# Activity #14, STAT184

Isaac Swope

2025-11-15

## Armed Forces Data Wrangling Redux (Activities #08 and #10)

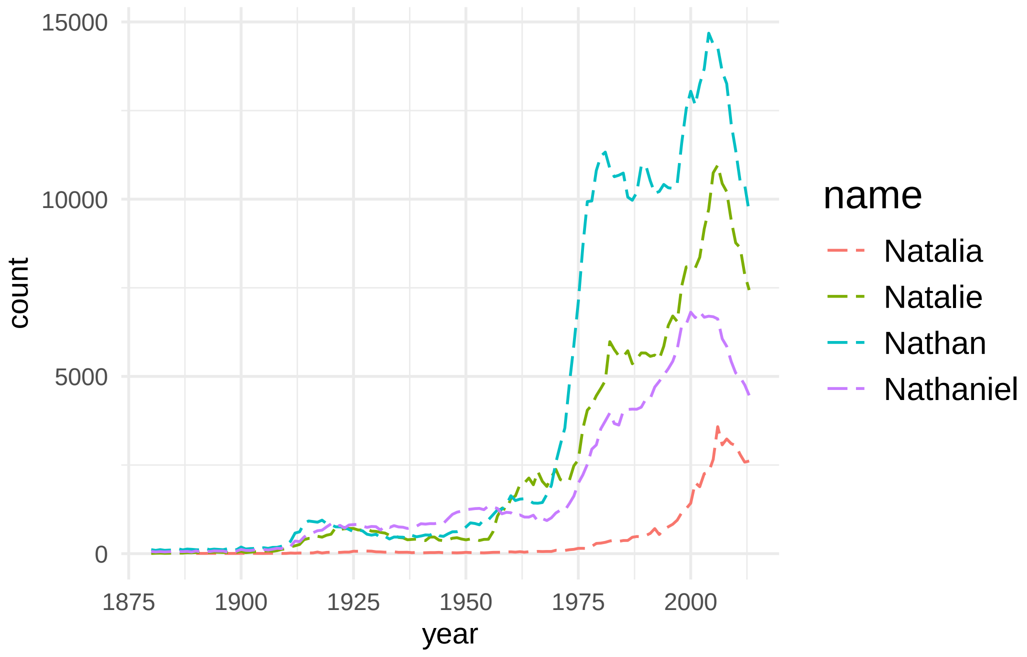
In this section, data wrangling and web scrapping is used to clean raw data. Code by Neil Hatfield can be found in the Code Appendix

## Popularity of Baby Names (Activity #13)

We will look at the popularity of four names given at birth. The data spans from 1880 to 2013 within the United States. The purpose of the following visualization is to compare names beginning with "Nat": two female, two male.

