STAT 184 Activity 14

Reginald Acierto

2025-11-11

1 US Armed Forces

The frequency table (Table 1) analyzes US Armed Forces personnel data. The data was processed and wrangled to include rank information. The code for the data wrangling can be seen in the code appendix at the bottom of the document.

Table 1: US Navy Officers by Rank and Sex (Counts and Row Percentages)

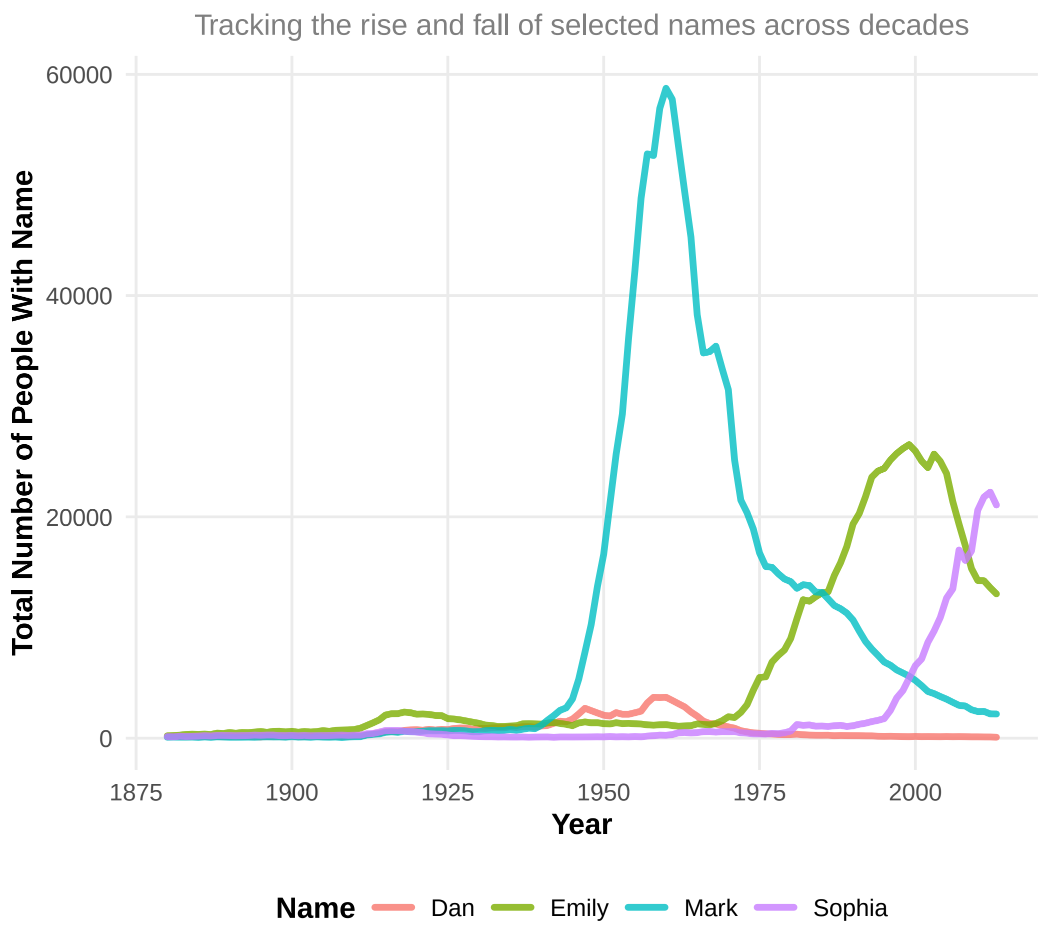
Rank	Male	Female	Total
Ensign	5,497 (75.7%)	1,766 (24.3%)	7,263 (100.0%)
Lieutenant Junior Grade	5,544 (76.4%)	1,716 (23.6%)	7,260 (100.0%)
Lieutenant	14,480 (75.0%)	4,830 (25.0%)	19,310 (100.0%)
Lieutenant Commander	7,983 (77.6%)	2,306 (22.4%)	10,289 (100.0%)
Commandar	5,525 (82.8%)	1,151 (17.2%)	6,676 (100.0%)
Captain	2,644 (85.4%)	452 (14.6%)	3,096 (100.0%)
Rear Admiral (Lower)	101 (95.3%)	5 (4.7%)	106 (100.0%)
Rear Admiral (Upper)	62 (91.2%)	6 (8.8%)	68 (100.0%)
Vice Admiral	32 (94.1%)	2 (5.9%)	34 (100.0%)
Admiral	8 (100.0%)	0 (0.0%)	8 (100.0%)
Total	41,876 (77.4%)	12,234 (22.6%)	54,110 (100.0%)

