Stats_506_PS5

Haoyu, Zhao

The Github link to this problem set is https://github.com/hzhaoar/Stats_506_PS5

Problem 1

```
rm(list = ls())
```

```
library(ggplot2)
```

```
# Read our data
nnmaps <- read.csv("./chicago-nmmaps.csv")
```

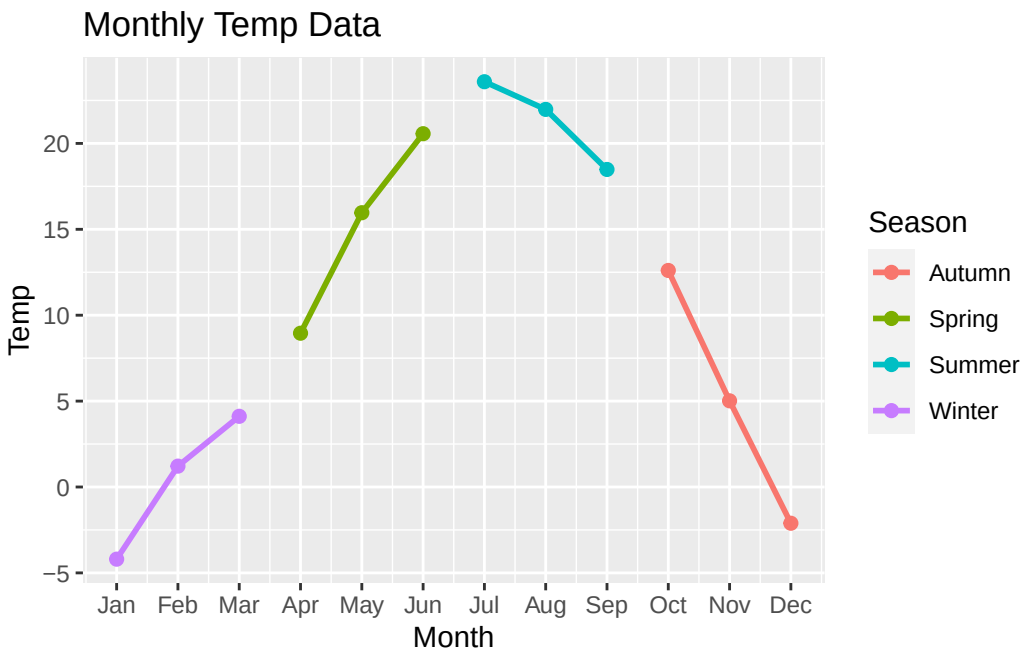
Task (a)

```
# Convert the temperature to celsius
nnmaps$temp_celsius <- (nnmaps$temp - 32) * 5/9

# Calculate the mean for each month across different years
suppressWarnings(monthly_avg <-
                   aggregate(nnmaps, by = list(nnmaps$month_numeric),
                             FUN = mean, na.rm = TRUE))

# Rename "season" based on values in "month_numeric"
seasons <- c("Winter", "Spring", "Summer", "Autumn")
for (i in (1:4)){
  monthly_avg[which(monthly_avg$month_numeric %in%
                      c(3*i-1,3*i-2,3*i)), "season"] <- seasons[i]
}
```

```r
# Draw the plot
ggplot(monthly_avg, aes(x = month_numeric, y = temp_celsius, color = season)) +
  geom_point(size = 2) +
  geom_line(aes(group = season), linewidth = 1) +
  labs(title = "Monthly Temp Data", x = "Month", y = "Temp") +
  scale_x_continuous(breaks = 1:12, labels = month.abb) +
  guides(color = guide_legend(title = "Season"))
```



Task (b)

```r
# Add temp_celsius to the plot
p <- ggplot(monthly_avg) +
  geom_point(aes(x = month_numeric, y = temp_celsius, color = "Temp"), size = 2) +
  geom_line(aes(x = month_numeric, y = temp_celsius, color = "Temp"),
            linewidth = 1) +
  labs(title = "Monthly Data", x = "Month", y = "Values") +
  scale_x_continuous(breaks = 1:12, labels = month.abb)

# Add o3 to the plot
p <- p +
```

