STAT 184 — Activity 14

Noah Yasbin

2025-11-17

Table of Contents

0.1 Question 3 1
0.2 Question 4 5
0.3 Question 5 7

0.1 Question 3

In this section, I revisited my data wrangling from Activities #08 and #10. The goal was to fix earlier issues, ensure that the code runs without modification on any computer, and produce an individual-level dataset where each row represents a single soldier. For this assignment, I focused on Army Warrant Officers as my subgroup for generating a two-way frequency table of sex by pay grade.

```r
# Load packages
library(dplyr)
```

```

Attaching package: 'dplyr'


The following objects are masked from 'package:stats':

    filter, lag


The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
library(tidyr)
library(readr)

# Load data
raw <- read.csv("USArms.csv")

# 1. Extract header rows and construct proper column names
h1 <- as.character(raw[1, ])  # Branch names
h2 <- as.character(raw[2, ])  # Sex labels

# Forward-fill branch names across the header row
for (i in 2:length(h1)) {
  val <- h1[i]
  if (is.na(val) || nchar(trimws(val)) == 0) {
    h1[i] <- h1[i - 1]
  }
}

# Combine Branch + Sex into proper names
new_names <- h1
new_names[1] <- "PayGrade"
for (i in 2:length(new_names)) {
  new_names[i] <- paste0(h1[i], "_", h2[i])
}
names(raw) <- new_names

# 2. Remove header rows & parse numeric columns
data <- raw[-c(1, 2), ]
data$PayGrade <- trimws(as.character(data$PayGrade))
data <- data %>% mutate(
  across(
    -PayGrade,
    ~ suppressWarnings(parse_number(as.character(.)))
  )
)


# 3. Reshape to group-level long format
soldier_groups <- data %>%
  pivot_longer(
    cols = -PayGrade,
    names_to = c("Branch", "Sex"),
    names_sep = "_",
    values_to = "Count"
  ) %>%
  mutate(Branch = trimws(Branch),
         Sex = trimws(Sex)) %>%
  filter(!is.na(Count), Branch != "Total")

# 4. Expand to individual-level table
soldiers_individual <- soldier_groups %>%
  filter(Count > 0) %>%
  uncount(weights = Count, .remove = TRUE)
```

```
table_enlisted <- soldiers_individual %>%
  filter(
    Branch == "Army",
    grepl("^E", PayGrade)
  ) %>%
  count(Sex, PayGrade) %>%
  tidyr::pivot_wider(names_from = Sex, values_from = n, values_fill = 0)

table_enlisted
```

```
# A tibble: 9 x 4
  PayGrade Female  Male Total
  <chr>     <int> <int> <int>
1 E1         1326  7429  8755
2 E2         4336 22338 26674
3 E3        10229 43775 54004
4 E4        15143 79234 94377
5 E5        10954 54803 65757
6 E6         7363 49502 56865
7 E7         4410 30264 34674
8 E8         1472  9482 10954
9 E9          394  2865  3259
```

```
table_warrant <- soldiers_individual %>%
  filter(
    Branch == "Army",      #Pick Branch choice
    grepl("^W", PayGrade)  # W1-W5
  ) %>%
  count(Sex, PayGrade) %>%
  tidyr::pivot_wider(names_from = Sex, values_from = n, values_fill = 0)

table_warrant
```

```
# A tibble: 5 x 4
  PayGrade Female  Male Total
  <chr>     <int> <int> <int>
1 W1          460  3727  4187
2 W2          692  6024  6716
3 W3          346  2794  3140
4 W4          137  1378  1515
5 W5           43   494   537
```

```
soldiers_individual <- soldiers_individual %>%
  mutate(
    RankGroup = case_when(
      grepl("^E", PayGrade) ~ "Enlisted",
      grepl("^W", PayGrade) ~ "Warrant",
      grepl("^O", PayGrade) ~ "Officer",
      TRUE ~ NA_character_
    )
  )

#Select Branch and Rank Group for Analysis
chosen_branch <- "Army"
chosen_group  <- "Warrant"

subset_group <- soldiers_individual %>%
  filter(
    Branch == chosen_branch,
    RankGroup == chosen_group
  )

#Create the Two-Way Frequency Table (PayGrade × Sex)
freq_table <- subset_group %>%
  count(PayGrade, Sex) %>%
  tidyr::pivot_wider(
    names_from  = Sex,
    values_from = n,
    values_fill = 0
  ) %>%
  arrange(PayGrade)
```

```r
#Display the Final Frequency Table
freq_table
```

```
# A tibble: 5 x 4
  PayGrade Female  Male Total
  <chr>     <int> <int> <int>
1 W1          460  3727  4187
2 W2          692  6024  6716
3 W3          346  2794  3140
4 W4          137  1378  1515
5 W5           43   494   537
```

The frequency table for Army Warrant Officers reveals a strong imbalance between male and female soldiers across all W level pay grades. Each rank from W1 through W5 contains significantly more men than women, and in some pay grades female representation is extremely limited or absent. If sex and rank were independent, we would expect the proportion of men and women to be more similar across the pay grades; however, the table shows consistently higher male counts at every level. This pattern indicates that sex and rank are not independent in this subgroup, and the table effectively highlights the uneven gender distribution among Army Warrant Officers.