This frequency table shows the counts of men and women in US Navy Officer positions. The table provides information that sex and rank are not independent. Overall, proportionally there are much more males in officer positions than women for all the ranks, with 25% being the highest proportion of females (Lieutenant). As the ranks increase, the proportion of females in these ranks decreases dramatically, going from 22-25% to 14-17% at the middle ranks, and going all the way to 0% at the highest rank. These statistics show that the proportion of a given sex is dependent on the rank.

2 Popular Baby Names

The following visualization (Figure 1) analyzes the BabyNames dataset from the dcData package in Rstudio. The code for the data wrangling as well as the visualization can be seen in the code appendix.

Figure 1: Popularity of Names Over Time: 1880-2013

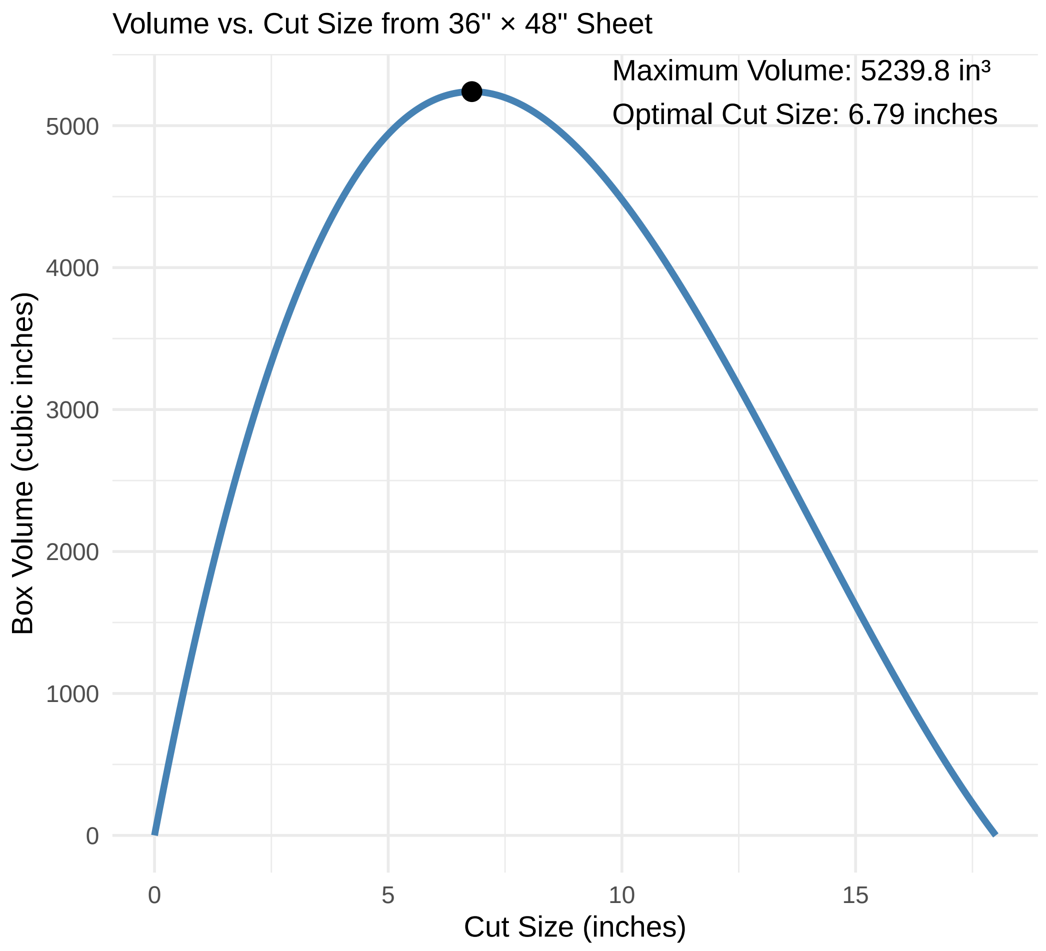


This visualization is a time series plot that shows the patterns in popularity of four names: Dan, Emily, Mark and Sophia. The data was collected over time from 1880 to 2013. I selected four popular male and female names to allow for a balanced comparison of popularity trends across genders. The graph shows that Mark had a significant peak around the 1950s and declined, reaching almost 60,000, while Emily and Sophia peaked later on in the 2000's, and only reaching 20,000-25,000 people. It also shows that Dan remained low throughout the timeline. From this visualization, we can learn the association between the popularity of names and the cultural trends at the time.

3 Box Problem

The box problem is a famous mathematics problem in which involves finding the optimal cut size from each corner of a sheet of paper that results in the maximum possible volume. The graph seen in Figure 2 shows a visual representation of the optimal cut size. The function to calculate this volume as well as the visualization can be seen in the code appendix.

Figure 2: Box Volume Optimization



