_summary, aes(x = Year)) +
  geom_line(aes(y = avg_tmax, color = "Average Max Temperature"), size = 1) +
  geom_line(aes(y = avg_tmin, color = "Average Min Temperature"), size = 1) +
  scale_color_manual(values = c("red", "blue")) +
  labs(
    title = "Average Maximum and Minimum Temperatures (1900-2021)",
    x = "Year",
    y = "Temperature (°F)",
    color = "Temperature Type"
  ) +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Question 2.6: While still debated, many climate scientists agree that the effects of climate change began to surface in the early 1960s as a result of elevated levels of greenhouse gas emissions. Does the graph you produced in **Question 2.5** support the claim that modern-day global warming began in the early 1960s?

I would say the graph shows from around 1935-1940 when there was a big spike in temps and beyond there has been an upward trend in both max and min temperatures.

Averaging temperatures across an entire year can obscure some effects of climate change. For example, if summers get hotter but winters get colder, the annual average may not change much. Let's investigate how average **monthly maximum temperatures** have changed over time in Needles.

We will consider two time spans, the period from 1900-1960 (i.e., the `"Past"`) and the period from 2019-2021 (i.e., the `"Present"`). The following function converts a year, expressed as a double, to a string representing its corresponding period. A year not covered by either period is classified into a third category `"Other"`.

```
year_to_period <- function(year) {
  if(between(year, 1900, 1960)) {
    "Past"
  } else if (between(year, 2019, 2021)) {
    "Present"
  } else {
    "Other"
  }
```

```
}

year_to_period(2020) # example call

## [1] "Present"
```

Question 2.7: Create a `monthly_increases` tibble with one row per month and the following four columns in order:

1. `"Month"`: The month (such as `"10 (Oct)"`)
2. `"Past"`: The average **max temperature** in that month from 1900-1960 (both ends inclusive)
3. `"Present"`: The average **max temperature** in that month from 2019-2021 (both ends inclusive)
4. `"Increase"`: The difference between the present and past average max temperatures in that month

First, make a copy of the `needles_complete` tibble with a new column containing the corresponding **period** for each row. You may find the `year_to_period` function helpful for this. Then, use this new tibble to construct `monthly_increases`.

```
needles_with_periods <- needles_complete |>
  mutate(Period = sapply(Year, year_to_period))


needles_with_periods <- needles_with_periods |>
  mutate(Month = format(ymd(Date), "%m (%b)"))


all_months <- c("01 (Jan)", "02 (Feb)", "03 (Mar)", "04 (Apr)", "05 (May)",
"06 (Jun)",
                "07 (Jul)", "08 (Aug)", "09 (Sep)", "10 (Oct)", "11 (Nov)",
"12 (Dec)")

# Now create the monthly_increases tibble
monthly_increases <- needles_with_periods |>
  filter(Period %in% c("Past", "Present")) |>
  group_by(Month, Period) |>
  summarize(
    avg_max_temp = mean(tmax, na.rm = TRUE)
  ) |>
  spread(key = Period, value = avg_max_temp) |>
  mutate(
    Increase = Present - Past
  ) |>
  ungroup()

## `summarise()` has grouped output by 'Month'. You can override using the
## `.groups` argument.
```

```
monthly_increases <- monthly_increases |>
  complete(Month = all_months, fill = list(Past = NA, Present = NA, Increase
= NA))


monthly_increases
```

```
## # A tibble: 12 × 4
##    Month     Past Present Increase
##    <chr>    <dbl>   <dbl>    <dbl>
##  1 01 (Jan)  63.2    66.1    2.88
##  2 02 (Feb)  68.7    68.9    0.171
##  3 03 (Mar)  76.1    75.5   -0.621
##  4 04 (Apr)  85.2    88.9    3.68
##  5 05 (May)  93.9    95.7    1.76
##  6 06 (Jun) 103.    107.     3.95
##  7 07 (Jul) 108.    112.     3.80
##  8 08 (Aug) 105.    112.     6.41
##  9 09 (Sep)  99.4   103.     4.02
## 10 10 (Oct)  86.1    88.8    2.62
## 11 11 (Nov)  72.4    77.3    4.91
## 12 12 (Dec)  63.3    64.1    0.800
```

```
. = ottr::check("tests/part2_q7.R")
```

```
## All tests passed!
```