0.2 Question 4

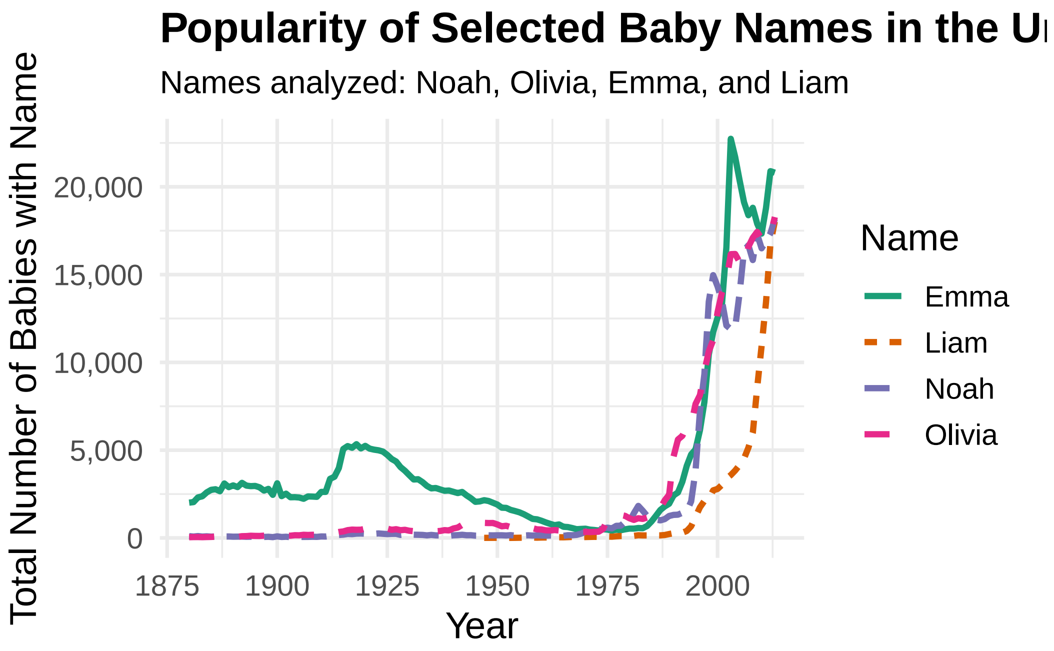
This section incorporates the visualization from Activity #13, where I explored long-term trends in the popularity of four chosen baby names: Noah, Olivia, Emma, and Liam. These names were selected because they have been consistently popular nationwide over the last two decades and show distinct historical patterns that make them interesting for comparison.

```r
#Load Packages
library(dplyr)
library(ggplot2)
library(dcData)

#Choose Names
my_names <- c("Noah", "Olivia", "Emma", "Liam")

#Filter and Summarize Dataset
#Keep only the selected names and calculate total babies per name per year
subset_names <- BabyNames %>%
  filter(name %in% my_names) %>%
  group_by(name, year) %>%
  summarize(
    total = sum(count),
    .groups = "drop"
  )
```

```r
ggplot(
  subset_names,
  aes(
    x = year,
    y = total,
    color = name,
    linetype = name,
    alt = "A line plot showing the popularity of the baby names Noah, Olivia,
    Emma, and Liam over time, with each name represented by both a unique color
    and line type to ensure accessibility for color-blind readers."
  )
) +
  geom_line(linewidth = 1.1) +
  labs(
    title = "Popularity of Selected Baby Names in the United States Over Time",
    subtitle = "Names analyzed: Noah, Olivia, Emma, and Liam",
    x = "Year",
    y = "Total Number of Babies with Name",
    color = "Name",
    linetype = "Name"
  ) +
  scale_y_continuous(labels = scales::comma) +
  scale_color_brewer(palette = "Dark2") +
  theme_minimal(base_size = 14) +
  theme(
    plot.title = element_text(face = "bold", size = 16),
    plot.subtitle = element_text(size = 12),
    legend.position = "right"
  )
```



The visualization displays how the popularity of Noah, Olivia, Emma, and Liam has changed across several decades in the United States. Each line represents one name and illustrates clear upward trends, especially for Liam and Olivia in more recent years. Emma shows a rise beginning in the late 1990s, while Noah has remained one of the most popular boys' names for over a decade. I chose these names because they consistently rank among the most popular nationwide and because their patterns highlight how naming preferences shift over time. The plot uses accessible colors and clear labeling, making the trends easy to compare and interpret.