This visualization shows the possible volumes from a range of cut sizes on a 36" x 48" sheet of paper. Through this plot, we can determine the cut size that creates the box with the maximum volume possible. Starting at 0, the volume increases greatly as the cut size increases. This makes sense because we create depth by increasing the height of the box. However at some point the graph peaks and starts decreasing. This happens because the cut size is so large that the base of the box is small and sacrifices volume even though the height is still increasing. We can learn that the optimal cut size happens at the peak of this graph. The code of this visualization calculates the optimal cut size, and we can see the point of this at the peak of the graph. The optimal cut size is approximately 6.79 inches and creates a box with a volume of 5239.8 in^3.

4 What I have learned so far

Learning a new programming language has always been difficult for me, and this one is no exception. However, I learned numerous skills in this course that helped me learn and understand R and helped make me a better programmer and data analyst. One of the biggest things I learned in this course was the process of planning, coding, improving, and polishing. In previous coding courses, I never planned out my code, and it was always frustrating being stuck when writing out code. Learning how to take time to create a detailed plan really helped understand what the assignment is and what I want my code to do. I also learned a lot from this course focusing a lot about organizing and commenting my code. It was difficult at first, but over the last few months I was able to learn how to organize and comment my code and get used to doing that. These are some of the things that I know that it is important to do when writing code, but I never done it. Now I am glad to have learned these skills because being organized and having a plan will make coding analyzing data much easier.

