MECH476: Engineering Data Analysis in R

Chapter 3 Homework: Fort Collins Ozone

Michael Thill

11 December, 2025

Note: Homework can be submitted as *either* .html or .pdf documents. If you haven't installed *LaTex*, change the output mode in the above YAML to `html_document` for ease of knitting and homework submission.

This R Markdown (.Rmd) file is a template for your Chapter 3 Homework. Do everything within this file. Make it your own, but be careful not to change the code-figure-text integration I set up with the code appendix and the global options. If you have used R Markdown before and are comfortable with the extra options, feel free to customize to your heart's desire. In the end, we will grade the **knitted** PDF or HTML document from within your private GitHub repository. Remember to make regular, small commits (e.g., at least one commit per question) to save your work. We will grade the latest knit, as long as it occurs *before* the start of the class in which we advance to the next chapter. As always, reach out with questions via GitHub Issues or during office hours.

Ozone Data

The corresponding data file (.csv) for Homework 3 contains *hourly* ozone data from two sites in Fort Collins.

Background

Incidentally, the ozone standard is set to 0.07 parts per million (ppm). Outdoor ozone levels are measured every hour, but the Environmental Protection Agency states that the limit should be judged against an eight-hour rolling average (a transformation that is possible in R, but outside the purview of this chapter). Fort Collins, and most of the Front Range, is in non-attainment for this standard, which is why you are required to get the emissions checked on your car every year.

Question 0: Load R Packages

Question 1: Preparation

Import, Select, and Clean Data

Using the pipe (`%>%`) to connect **three** lines of code, *import* the file with the appropriate `readr` function and a relative pathname, *select* the below variables, and *drop* missing observations. Remember to assign the output to a `tibble` object with an informative name on the left side of the `gets arrow`.

Retain the following variables in Step 2 of the "pipe":

- `sample_measurement` (ozone measurement in ppm)
- `datetime` (date in YYYY-MM-DD format and time of measurement in HH:MM:SS)

`sample_measurement` is a vague variable name; what is being measured? It is also a little long. Add a fourth line of code to your pipe that renames this variable as `ozone_ppm`, indicating an ozone concentration measurement in parts per million (ppm). *FYI*: If the dataset had multiple ozone measurements on different time scales, it would be important to include that information in the variable name (e.g., `ozone_ppm_hourly`).

```
## # A tibble: 6 x 2
##   ozone_ppm datetime
##       <dbl> <dttm>
## 1     0.017 2019-01-01 07:00:00
## 2     0.017 2019-01-01 08:00:00
## 3     0.017 2019-01-01 09:00:00
## 4     0.017 2019-01-01 10:00:00
## 5     0.015 2019-01-01 11:00:00
## 6     0.017 2019-01-01 12:00:00
```

Examine Data

Examine the structure and contents of the dataframe to confirm the file imported and was manipulated properly. How many observations were dropped due to missing values? *Hint*: Only consider the `ozone_ppm` variable; there were no missing values for `datetime`.

```
## [1] 606
```

Question 2: Extract and Compare

Using a variant of the `dplyr::slice()` function with **two** arguments (one to specify number of observations to extract and one to specify by which variable R should the observations in the output), extract the top ten ozone values and assign them to a separate object.

```
## # A tibble: 10 x 2
##    ozone_ppm datetime
##        <dbl> <dttm>
##  1     0.096 2019-07-03 20:00:00
##  2     0.093 2019-07-03 21:00:00
##  3     0.09  2019-07-03 20:00:00
##  4     0.089 2019-07-03 19:00:00
##  5     0.086 2019-07-24 20:00:00
##  6     0.083 2019-06-29 20:00:00
##  7     0.082 2019-09-10 23:00:00
##  8     0.081 2019-07-03 19:00:00
##  9     0.081 2019-06-07 19:00:00
## 10     0.081 2019-08-03 19:00:00
```

Now, complete the same process for the bottom ten ozone values.

```
## # A tibble: 10 x 2
##    ozone_ppm datetime
```

```
##        <dbl> <dttm>
##  1         0 2019-01-05 07:00:00
##  2         0 2019-02-02 07:00:00
##  3         0 2019-02-04 07:00:00
##  4         0 2019-02-28 11:00:00
##  5         0 2019-03-01 07:00:00
##  6         0 2019-03-08 07:00:00
##  7         0 2019-05-11 11:00:00
##  8         0 2019-11-24 10:00:00
##  9         0 2019-12-20 00:00:00
## 10         0 2019-12-20 01:00:00
```

Do the highest and lowest values tend to occur at certain times of the day?

