Activity 14

Harvey Barnes

Table of contents

0.1 Armed Forces: Data Wrangling

In this Section I wrangle the US armed forces data into a data frame where each row represents a single service member including their branch, sex and rank

```r
library(dplyr)
library(tidyr)
library(knitr)

# Read the Armed Forces from google sheets
armed_url <- "https://docs.google.com/spreadsheets/d/19xQnI1cBh6Jkw7eP8YQuuicMlVDF7Gr-nXCb5q

armed_raw <- read.csv(
  file = armed_url,
  check.names = FALSE,
  na.strings = c("", "NA")
)

# Remove the first two haeder rows
armed_data <- armed_raw[-c(1, 2), ]

#new column names
names(armed_data) <- c(
  "PayGrade",
  "Army_Male",     "Army_Female",     "Army_Total",
  "Navy_Male",     "Navy_Female",     "Navy_Total",
  "Marines_Male",  "Marines_Female",  "Marines_Total",
  "AirForce_Male", "AirForce_Female", "AirForce_Total",
  "Space_Male",    "Space_Female",    "Space_Total",
  "DoD_Male",      "DoD_Female",      "DoD_Total"
)

# Convert all count cols to numeric
armed_data <- armed_data |>
  dplyr::mutate(
    dplyr::across(
      .cols = -PayGrade,
      .fns = ~ as.numeric(gsub(",", "", .))
    )
  )

# Pivot longer with branch, sex, and count
armed_long <- armed_data |>
  dplyr::select(-dplyr::ends_with("Total")) |>
  tidyr::pivot_longer(
    cols = -PayGrade,
    names_to = c("Branch", "SexOrTotal"),
    names_sep = "_",
    values_to = "Count"
  ) |>
  dplyr::filter(SexOrTotal %in% c("Male", "Female")) |>
  dplyr::rename(Sex = SexOrTotal) |>
  dplyr::filter(!is.na(Count), Count > 0)

# Expand so that each row is one individual soldier
armed_soldiers <- armed_long |>
  tidyr::uncount(
    weights = Count,
    .remove = TRUE
  ) |>
  dplyr::rename(Rank = PayGrade)
```

Choose Subgroup and Display table:

```r
# Choose a subgroup: Army enlisted ranks (E1-E9)
army_enlisted <- armed_soldiers |>
  dplyr::filter(
    Branch == "Army",
    grepl(pattern = "^E[0-9]", x = Rank)
  )

# Create a frequency table
army_counts <- army_enlisted |>
  dplyr::count(Rank, Sex)

army_table <- army_counts |>
  tidyr::pivot_wider(
    names_from = Sex,
    values_from = n,
    values_fill = 0
  ) |>
  dplyr::arrange(Rank)

army_table
```

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

Army Enlisted Male vs Female:

In the table, we can see the distribution of male and female enlisted soldiers across the E1–E9 pay grades in the U.S. Army. Overall, male soldiers outnumber female soldiers at every enlisted rank. Although women are present at all pay grades, their counts are much smaller, and the gap between male and female soldiers is fairly consistent across ranks. This pattern suggests that sex and rank do not appear to be independent within this subgroup, because the representation of women is not proportionally similar across the enlisted pay grades when compared to men.

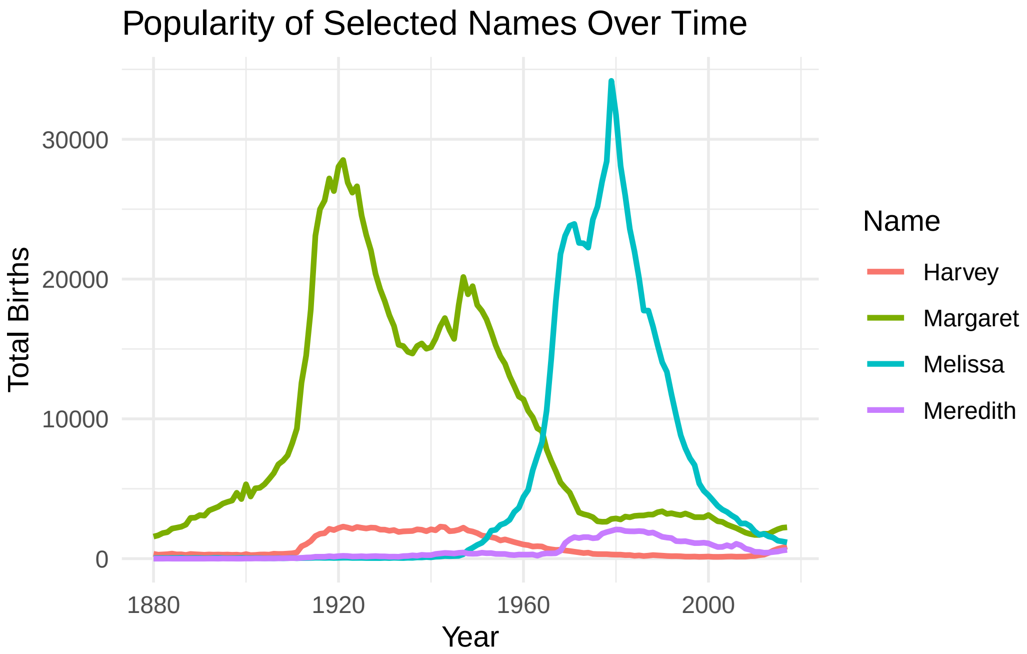
0.2 Activity 13 Baby Names:

Select 4 names from data set and filter for them. Get counts for each by year and put into a final dataframe

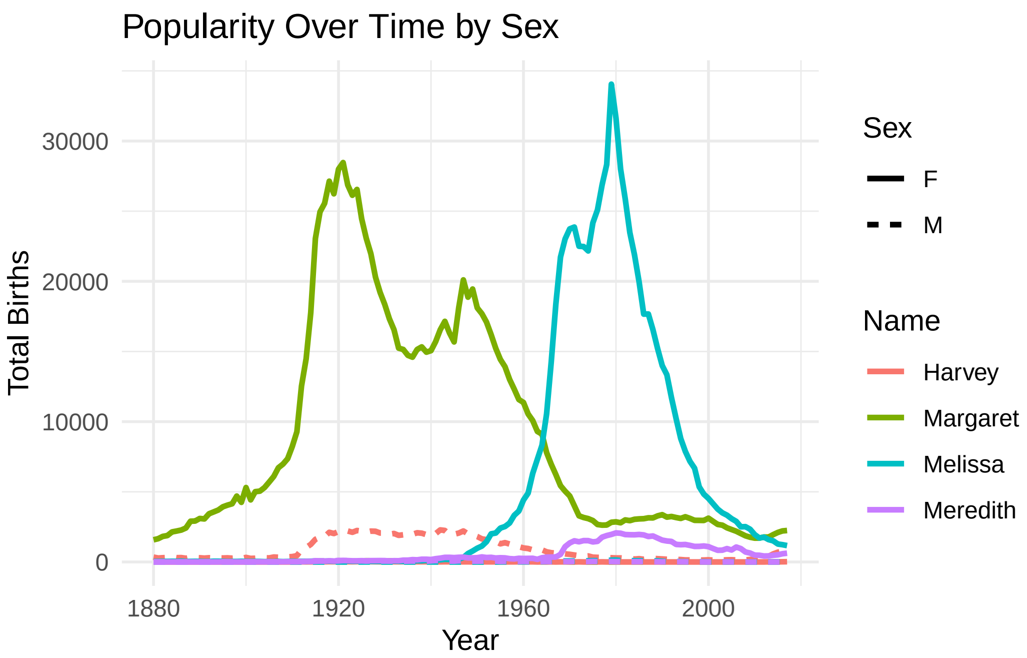
```r
 #install.packages("babynames")
 #install.packages("dplyr")
 #install.packages("tidyr")

library(babynames)
library(dplyr)
library(tidyr)

# Names of interest
my_names <- c("Harvey", "Meredith", "Melissa", "Margaret")

# Counts by name, sex, and year (fill in missing years with 0)
df_by_sex <- babynames |>
  filter(name %in% my_names) |>
  group_by(name, sex, year) |>
  summarise(total_n = sum(n), .groups = "drop") |>
  complete(
    name,
    sex,
    year = full_seq(year, period = 1),
    fill = list(total_n = 0)
  )

# Combined counts over sex
df_combined <- df_by_sex |>
  group_by(name, year) |>
  summarise(total_n = sum(total_n), .groups = "drop") |>
  complete(
    name,
    year = full_seq(year, period = 1),
    fill = list(total_n = 0)
  )
```

Create Visual Showing selected 4 names over time:

```r
library(ggplot2)

ggplot(df_combined, aes(year, total_n, color = name)) +
  geom_line(linewidth = 1) +
  labs(title = "Popularity of Selected Names Over Time",
       x = "Year", y = "Total Births", color = "Name") +
  theme_minimal()
```



```r
ggplot(df_by_sex, aes(year, total_n, color = name, linetype = sex)) +
  geom_line(linewidth = 1) +
  labs(title = "Popularity Over Time by Sex",
       x = "Year", y = "Total Births", color = "Name", linetype = "Sex") +
  theme_minimal()
```



Baby Names Vs Time and Sex:

The visualization tracks how 4 names rise and fall in popularity over the 1880s to 2020s. The y axis is total births and x axis is year. Melissa shows a boom in the 1960s to 1990s then falls fast. Harvey peaks between 1910 and 1960. Nathan peaks around 2010. Meredith peaks around 1980 and remains steady . Read solid as females and dashed as males. Showcasing that for each name numbers are heavily influenced by gender.