5 Code Appendix

```r
# -----------------------------------------------
# Armed Forces Data Wrangling
# This code processes armed forces data from two sources and creates two tidied data frames
# One where a case = group and one where a case = soldier
# The data frames include rank information
# -----------------------------------------------

# Load Packages
library(tidyverse)
library(rvest)
library(googlesheets4)

# Scrape Rank Data
ranks_web <- read_html("https://neilhatfield.github.io/Stat184_PayGradeRanks.html") %>%
  html_elements(css = "table") %>% # Extract all HTML tables
  html_table() # Convert tables to data frames

ranks_raw <- ranks_web[[1]] # Extract the first table


# Ranks Data Wrangling
ranks_raw[1, 1] <- "Type"  # Fill empty first cell
rank_headers <- ranks_raw[1, ]  # Extract header row
names(ranks_raw) <- rank_headers[1,]  # Set column names
ranks_raw <- ranks_raw[-c(1, 26), ]  # Remove header row and last row

ranks_tidy <- ranks_raw %>%
  select( # Removes Unnecessary Columns
    -Type,
    -`Coast Guard`
  ) %>%
  pivot_longer( # Wide to long format
    cols = !`Pay Grade`,
    names_to = "Branch", # Army, Navy, etc. go to Branch column
    values_to = "Rank" # Rank names go to Rank column
  ) %>%
  mutate(
    Rank = na_if(x = Rank, y = "--")  # Convert "--" to NA
  )


# Load Armed Forces Data
gs4_deauth()  # Access Google Sheets without authentication
```

```
armed_forces_raw <- read_sheet(
  ss = "https://docs.google.com/spreadsheets/d/19xQnI1cBh6Jkw7eP8YQuuicMlVDF7Gr-nXCb5qbwb_E/edi
  col_names = FALSE, # No column names
  skip = 2, # Skip the first two rows
  n_max = 32, # 32 rows in total
  na = c("N/A*")  # Treat "N/A*" as missing values
)


## Data Wrangling
tidy_groups <- armed_forces_raw %>%
  rename( # Assign column names
    Pay_Grade = ...1,
    army_male = ...2,
    army_female = ...3,
    army_total = ...4,
    navy_male = ...5,
    navy_female = ...6,
    navy_total = ...7,
    mc_male = ...8,
    mc_female = ...9,
    mc_total = ...10,
    af_male = ...11,
    af_female = ...12,
    af_total = ...13,
    sp_male = ...14,
    sp_female = ...15,
    sp_total = ...16,
    total_male = ...17,
    total_female = ...18,
    total_total = ...19
  ) %>%
  filter( # Filter out unnecessary rows
    Pay_Grade != "Pay Grade",
    Pay_Grade != "Total Enlisted",
    Pay_Grade != "Total Warrant Officers",
    Pay_Grade != "Total Officers",
    Pay_Grade != "Total",
    Pay_Grade != "Source: DMDC Active-Duty Military Personnel Master File (June 2025)"
  ) %>%
  select( # Remove columns with total
    -contains("total")
  ) %>%
  pivot_longer( # Combines the columns into service branch, gender, and count
    cols = -Pay_Grade,
    names_to = c("Branch", "Gender"),  # Split column names into two variables
    names_sep = "_",
    values_to = "Count" # Stores in Count column
  ) %>%
  mutate(
    Branch = case_when( # Expands abbreviation
      Branch == "army" ~ "Army",
      Branch == "navy" ~ "Navy",
      Branch == "mc" ~ "Marine Corps",
      Branch == "af" ~ "Air Force",
      Branch == "sp" ~ "Space Force"
    )
  ) %>%
  mutate( # Capitalizes value names
    Gender = case_when(
      Gender == "male" ~ "Male",
      Gender == "female" ~ "Female"
    )
  ) %>%
  mutate(
    across(Count, ~ifelse(is.na(.), 0, .)), # Replace NA values with 0
    Count = as.numeric(Count) # Changes values in Count column to numeric
  )

# Merge rank information with armed forces data
armed_forces_ranked_groups <- left_join(
  x = tidy_groups, # armed forces data
  y = ranks_tidy, # rank information
  by = join_by(Pay_Grade == `Pay Grade`, Branch == Branch)
)

# Expand into individual soldier rows
armed_forces_ranked_soldier <- armed_forces_ranked_groups %>%
  uncount(
    weights = Count
  )
# -------------------------------------------------------
# This code creates a two way frequency table to analyze
# the relationship between rank and sex among Navy officers
# -------------------------------------------------------

# Load Packages
library(tidyverse) # For data wrangling
library(kableExtra) # For creating professional tables
library(janitor) # For tabyl() function

# Filter armed forces data
navy_officers_data <- armed_forces_ranked_soldier %>%
  filter(
```

```r
    str_starts(Pay_Grade, "O"), # Select only officers
    Branch == "Navy" # Select only Navy
  )

# Original order of ranks
rank_order <- unique(navy_officers_data$Rank) # Extracts Rank column

# Convert to factor with this order
navy_officers_data <- navy_officers_data %>%
  mutate( # Converts the Rank column to a categorical variable with original rank order