Part III: February in Needles

The `"Past"` column values are averaged over many decades, and so they are reliable estimates of the average high temperatures in those months before the effects of modern climate change. However, the `"Present"` column is based on only three years of observations. February, the shortest month, has the fewest total observations: only 85 days. Run the following cell to see this.

```
feb_present <- needles_complete |>
  filter(between(Year, 2019, 2021) & Month == "02 (Feb)")
feb_present
```

```
## # A tibble: 85 × 6
##    Date        tmax  tmin  prcp  Year Month
##    <date>     <dbl> <dbl> <dbl> <dbl> <chr>
##  1 2019-02-01  73    48.9  0     2019 02 (Feb)
##  2 2019-02-02  69.1  57    0.2   2019 02 (Feb)
##  3 2019-02-03  66    51.1  0     2019 02 (Feb)
##  4 2019-02-04  66    51.1  0.01  2019 02 (Feb)
##  5 2019-02-05  66    50    0     2019 02 (Feb)
##  6 2019-02-06  57    43    0     2019 02 (Feb)
##  7 2019-02-07  57.9  35.1  0     2019 02 (Feb)
##  8 2019-02-08  61    36    0     2019 02 (Feb)
##  9 2019-02-09  64    42.1  0     2019 02 (Feb)
```

```
## 10 2019-02-10  64.9  43    0     2019 02 (Feb)
## # ℹ 75 more rows
```

Look back to your `monthly_increases` tibble. The increase for the month of February is quite small; the February difference is close to zero. Run the following cell to print out our observed difference (in the `Increase` column).

```
monthly_increases |>
  slice(2)
```

```
## # A tibble: 1 × 4
##   Month     Past Present Increase
##   <chr>    <dbl>   <dbl>    <dbl>
## 1 02 (Feb)  68.7    68.9    0.171
```

Perhaps that small difference is somehow due to chance! Let's investigate this further.

We can observe all of the February maximum temperatures from 2019 to 2021 (the present period), so we have access to the census; there's no random sampling involved. But, we can imagine that if more years pass with the same present-day climate, there would be different but similar maximum temperatures in future February days. From the data we observe, we can try to estimate the **average maximum February temperature** in this imaginary collection of all future February days that would occur in our modern climate, assuming the climate doesn't change any further and many years pass.

We can also imagine that the maximum temperature each day is like a **random draw from a distribution of max daily temperatures for that month**. Treating actual observations of natural events as if they were each *randomly* sampled from some unknown distribution is a simplifying assumption. These temperatures were not actually sampled at random – instead they occurred due to the complex interactions of the Earth's climate – but treating them as if they were random abstracts away the details of this naturally occurring process and allows us to carry out statistical inference. Conclusions are only as valid as the assumptions upon which they rest, but in this case thinking of daily temperatures as random samples from some unknown climate distribution seems at least plausible.

If we assume that the **actual temperatures were drawn at random from some large population of possible February days** in our modern climate, then we can not only estimate the population average of this distribution, but also quantify our uncertainty about that estimate using a confidence interval.

We will now compute the confidence interval of the present February average max daily temperature. To unpack this statement, we are saying that this confidence interval represents present-day February conditions. We will compare this confidence interval to the historical average (i.e., the `"Past"` value in our `monthly_increases` tibble). How will we do the comparison? Since we are interested in seeing if the average February max daily temperatures have ***changed*** since the past, we care about whether the historical average lies within the confidence interval we create.

Based on the information above, think what the null hypothesis and alternative hypothesis are.

Question 3.1: Complete the implementation of the function `generate_conf_int`, which takes as arguments a given month `month` and a confidence `level` percentage such as 0.95 or 0.99. In this function, a tibble `sample_observations` is generated with corresponding sample observations from the `needles_complete` tibble (code provided). **It then returns a two-element vector containing the lower and upper bound in that order, representing a confidence interval** for the population mean constructed using 1,000 bootstrap resamples.