Question 3: Maximum and Minimum

Using the output from the previous question, on what day does the highest value occur? The lowest?

```
## # A tibble: 1 x 1
##   datetime
##   <dttm>
## 1 2019-07-03 20:00:00
```

```
## # A tibble: 10 x 1
##    datetime
##    <dttm>
##  1 2019-01-05 07:00:00
##  2 2019-02-02 07:00:00
##  3 2019-02-04 07:00:00
##  4 2019-02-28 11:00:00
##  5 2019-03-01 07:00:00
##  6 2019-03-08 07:00:00
##  7 2019-05-11 11:00:00
##  8 2019-11-24 10:00:00
##  9 2019-12-20 00:00:00
## 10 2019-12-20 01:00:00
```

Question 4: Mutate

Create a new variable (`ozone_ugm3`) that provides ozone concentration in micrograms per cubic meter (ug/m3) instead of parts per million (ppm), as in `ozone_ppm`. Because we do not have access to crucial measurements such as atmospheric pressure or temperature, the following exercise will not give the true ozone concentration values (ug/m3) but will give you practice using the appropriate `dplyr` verb.

Although the real data are not available in this dataset, you will use the following information to complete the conversion. The hard-coded values are illustrative estimates, not accurate readings.

- ozone concentration in parts per million (`ozone_ppm`)
- molecular weight of ozone (47.998 g/mol)
- Celsius to Kelvin temperature conversion (K = 273.15 + C) (will be used twice but in different ways in the numerator and denominator)

- estimated atmospheric pressure in Fort Collins, CO (637 mmHg)
- universal gas constant (22.4136)
- estimated temperature in Fort Collins, CO (10° Celsius)

Then, you will need to embed the following ideal gas law equation into a function call that creates a new variable based on the values from `ozone_ppm`. *Hint*: Based on this approach, an ozone concentration of 0.001 ppm converts to approximately 2.066 ug/m3.

$$\texttt{ozoneugm3} = 1000 * \frac{\Big((\texttt{ozoneppm})(\text{molecular weight of ozone})(\text{Celsius to Kelvin value})(\text{atmospheric pressure})\Big)}{\Big((\text{universal gas constant})(\text{Celsius to Kelvin value} + \text{temperature})(\text{atmospheric pressure})\Big)}$$

```
## # A tibble: 6 x 2
##   ozone_ugm3 datetime
##        <dbl> <dttm>
## 1       35.1 2019-01-01 07:00:00
## 2       35.1 2019-01-01 08:00:00
## 3       35.1 2019-01-01 09:00:00
## 4       35.1 2019-01-01 10:00:00
## 5       31.0 2019-01-01 11:00:00
## 6       35.1 2019-01-01 12:00:00
```

Appendix

```r
# set global options for figures, code, warnings, and messages
knitr::opts_chunk$set(fig.width = 6, fig.height = 4, fig.path = "../figs/",
                      echo = FALSE, warning = FALSE, message = FALSE)
# do you need to install each R package?
# load packages for current R session
library("tidyverse")
# ozone: import, select, drop missing observations, rename
 # use relative pathname
 # select needed variables
 # drop missing observations
 # rename main variable

raw_data <- read_csv("../Data/ftc_o3.csv")

ozone_data <- read_csv("../Data/ftc_o3.csv") %>%
  select(sample_measurement, datetime) %>%
  drop_na() %>%
  rename(ozone_ppm = sample_measurement)

head(ozone_data)
# examine tibble
# calculate number of missing observations

sum(is.na(raw_data$sample_measurement))

# extract top ten ozone values and save them to df

top10_values <- slice_max(ozone_data,
                          order_by = ozone_ppm,
                          n = 10)

top10_values

# extract bottom ten ozone values and save them to df

bottom10_values <- slice_min(ozone_data,
                          order_by = ozone_ppm,
                          n = 10)

bottom10_values

# extract date/time of highest ozone concentration

max_value_date <- slice_max(top10_values,
                        order_by = ozone_ppm,
                        n = 1) %>%
  select(datetime)

max_value_date

# extract date/time of lowest ozone concentration
```

```r
min_value_date <- slice_min(bottom10_values,
                      order_by = ozone_ppm,
                      n = 1) %>%
  select(datetime)

min_value_date

# create new variable of ug/m3 from ppm and overwrite dataset

convert_ozone <- function(data) {
  MW <- 47.998        # molecular weight of ozone (g/mol)
  CK <- 273.15        # Celsius to Kelvin
  P <- 637            # pressure (mmHg)
  R  <- 22.4136       # universal gas constant
  T  <- 10            # temperature in Celsius


  data %>%
    mutate(ozone_ppm = 1000 * ((ozone_ppm * MW * CK * P) / (R * (CK + T) * P))) %>%
    rename(ozone_ugm3 = ozone_ppm)
}

ozone_converted <- convert_ozone(ozone_data)

head(ozone_converted)
```