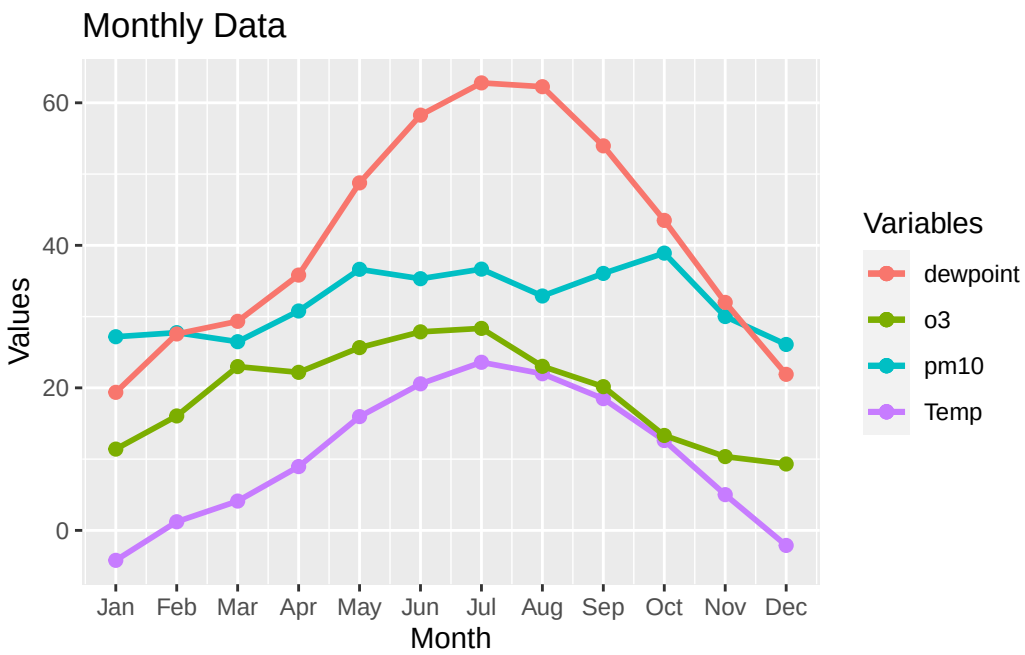
```r
  geom_point(aes(x = month_numeric, y = o3, color = "o3"), size = 2) +
  geom_line(aes(x = month_numeric, y = o3, color = "o3"),
            linewidth = 1)

# Add pm10 to the plot
p <- p +
  geom_point(aes(x = month_numeric, y = pm10, color = "pm10"), size = 2) +
  geom_line(aes(x = month_numeric, y = pm10, color = "pm10"),
            linewidth = 1)

# Add dewpoint to the plot
p <- p +
  geom_point(aes(x = month_numeric, y = dewpoint, color = "dewpoint"), size = 2) +
  geom_line(aes(x = month_numeric, y = dewpoint, color = "dewpoint"),
            linewidth = 1)

# Rename the title of legend
p <- p +
  guides(color = guide_legend(title = "Variables"))

# Show the plot
p
```



From the plot, we can observe that `pm10` is the variable with least seasonal trend.

```
rm(list = ls())
```

Problem 2

Task (a)