We provided a line of code that calls your `generate_conf_int` function on the present-day February max temperatures to generate the 99% confidence interval for the average of daily max temperatures in February. The result should be around 67 degrees for the lower bound and around 71 degrees for the upper bound of the interval.

Hint: To implement the resampling procedure, it will be helpful to define a second function (say, `one_resampled_stat`) that simulates one resampled mean, which is then used by the `generate_conf_int` function.

```
one_resampled_stat <- function(sample_data) {
  resample <- sample(sample_data, length(sample_data), replace = TRUE)
  mean(resample)
}

generate_conf_int <- function(month_label, level) {
  sample_observations <- needles_complete |>
    filter(between(Year, 2019, 2021) & Month == month_label) |>
  pull(tmax)
  bootstrapped_means <- replicate(1000,
one_resampled_stat(sample_observations))


  lower_bound <- quantile(bootstrapped_means, (1 - level) / 2)
  upper_bound <- quantile(bootstrapped_means, 1 - (1 - level) / 2)

  return(c(lower_bound, upper_bound))
}


feb_present_ci <- generate_conf_int("02 (Feb)", 0.99)
feb_present_ci
```

```
##     0.5%    99.5%
## 66.67363 70.74246
```

```
. = ottr::check("tests/part3_q1.R")
```

```
## All tests passed!
```

Question 3.2: The `feb_present_ci` 99% confidence interval contains the observed past February average maximum temperature of 68.7 (from the `monthly_increases` tibble). What conclusion can you draw about the effect of climate change on February maximum temperatures in Needles from this information? Use a 1% P-value cutoff.

Since the historical average of 68.7°F lies within the calculated 99% confidence interval of 66.76°F to 70.71°F, we cannot conclude that there has been a significant change in the average February maximum temperature between the past and the present (2019-2021). There is no strong statistical evidence to suggest that climate change has significantly affected the average maximum temperatures in February in Needles, California. ## 🛑 STOP 🛑 : You finished the project!

This is the end of the required component of the project. If you do not plan on completing the extra credit portion, simply submit both the .Rmd notebook file at this point **AND** a knitted PDF document before the final deadline. When submitting to Gradescope, disregard any failing automatic tests that correspond to Parts 4 and 5 of the project; you will not be penalized for this.

The remainder of the project is extra credit ONLY. Do NOT continue unless you have fully completed and are confident in your solutions in Parts 1, 2, and 3.

Part IV: All Months (EXTRA CREDIT)

Let us extend the analysis to see whether the **past average** is contained within the 99% confidence interval of the present average **for each month**. We will repeat the process of calling your `generate_conf_int` function for each month and organize the results into a tibble `all_months_ci`. Run the following cell to perform the experiment. Recall that these "averages" are averages of the max daily temperatures within those time periods.

```
all_months_ci <- monthly_increases |>
  mutate(ci = map(Month, \(x) generate_conf_int(x, 0.99))) |>
  unnest_wider(ci, names_sep = "_")
all_months_ci
```

```
## # A tibble: 12 × 6
##    Month     Past Present Increase `ci_0.5%` `ci_99.5%`
##    <chr>    <dbl>   <dbl>    <dbl>     <dbl>      <dbl>
##  1 01 (Jan)  63.2    66.1    2.88       64.4       67.5
##  2 02 (Feb)  68.7    68.9    0.171      67.0       70.7
##  3 03 (Mar)  76.1    75.5   -0.621      73.8       77.4
##  4 04 (Apr)  85.2    88.9    3.68       86.6       91.4
##  5 05 (May)  93.9    95.7    1.76       93.7       97.9
##  6 06 (Jun) 103.    107.     3.95      106.       109.
##  7 07 (Jul) 108.    112.     3.80      110.       113.
##  8 08 (Aug) 105.    112.     6.41      110.       113.
##  9 09 (Sep)  99.4   103.     4.02      101.       106.
## 10 10 (Oct)  86.1    88.8    2.62       85.8       91.4
## 11 11 (Nov)  72.4    77.3    4.91       74.9       79.7
## 12 12 (Dec)  63.3    64.1    0.800      62.5       65.8
```