### Visualization

Figure 1: Frequency of Selected Names at Birth from 1880 - 2013

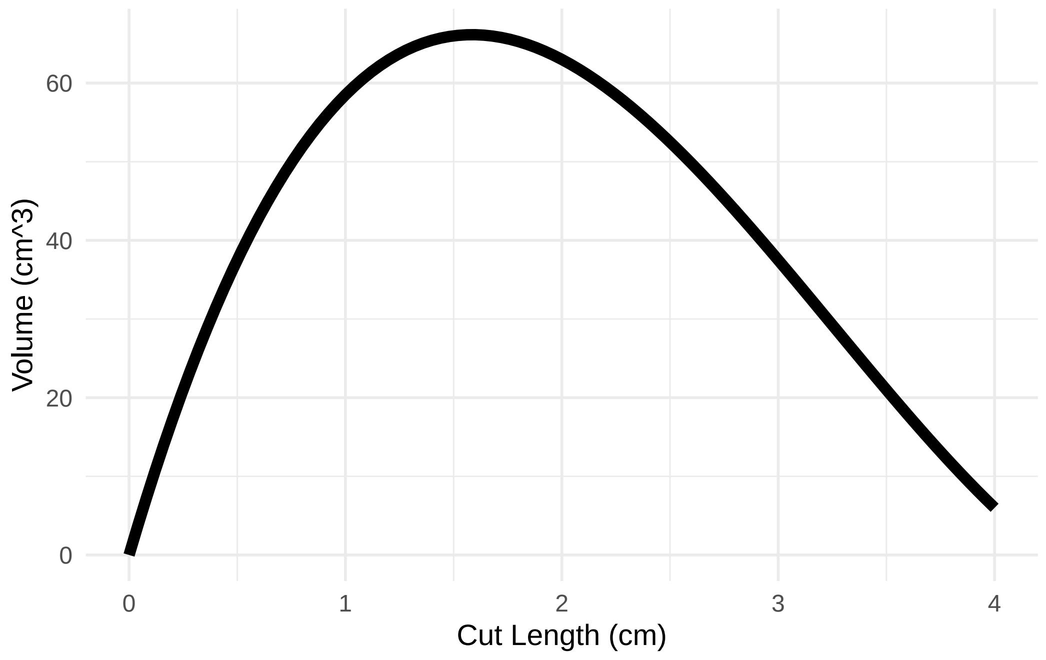


## Plotting a Mathematical Function (Activity #04)

A rectangular piece of paper with a length of 11cm and width of 8.5cm has a square cut out at each corner. The sides are then folded up to make an open top box. The purpose of this section is to visualize the relationship between the cuts and the volume in a box and find the cut that results in the largest volume.

### Graph

Figure 2: Volume by Cut Length



```
$optimal_cut
[1] 1.59

$max_volume
[1] 66.14782
```

The function defined in @box-problem-code, estimates the maximum volume of 66.15 cm^3 is attained with a cut length of 1.59 cm

## What I Feel I've Learned So Far

1. Planning

   i) Defining goals and needs
   ii) Writing out a plan

2. Data Wrangling

    i) Tidying ii)Scraping

3. Creating Visualizations

    i) Tables
    ii) ggplot

4. Using Quarto
5. Using Git

Overall I would say that I got an introduction to using R studio. I feel comfortable importing data and doing minor adjustments, I am still not fully confident in doing large data wrangling tasks that take many steps. I do feel comfortable creating visualizations and displays. I know the general guidelines to follow laid out by Tufte and Kosselyn. Creating displays, labeling, alt-text, and long descriptions are all things I know I can do. We just began using quarto, but it seems quite simple, and I will improve with more practice. We also started using Git, which will be a very helpful tool going forward. Within Github, so far I have learned making projects, creating branches, making commits, and doing pull requests. Overall I feel as though I am getting a good grasp of coding in R, but I still have so much more to practice and learn.