    Rank = factor(Rank, levels = rank_order)
  )

# Create frequency table
navy_officers_table <- navy_officers_data %>%
  tabyl(Rank, Gender) %>% # rows = rank, col = gender
  adorn_totals(c("row", "col")) %>% # add row and column totals
  adorn_percentages("row") %>% # convert counts to row percentages
  adorn_pct_formatting() %>% # format percentages
  adorn_ns(position = "front") %>% # add counts before percentages
  select(Rank, Male, Female, Total)

# Create Professional table
navy_officers_table %>%
  kable(
    align = c("l", "r", "r", "r") # text on left side, numbers on right side
  ) %>%
  kable_classic()
# -------------------------------------------------------------
# Baby Names Data Wrangling
# This code processes and wrangles the BabyNames dataset
# -------------------------------------------------------------

# Load packages
library(dcData) # For BabyNames data
library(tidyverse) # For data wrangling

# Data Wrangling
babynames_clean <- BabyNames %>%
  # Step 1: Filter by name and sex
  filter(
    (name == "Emily" & sex == "F") |
      (name == "Sophia" & sex == "F") |
      (name == "Mark" & sex == "M") |
      (name == "Dan" & sex == "M")
  ) %>%
  # Step 2: Arrange by name
  arrange(name)
# -------------------------------------------------------------
# This visualization analyzes the popularity trends of selected baby names over time
# Compares two popular female names and two popular male names
# -------------------------------------------------------------

# Data Visualization
babynames_clean %>%
  # Step 1: Create framework/set aesthetics
  ggplot(
    aes(
      x = year,
      y = count,
      color = name # different color for each name
    )
  ) +
  # Step 2: Geometries
  geom_line(
    linewidth = 1.2,
    alpha = 0.8,
    lineend = "round"
  ) +
  # Step 3: Set labels
  labs(
    subtitle = "Tracking the rise and fall of selected names across decades",
    x = "Year",
    y = "Total Number of People With Name",
    color = "Name"
  ) +
  # Step 4: Add themes
  theme_minimal() +
  theme(
    plot.title = element_text(face = "bold", size = 14, hjust = 0.5),
    plot.subtitle = element_text(hjust = 0.5, color = "gray50"),
    legend.position = "bottom",     # Better for multiple lines
    legend.title = element_text(face = "bold"),
    axis.title = element_text(face = "bold"),
    panel.grid.minor = element_blank(),  # Remove minor gridlines
  )
# ---------------------------------------------
# Box Problem
# This function calculates the volume of a box
# by cutting squares from each corner of the box
# ---------------------------------------------

box_volume <- function(cut_size) {
  # Defines original dimensions of paper in inches
  LENGTH = 48
  WIDTH = 36

  # Calculate dimensions of box
  height = cut_size
  length = LENGTH - (2 * cut_size)
  width = WIDTH - (2 * cut_size)

  # Calculates volume
  volume = length * width * height

  return(volume)
}
# ----------------------------------------------------------
# This code creates a plot showing how box volume changes with cut size
# and identifies the optimal cut that maximizes volume
# ----------------------------------------------------------

# Load packages
library(ggplot2)

# Calculate optimal volume and cut size
cut_range <- seq(0, 18, by = 0.01) # Create a sequence of possible cut sizes from 0 to 18
volumes <- box_volume(cut_range) # Calculate volume for each cut size
optimal_index <- which.max(volumes) # Find index of max volume
optimal_cut <- cut_range[optimal_index] # Extract cut size that give max volume
optimal_volume <- volumes[optimal_index] # Extract max volume

# Create visualization
ggplot() +
  stat_function(
    fun = box_volume, # Volume function
    geom = "line", # Plot as a line
    color = "steelblue", # Line color
    linewidth = 1.2, # Line thickness
    n = 1000  # Number of points to calculate (smoother curve)
  ) +
  geom_point( # Add optimal solution
    aes(
      x = optimal_cut,
      y = optimal_volume
    ),
    size = 3
  ) +
  annotate( # Text to display optimal value
    "text",
    x = optimal_cut + 3, # Position text 3 inches right of optimal point
    y = optimal_volume, # Position text at same height as optimal volume
    label = paste0("Maximum Volume: ", round(optimal_volume, 1), " in³\n",
                   "Optimal Cut Size: ", round(optimal_cut, 2), " inches"),
    size = 4,
    hjust = 0
  ) +
  # Customize labels and titles
  labs(
    subtitle = "Volume vs. Cut Size from 36\" × 48\" Sheet",
    x = "Cut Size (inches)",
    y = "Box Volume (cubic inches)"
  ) +
  # Set the x-axis limits
  xlim(0, 18) +  # From 0 to 18
  theme_minimal()
```