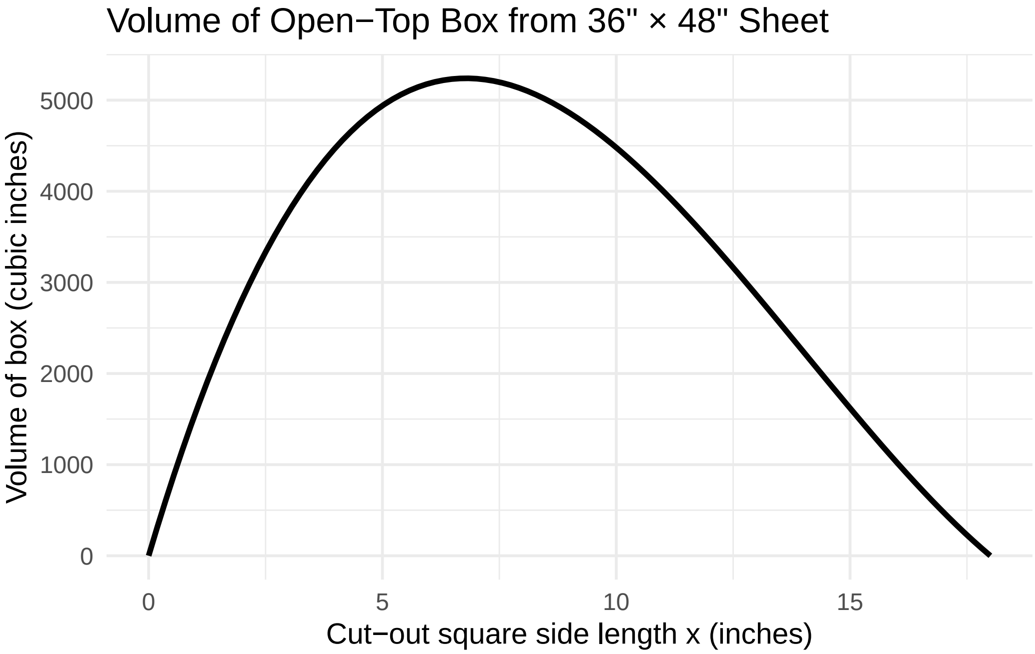
0.3 Folding Box Problem:

Create Function First to find the volume given the cut lengths

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

Create Graph:

```
box_domain <- data.frame(x = c(0, 18))

# Plot the volume function
ggplot(data = box_domain, mapping = aes(x = x)) +
  stat_function(
    fun = V_box,
    linewidth = 1
  ) +
  labs(
    x = "Cut-out square side length x (inches)",
    y = "Volume of box (cubic inches)",
    title = "Volume of Open-Top Box from 36\" × 48\" Sheet"
  ) +
  theme_minimal()
```



Cut out Length vs Volume:

In the graph for the box problem, the curve shows how the volume of the open-top box changes as the cut-out square side length x increases from 0 to 18 inches. The volume starts at 0 when x = 0 (no cuts, so there is no box), rises smoothly to a single highest point, and then decreases back to 0 as x approaches 18 inches. This shape makes sense in context: when x is very small, the box is almost flat and has very little volume; when x is very large, the base of the box becomes too small to hold much.

From the peak of the curve, we find that the maximum volume occurs when the cut-out squares have a side length of about 6.8 inches. At this value of x, the box made from a 36 inch by 48 inch sheet has a maximum volume of about 5,240 cubic inches. So, to get the largest possible box from this piece of paper, we should cut out squares of roughly 6.8 inches from each corner.

0.4 Reflection:

I have learned how to properly investigate data using R. So far we have learned an abundance of topics from functions, to ggplot2. Everything we have learned has been extermeley useful and has real world applications. The immense amount of time spent on data manipulation in order to clean and reformat data has been a huge addition to my skill set. GGplot2 is also a valuable asset to create plots that can portray rich complicated data in order to add proof to my claims. The activities involving wrangling and drawing conclusions from data has been incredibly helpful to expand my r tool kit.