In my `poly` class, polynomial expressions are represented by two vectors: `coefficients` and `powers`. For example, the polynomial $7x^3 - 3x + 2$ is represented by coefficients = (7, -3, 2) and powers = (3, 1, 0)

```
# Declare the poly class
setClass("poly",
         slots = c(coefficients = "numeric",
                   powers = "numeric"))
```

```
# Constructor function
make_poly <- function(coefficients, powers) {

  # Use tapply to sum coefficients based on powers
  result <- tapply(coefficients, powers, sum)

  # Extract the powers and coefficients from the result
  unique_powers <- as.numeric(names(result))
  unique_coeff <- as.numeric(result)

  # Remove the entry with coefficient 0
  non_zero_coeff <- which(unique_coeff != 0)
  unique_powers <- unique_powers[non_zero_coeff]
  unique_coeff <- unique_coeff[non_zero_coeff]

  return(new("poly", coefficients = unique_coeff,
             powers = unique_powers))
}
```

```
# Validator function
setValidity("poly", function(object){
  if (! all(sapply(object@powers, function(n){
    return((n == floor(n)) && (n >= 0))
```

```
  }))) {
    stop("Powers should be a non-negative integer-valued vector")
  }

  if (!(length(object@coefficients) == length(object@powers))) {
    stop(cat("Coefficients and powers should have the same length"))
  }
  return(TRUE)
})
```

```
Class "poly" [in ".GlobalEnv"]

Slots:

Name:  coefficients       powers
Class:      numeric      numeric
```

```
# Show method
setMethod("show", "poly",
          function(object) {
            all_powers <- object@powers
            all_coefficients <- object@coefficients

            if (all(all_coefficients == 0)){
              return(cat("Polynomial: ", "0", "\n"))
            }

            # Order the polynomial entries based on powers
            order_indices <- order(all_powers, decreasing = TRUE)
            all_powers <- all_powers[order_indices]
            all_coefficients <- all_coefficients[order_indices]

            # Generate a string to display the polynomial
            terms <- paste(paste(all_coefficients, "x^", all_powers,
                                 sep = "", collapse = " + "),
                           collapse = " ")

            terms <- paste(terms, " ")

            # Replace x^1 with x
            terms <- gsub("x\\^1\\s", "x ", terms)
```

```
            # Remove x^0 from the string
            terms <- gsub("x\\^0", "", terms)
            # Replace -1x with -x
            terms <- gsub("\\-1x", "-x", terms)
            # Replace 1x with x
            terms <- gsub("[^.][1]x", " x", terms)
            # Special case for leading 1
            terms <- gsub("^1x", "x", terms)
            # Replace + - with -
            terms <- gsub("\\+\\s-", "- ", terms)

            # Print the string
            return(cat("Polynomial: ", terms, "\n"))
          })
```

```
# Addition method
setMethod("+", signature(e1 = "poly",
                         e2 = "poly"),
          function(e1, e2) {
            df_e1 <- data.frame(power = e1@powers, coeff = e1@coefficients)
            df_e2 <- data.frame(power = e2@powers, coeff = e2@coefficients)

            # Merge the coefficients and powers
            merged_df <- merge(df_e1, df_e2, by = "power", all = TRUE)
            merged_df[is.na(merged_df)] <- 0

            # Calculate the sum of coefficients for each power
            all_powers <- merged_df$power
            result_coeff <- rowSums(merged_df[, c("coeff.x", "coeff.y")],
                                    na.rm = TRUE)

            # Construct the new polynomial object
            return(make_poly(coefficients = result_coeff,
                             powers = all_powers))
          })
```

```
# Subtraction method
setMethod("-", signature(e1 = "poly",
                         e2 = "poly"),
          function(e1, e2) {
            df_e1 <- data.frame(power = e1@powers, coeff = e1@coefficients)
```

```
    df_e2 <- data.frame(power = e2@powers, coeff = e2@coefficients)

    # Merge the coefficients and powers
    merged_df <- merge(df_e1, df_e2, by = "power", all = TRUE)
    merged_df[is.na(merged_df)] <- 0

    # Calculate the difference between coefficients for each power
    merged_df[, "coeff.y"] <- (-1)*merged_df[, "coeff.y"]
    all_powers <- merged_df$power
    result_coeff <- rowSums(merged_df[, c("coeff.x", "coeff.y")],
                            na.rm = TRUE)

    # Construct the new polynomial object
    return(make_poly(coefficients = result_coeff,
                     powers = all_powers))
}
```