Code Appendix

```r
#|label: Loading Packages
#Coding Style: Tidyverse Style
library(tidyverse)
library(tidyr)
library(knitr)
library(ggplot2)
library(dcData)
library(googlesheets4)
library(rvest)


# Tidy Armed Forces Data ----
## Make two tied data frames for the June 2025 armed forces data
## 1 where case is a group of soliders and 1 where the case is
## an individual solider. Both need to have rank added.

# Step 1: Load Packages ----
library(tidyverse)
library(rvest)
library(googlesheets4)

# Step 2: Scrape Rank Data ----
webRanks <- read_html("https://neilhatfield.github.io/Stat184_PayGradeRanks.html") %>%
  html_elements(css = "table") %>%
  html_table()

rawRanks <- webRanks[[1]] # Extract the data frame of ranks

# Step 3: Wrangle Rank Data ----
## Enter a value in the first cell (1, 1)
rawRanks[1, 1] <- "Type"
## Extract actual column headers
rankHeaders <- rawRanks[1, ]
## Apply headers as column names
names(rawRanks) <- rankHeaders[1,]
## Remove redundant first row and last row
rawRanks <- rawRanks[-c(1, 26), ]

cleanRanks <- rawRanks %>%
  dplyr::select(!Type) %>% # Remove extra column
  pivot_longer(
    cols = !`Pay Grade`, # The improper name requires backticks
    names_to = "Branch",
    values_to = "Rank"
  ) %>%
  mutate(
    Rank = na_if(x = Rank, y = "--")
  )

# Step 4: Load Armed Forces Data ----
gs4_deauth() # Prevents needing to sign into a Google account
forcesHeaders <- read_sheet(
  ss = "https://docs.google.com/spreadsheets/d/19xQnI1cBh6Jkw7eP8YQuuicMlVDF7Gr-nXCb5qbwb_E/edi
  col_names = FALSE, # Turn off Column Names
  n_max = 3 # read only the first three rows
)

rawForces <- read_sheet(
  ss = "https://docs.google.com/spreadsheets/d/19xQnI1cBh6Jkw7eP8YQuuicMlVDF7Gr-nXCb5qbwb_E/edi
  col_names = FALSE, # Turn off Column Names
  skip = 3, # Skip the first three rows
  n_max = 28, # Read only the next 28 rows; drops footer
  na = c("N/A*") # Tells R to treat the N/A* as missing values
)


# Step 5: Wrangle Armed Forces Data ----
## Step 5a: Create good column names ----
### Pattern is Pay Grade followed by 3 columns for each branch in the order
### Army, Navy, Marine Corp, Air Force, Space Force, and Total
branchNames <- rep( # Create three copies of each branch
  x = c("Army", "Navy", "Marine Corps", "Air Force", "Space Force", "Total"),
  each = 3
)
tempHeaders <- paste( # Combine branch with other headers
  c("", branchNames),
  forcesHeaders[3,],
  sep = "."
)


names(rawForces) <- tempHeaders

## Wrangle Armed Forces Data ----
cleanForces <- rawForces %>%
  rename(Pay.Grade = `.Pay Grade`) %>%
  dplyr::select(!contains("Total")) %>% # Remove total columns
  filter( # Remove total rows; see note below
    Pay.Grade != "Total Enlisted" &
      Pay.Grade != "Total Warrant Officers" &
      Pay.Grade != "Total Officers" &
      Pay.Grade != "Total"
  ) %>%
  pivot_longer( # Reshape data
    cols = !Pay.Grade,
    names_to = "Branch.Sex",
    values_to = "Frequency"
  ) %>%
  separate_wider_delim( # Separate branches and sex
    cols = Branch.Sex,
    delim = ".",
    names = c("Branch", "Sex")
  )

# Step 6: Merge Data Frames ----
key_forcesRanks <- left_join(
  x = cleanForces,
  y = cleanRanks,
  by = join_by(Pay.Grade == `Pay Grade`, Branch == Branch)
)

# Step 7: Transform Group into Individual ----
key_individualRanks <- key_forcesRanks %>%
  filter(!is.na(Frequency)) %>% # Remove all cases with missing counts
  uncount(
    weights = Frequency
  )

#|label: Baby-Names-Wrangling

# Baby Names Wrangling

names_clean <- BabyNames %>%
  filter(name %in% c("Nathan", "Natalia", "Nathaniel", "Natalie")) %>% #filtering for selected
  group_by(name, year) %>% # grouping by name and year
  summarize(count = sum(count)) #sum count of sexes
# Creating Plot of Baby Names

ggplot(data = names_clean) +
  aes(
    x = year,
    y = count,
    color = name
  ) +
  geom_line(
    linewidth = 0.5,
    linetype = "longdash"
  ) +
  theme_minimal() +
  theme(
```

```r
    legend.text = element_text(size = 12),
    legend.title = element_text(size = 15)
  )

#Creaing Graph
ggplot(data.frame(x = c(0, 4)) , aes(x = x)) +
  stat_function(fun = function(x) x*(11-2*x)*(8.5-2*x), size = 2) +
  theme_minimal() +
  labs(y = "Volume (cm^3)", x = "Cut Length (cm)")

# Getting Max Volume and Optimum Cut Length for the Box Problem

getVolume <- function(x, w = 8.5, l = 11){
  V<- x*(l-2*x)*(w-2*x)
  return(V)
}

solve_box_problem <- function(length, width, step = 0.01) {
  # Define the range of possible cut sizes (x)
  max_cut <- min(length, width) / 2
  x_vals <- seq(0, max_cut, by = step)


  # Compute volume for each x
  volumes <- (length - 2 * x_vals) * (width - 2 * x_vals) * x_vals


  # Find the maximum volume and corresponding x
  max_index <- which.max(volumes)
  optimal_x <- x_vals[max_index]
  max_volume <- volumes[max_index]


  # Return results
  return(list(
    optimal_cut = optimal_x,
    max_volume = max_volume
  ))
}
solve_box_problem(length=11, width=8.5)
```