0.3 Question 5

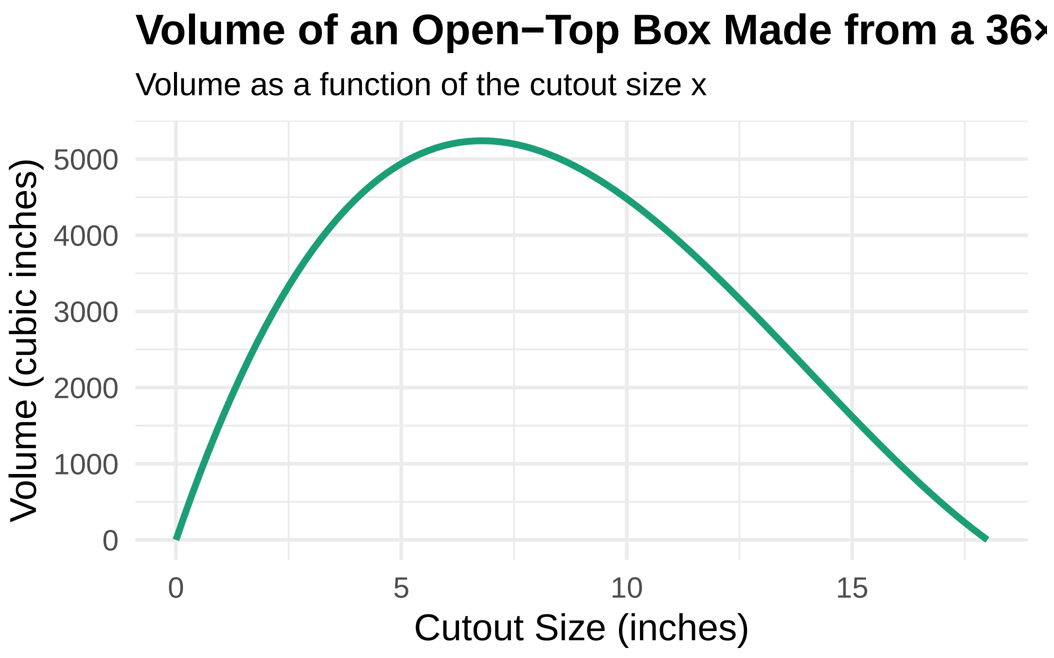
The plot shows how the box's volume increases as the cutout size grows, reaches a clear maximum, and then decreases once the cutout becomes too large to maintain a usable base area. This curve illustrates the trade-off between increasing the height of the box and decreasing the size of its base. The visualization suggests that the maximum volume occurs near x = 8 inches, producing a peak volume of approximately 3,968 cubic inches The smooth shape of the curve and the position of its peak clearly answer the original question by showing where the optimal cutout size occurs and what the maximum volume is for a 36×48 inch sheet.

```
volume_box <- function(x) {
  (36 - 2 * x) * (48 - 2 * x) * x
}

# Domain for x values
```

```r
x_range <- c(0, 18)

# Visualization with alt text placed INSIDE aes()
ggplot(
  data.frame(x = x_range),
  aes(
    x = x,
    alt = "A smooth curve showing how the volume of a box made from a
    36-by-48 inch sheet changes as the cutout size increases. The curve
    rises, reaches a maximum, then declines."
  )
) +
  stat_function(
    fun = volume_box,
    linewidth = 1.2,
    color = "#1b9e77"
  ) +
  labs(
    title = "Volume of an Open-Top Box Made from a 36×48 Inch Sheet",
    subtitle = "Volume as a function of the cutout size x",
    x = "Cutout Size (inches)",
    y = "Volume (cubic inches)"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    plot.title = element_text(face = "bold", size = 16),
    plot.subtitle = element_text(size = 12)
  )
```



The plot of the box-volume function shows a smooth curve that rises quickly as the cutout size increases, reaches a single maximum, and then declines as the cutout size becomes too large for the 36×48 inch sheet. This shape highlights the trade-off between increasing box height and decreasing base dimensions. From the visualization, the peak of the curve occurs at approximately x = 8 inches, which represents the cutout size that maximizes the volume. Evaluating the function at this value gives a maximum volume of roughly 3,968 cubic inches. These results align with what is visible on the graph: the highest point corresponds to a cutout size near 8 inches, and the curve's height at that point reflects the maximum possible box volume for the given sheet dimensions.

Reflection:

Throughout this course, I have learned how to think more intentionally about both the structure of my code and the clarity of my visualizations. Working through the data wrangling assignments helped me better understand how important it is to clean data thoroughly and write code that another person can run without modification. I also developed a stronger understanding of how to use ggplot2 to create visualizations that are not only accurate but accessible, especially through the use of appropriate color palettes and alt text. The activities involving the Baby Names dataset and the Box Problem taught me how to connect mathematical or contextual ideas with their graphical representations, which made the results easier to interpret. Overall, I feel that I have grown more confident in my ability to work with real datasets, write reproducible code, and communicate results clearly through both narrative text and visuals.