Question 4.1: Write `dplyr` code that adds a new Boolean variable to `all_months_ci` named `Contained`, which indicates whether the past average was contained in the interval (`TRUE`) or not (`FALSE`). Assign the resulting tibble to the name `all_months_ci`.

```
all_months_ci <- all_months_ci |>
  mutate(Contained = Past >= `ci_0.5%` & Past <= `ci_99.5%`)

all_months_ci
```

```
## # A tibble: 12 × 7
##    Month     Past Present Increase `ci_0.5%` `ci_99.5%` Contained
##    <chr>    <dbl>   <dbl>    <dbl>     <dbl>      <dbl> <lgl>
##  1 01 (Jan)  63.2    66.1    2.88       64.4       67.5 FALSE
##  2 02 (Feb)  68.7    68.9    0.171      67.0       70.7 TRUE
##  3 03 (Mar)  76.1    75.5   -0.621      73.8       77.4 TRUE
##  4 04 (Apr)  85.2    88.9    3.68       86.6       91.4 FALSE
##  5 05 (May)  93.9    95.7    1.76       93.7       97.9 TRUE
##  6 06 (Jun) 103.    107.     3.95      106.       109.  FALSE
##  7 07 (Jul) 108.    112.     3.80      110.       113.  FALSE
##  8 08 (Aug) 105.    112.     6.41      110.       113.  FALSE
##  9 09 (Sep)  99.4   103.     4.02      101.       106.  FALSE
## 10 10 (Oct)  86.1    88.8    2.62       85.8       91.4 TRUE
## 11 11 (Nov)  72.4    77.3    4.91       74.9       79.7 FALSE
## 12 12 (Dec)  63.3    64.1    0.800      62.5       65.8 TRUE
```

```
. = ottr::check("tests/part4_q1.R")
```

```
## All tests passed!
```

Question 4.2: Summarize your findings. After checking whether the past average (of max temperatures) is contained in the 99% confidence interval for each month, what conclusions can we make about the monthly average maximum temperature in historical (1900-1960) vs. modern (2019-2021) times in the twelve months? Put another way, what null hypothesis should you consider, and for which months would you reject, fail to reject, or accept the null hypothesis? Use a 1% P-value cutoff.

Hint: Do you notice any seasonal patterns?

There are more months where the historical average temp is not contained in the interval indicating that we can reject the null hypothesis and suggests that the temps are significantly different. I noticed that a lot of the colder months didn't show a lot of change compared to some of the warmer months.This could mean climate change has more noticeable effects during the warmer periods. Overall, there is prevalent variation to support warming trends in historical vs modern temperature data.

Part V: Drought (EXTRA CREDIT)

According to the United States Environmental Protection Agency, "Large portions of the Southwest have experienced drought conditions since weekly Drought Monitor records

began in 2000. For extended periods from 2002 to 2005 and from 2012 to 2020, nearly the entire region was abnormally dry or even drier."

Assessing the impact of drought is challenging with just city-level data because so much of the water that people use is transported from elsewhere, but we'll explore the data we have and see what we can learn.

Let's first take a look at the precipitation data in the Southwest region. The tibble `southwest` contains total annual precipitation for 13 cities in the southwestern United States for each year from 1960 to 2021. This dataset is aggregated from the daily data and includes only the Southwest cities from the original dataset that have consistent precipitation records back to 1960.

```
southwest
```

```
## # A tibble: 793 × 3
##    City         Year `Total Precipitation`
##    <chr>       <dbl>                 <dbl>
##  1 Albuquerque  1960                  8.12
##  2 Albuquerque  1961                  8.87
##  3 Albuquerque  1962                  5.39
##  4 Albuquerque  1963                  7.47
##  5 Albuquerque  1964                  7.44
##  6 Albuquerque  1965                  9.31
##  7 Albuquerque  1966                  6.81
##  8 Albuquerque  1967                  8.04
##  9 Albuquerque  1968                 10.7
## 10 Albuquerque  1969                 10.6
## # ℹ 783 more rows
```