Here are some basic tests for my `make_poly` function.

```
make_poly(coefficients = c(1, 2), powers = c(0, 2))
```

```
Polynomial:  2x^2 + 1
```

```
make_poly(coefficients = c(3.1, -2.1, 1.1), powers = c(17, 1, 0))
```

```
Polynomial:  3.1x^17 - 2.1x + 1.1
```

```
make_poly(coefficients = c(4, 2, 0), powers = c(2, 1, 3))
```

```
Polynomial:  4x^2 + 2x
```

```
make_poly(coefficients = c(0, 2), powers = c(2, 0))
```

```
Polynomial:  2
```

```
make_poly(coefficients = c(0, 0), powers = c(3, 7))
```

```
Polynomial:  0
```

Task (b)

```
# Example usage of the constructor function
p1 <- make_poly(coefficients = c(3, 2), powers = c(2, 0))
p2 <- make_poly(coefficients = c(7, -2, -1, 17), powers = c(3, 2, 1, 0))
```

```
# Test show method
p1
```

```
Polynomial:  3x^2 + 2
```

```
p2
```

```
Polynomial:  7x^3 - 2x^2 - x + 17
```

```
# Test plus sign and minus sign
p1 + p2
```

```
Polynomial:  7x^3 + x^2 - x + 19
```

```
p1 - p2
```

```
Polynomial:  -7x^3 + 5x^2 + x - 15
```

Problem 3

```
suppressWarnings(library(data.table))
suppressWarnings(library(nycflights13))
```

```
data("flights")
data("airports")
data("planes")

flights <- data.table(flights)
airports <- data.table(airports)
```

```
planes <- data.table(planes)
```

Task (a)

```
# Departure delay
departure_table <- flights[, .(mean_delay = mean(dep_delay, na.rm = TRUE),
                               median_delay = median(dep_delay, na.rm = TRUE),
                               numflights = .N), by = origin][numflights > 10]

departure_table <- merge(departure_table, airports,
                         by.x = "origin", by.y = "faa", all.x = TRUE)
departure_table <- departure_table[order(-mean_delay)]


departure_table[, .(Airport = name,
                    Mean_Delay = mean_delay,
                    Median_Delay = median_delay)]
```

```
              Airport Mean_Delay Median_Delay
1: Newark Liberty Intl   15.10795           -1
2: John F Kennedy Intl   12.11216           -1
3:          La Guardia   10.34688           -3
```

```
# Arrival delay
arrival_table <- flights[, .(mean_delay = mean(arr_delay, na.rm = TRUE),
                             median_delay = median(arr_delay, na.rm = TRUE),
                             numflights = .N), by = dest][numflights > 10]

arrival_table <- merge(arrival_table, airports,
                       by.x = "dest", by.y = "faa", all.x = TRUE)
arrival_table <- arrival_table[order(-mean_delay)]

# Replace the name with its faa if name is NA
arrival_table[is.na(arrival_table$name), "name"] <-
  arrival_table[is.na(arrival_table$name), "dest"]


arrival_table[, .(Airport = name,
                      Mean_Delay = mean_delay,
```

```
                        Median_Delay = median_delay)]
```

```
                    Airport Mean_Delay Median_Delay
  1:     Columbia Metropolitan  41.764151         28.0
  2:                Tulsa Intl  33.659864         14.0
  3:         Will Rogers World  30.619048         16.0
  4:      Jackson Hole Airport  28.095238         15.0
  5:             Mc Ghee Tyson  24.069204          2.0
 ---
 97:       Seattle Tacoma Intl  -1.099099        -11.0
 98:             Honolulu Intl  -1.365193         -7.0
 99:                       STT  -3.835907         -9.0
100: John Wayne Arpt Orange Co  -7.868227        -11.0
101:         Palm Springs Intl -12.722222        -13.5
```

Task (b)

```
fastest_model <- merge(flights, planes,
                       by.x = "tailnum", by.y = "tailnum", all.x = TRUE)
fastest_model[, mph := distance / (air_time / 60)]
fastest_model <- fastest_model[, .(avgmph = mean(mph, na.rm = TRUE),
                                   nflights = .N), by = model][order(-avgmph)][1]

fastest_model
```

```
    model   avgmph nflights
1: 777-222 482.6254        4
```