Question 5.1: Create a tibble `totals` that has one row for each year in chronological order. It should contain the following variables:

1. `"Year"`: The year (a number)
2. `"Precipitation"`: The total precipitation in all 13 southwestern cities that year

```
totals <- southwest |>
  group_by(Year) |>
  summarize(Precipitation = sum(`Total Precipitation`, na.rm = TRUE)) |>
  arrange(Year)
totals
```

```
## # A tibble: 61 × 2
##     Year Precipitation
##    <dbl>         <dbl>
##  1  1960          150.
##  2  1961          135.
##  3  1962          130.
##  4  1963          132.
##  5  1964          123.
##  6  1965          188.
```

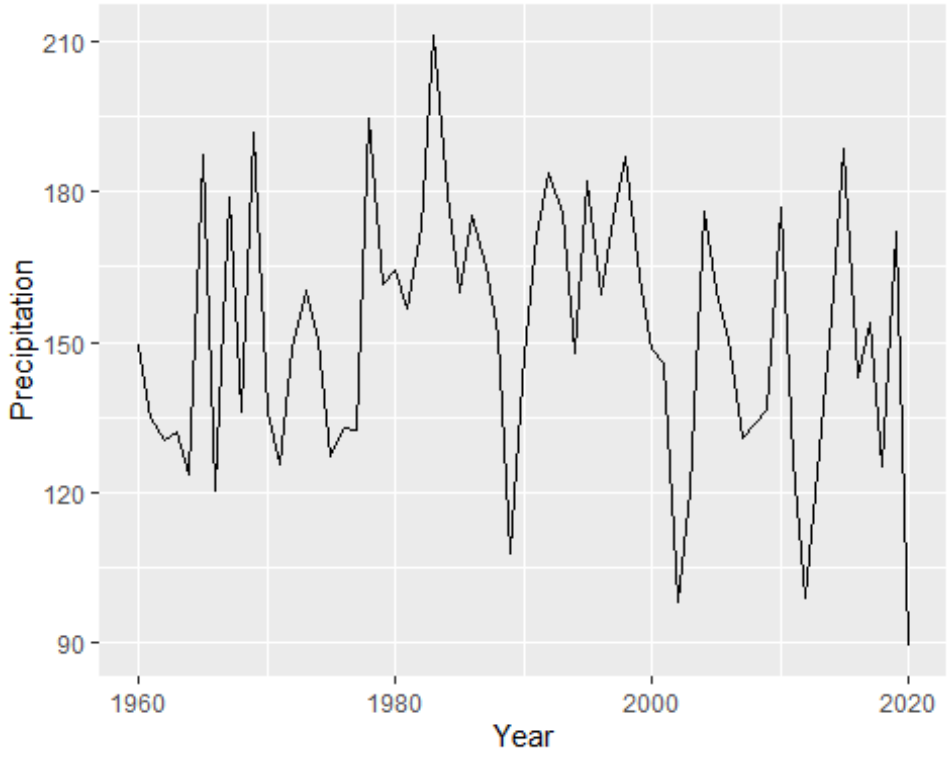
```
##  7  1966          120.
##  8  1967          179.
##  9  1968          136.
## 10  1969          192.
## # ℹ 51 more rows
```

```
. = ottr::check("tests/part5_q1.R")
```

```
## All tests passed!
```

Run the cell below to plot the total precipitation in these cities over time, so that we can try to spot the drought visually. As a reminder, the drought years given by the EPA were (2002-2005) and (2012-2020).

```
totals |>
  ggplot() +
  geom_line(aes(x = Year, y = Precipitation))
```



This plot isn't very revealing. Each year has a different amount of precipitation, and there is quite a bit of variability across years, as if each year's precipitation is a random draw from a distribution of possible outcomes.

Could it be that these so-called "drought conditions" from 2002-2005 and 2012-2020 can be explained by chance? In other words, could it be that the annual precipitation amounts in the Southwest for these drought years are like **random draws from the same underlying distribution** as for other years? Perhaps nothing about the Earth's

precipitation patterns has really changed, and the Southwest U.S. just happened to experience a few dry years close together.

To assess this idea, let's conduct a permutation test in which **each year's total precipitation** is an outcome, and the condition is **whether or not the year is in the EPA's drought period**.

The following function `year_to_drought` distinguishes between drought years as described in the U.S. EPA statement above (2002-2005 and 2012-2020) and other years. Note that the label "other" is perhaps misleading, since there were other droughts before 2000, such as the massive 1988 drought that affected much of the U.S. However, if we're interested in whether these modern drought periods (2002-2005 and 2012-2020) are *normal* or *abnormal*, it makes sense to distinguish the years in this way.

```
year_to_drought <- function(year) {
  if(between(year, 2002, 2005) | between(year, 2012, 2020)) {
    return('drought')
  } else {
    return('other')
  }
}
```

Question 5.2: Define null and alternative hypotheses for a permutation test that investigates whether drought years are **drier** (i.e., **have less precipitation**) than other years.

Null Hypothesis: There is no difference in the total precipitation between drought years provided and other years. The drought years are not significantly drier than other years.

Alternative Hypothesis: The drought years provided have less total precipitation than other years. The precipitation in the drought years is significantly lower compared to non-drought years.

Question 5.3: Form a tibble named `drought`. It should contain one row per year and the following two variables:

- `"Label"`: Denotes if a year is part of a `"drought"` year or an `"other"` year
- `"Precipitation"`: The sum of the total precipitation in 13 Southwest cities that year

```
drought <- totals |>
  mutate(Label = sapply(Year, year_to_drought)) |>
  select(Label, Precipitation)


drought
```

```
## # A tibble: 61 × 2
##    Label Precipitation
##    <chr>         <dbl>
##  1 other          150.
##  2 other          135.
```

```
##  3 other          130.
##  4 other          132.
##  5 other          123.
##  6 other          188.
##  7 other          120.
##  8 other          179.
##  9 other          136.
## 10 other          192.
## # ℹ 51 more rows
```

```
. = ottr::check("tests/part5_q3.R")
```

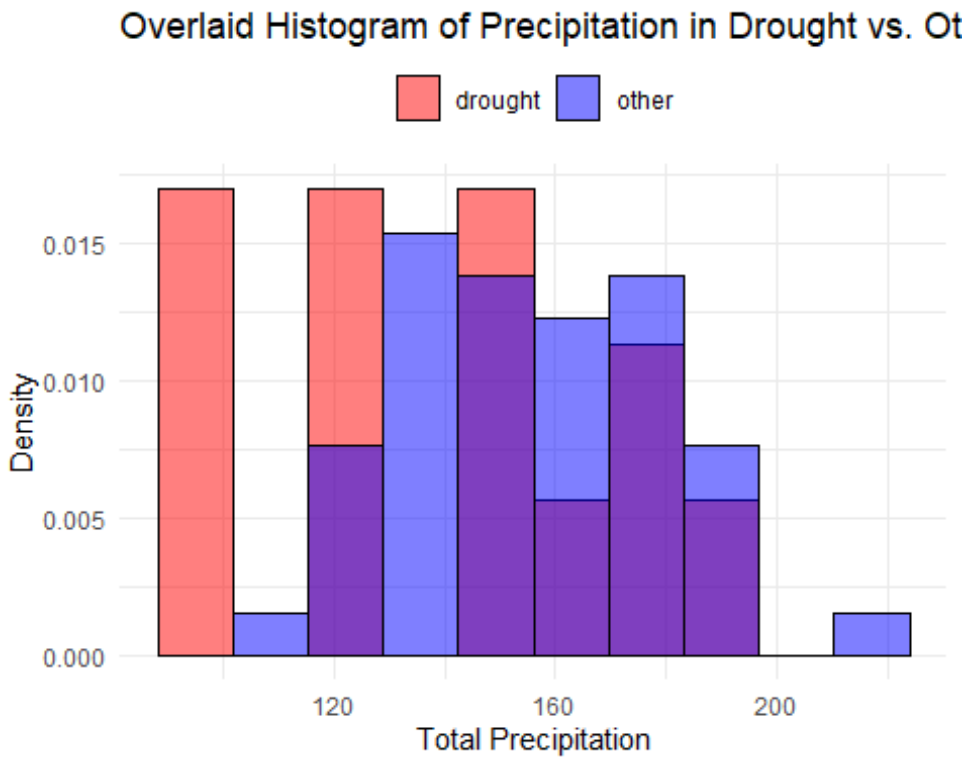
```
## All tests passed!
```

Question 5.4: Using the tibble `drought`, construct an overlaid histogram of two observed distributions: the total precipitation in drought years and the total precipitation in other years. We have provided bins to use (in `bins`) when creating your histogram.

```
bins <- seq(85, 215, 13)

ggplot(drought, aes(x = Precipitation, fill = Label)) +
  geom_histogram(data = subset(drought, Label == 'drought'), aes(y =
..density..),
                 bins = length(bins) - 1, alpha = 0.5, color = 'black') +
  geom_histogram(data = subset(drought, Label == 'other'), aes(y =
..density..),
                 bins = length(bins) - 1, alpha = 0.5, color = 'black') +
  scale_fill_manual(values = c('drought' = 'red', 'other' = 'blue')) +
  labs(title = "Overlaid Histogram of Precipitation in Drought vs. Other
Years",
       x = "Total Precipitation",
       y = "Density") +
  theme_minimal() +
  theme(legend.title = element_blank(), legend.position = "top")
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2
3.4.0.
## ℹ Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Before you continue, inspect the histogram you just created and try to guess the conclusion of the permutation test. Building intuition about the result of hypothesis testing from visualizations is quite useful for data science applications.

While we are at it, let us also check the `drought` tibble. It should have two variables `Label` and `Precipitation` with 61 rows, 13 of which are for `"drought"` years.

```
drought |>
  count(Label)
```

```
## # A tibble: 2 × 2
##   Label       n
##   <chr>   <int>
## 1 drought    13
## 2 other      48
```

Question 5.5. Our next step is to choose a test statistic based on the null and alternative hypotheses defined in **Question 5.2**. Define a good test statistic by writing a function `test_statistic` that implements the test statistic you have chosen. This function should receive a two-column tibble that is of the same form as `drought`.

Important requirements for your test statistic: To develop your test statistic, think about what kinds of values are evidence in favor of the alternative hypothesis and what values are insufficient evidence. You should look back to the hypotheses you defined in **Question 5.2**. What would a large positive value represent? A small negative value? A value close to 0?

```
test_statistic <- function(data) {

  drought_mean <- mean(data$Precipitation[data$Label == "drought"], na.rm =
TRUE)


  other_mean <- mean(data$Precipitation[data$Label == "other"], na.rm = TRUE)


  return(other_mean - drought_mean)
}
```

In the following cell, we used the function you defined to assign `observed_statistic` to the observed value of the test statistic.

```
observed_statistic <- test_statistic(drought)
observed_statistic
```

```
## [1] 15.85671
```

Question 5.6 Write a function to simulate the test statistic under the null hypothesis. The `simulate_precipitation_null` function should simulate the null hypothesis once (not 1,000 times) and return the value of the test statistic for that simulated sample.

```
simulate_precipitation_null <- function() {

  shuffled_drought <- drought |>
    mutate(Label = sample(Label))


  return(test_statistic(shuffled_drought))
}

# Run your function a couple times to make sure that it works
simulate_precipitation_null()
```

```
## [1] 0.7435737
```

```
. = ottr::check("tests/part5_q6.R")
```

```
## All tests passed!
```

Question 5.7 Write R code to simulate 1,000 values of the test statistic under the null hypothesis. Store the result in a vector named `sampled_stats`.

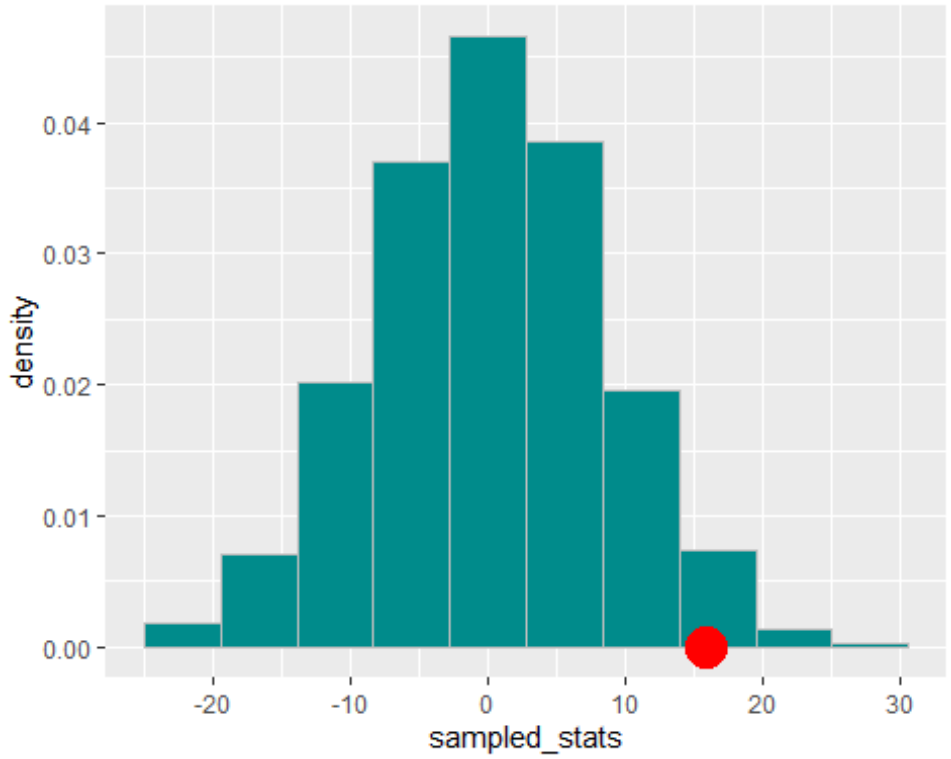
```
sampled_stats <- replicate(1000, simulate_precipitation_null())

. = ottr::check("tests/part5_q7.R")
```

```
## All tests passed!
```

Here is a histogram of the simulation results, annotated with the observed value of the test statistic.

```
ggplot(tibble(sampled_stats)) +
    geom_histogram(aes(x = sampled_stats, y = after_stat(density)),
                   color = "gray", fill = "darkcyan", bins = 10) +
    geom_point(aes(x = observed_statistic, y = 0), color = "red", size = 7)
```



Question 5.8 State a conclusion from this test. You should reference some conventional P-value cutoff, such as 1% or 5%. What have you learned about the EPA's statement on drought?

The test statistic function created calculates the the mean precipitation between the given set of drought years and other years. The large postive value from the simulated sample indicates that we could reject the null hypothesis. This is also shown in the histogram of the simulation results. The observed statistic is within the extreme 5% cutoff tail of the distribution suggesting that the provided droughts years (2002-2005 and 2012-2020) are significantly drier than other years. Overall, this would support what the EPA said about these years being unusually and exceptionally drier than other years.

Question 5.9 Does your conclusion from **Question 5.8** apply to the entire Southwest region of the U.S.? Why or why not? Feel free to look into geographical features of this region!

My conclusion from 5.8 may not necessarily apply to all areas of the Southwest region because it isn't just made up of dry desert habitats. The coastal areas of this region usually

have more precipitation and there are geographical features like mountains and plains that can affect the overall climate of certain areas in the region.

This is the end of the extra credit portion of the project – well done!!

Make sure that all automatic tests are passing and that you are fully confident in the answers you have given. As with all your assignments, submit both the final .Rmd notebook file *AND* a generated PDF document before the final deadline noted at the top of this notebook and on Gradescope.