Activity_14

2025-11-10

#1 Armed Forces Data Wrangling Redux

Frequency Table based on Subset of Soliders

```
# Load Packages

library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.2     v tibble    3.3.0
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.1.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(rvest)
```

```
Attaching package: 'rvest'

The following object is masked from 'package:readr':

    guess_encoding
```

```r
# Load Data sources

sheet_url <- "https://docs.google.com/spreadsheets/d/19xQnI1cBh6Jkw7eP8YQuuicMlVDF7Gr-nXCb5q
ranks_url <- "https://neilhatfield.github.io/Stat184_PayGradeRanks.html"

# Load the Sheet & build headers

raw <- readr::read_csv(sheet_url, col_names = FALSE, show_col_types = FALSE)
if (ncol(raw) < 2) raw <- readr::read_delim(sheet_url, delim = ",", col_names = FALSE, show_

row1 <- as.character(unlist(raw[1, ]))
row2 <- as.character(unlist(raw[2, ]))

blank    <- (row1 == "" | is.na(row1))
last_ix  <- cumsum(!blank)
row1fill <- row1
vals     <- row1[!blank]
pos      <- which(blank & last_ix > 0)
row1fill[pos] <- vals[last_ix[pos]]

sex2 <- ifelse(row2[-1] == "" | is.na(row2[-1]), "Total", row2[-1])
nm   <- c("pay_grade", paste(row1fill[-1], sex2, sep = " - ") |> stringr::str_squish())
suf  <- ave(seq_along(nm), nm, FUN = seq_along)
nm   <- ifelse(suf == 1, nm, paste0(nm, "_", suf))

df <- as.data.frame(raw[-c(1,2), ], stringsAsFactors = FALSE)
names(df) <- nm

# Begin Tidy data format

groups <- df %>%
  filter(!str_detect(pay_grade, regex("^Total", ignore_case = TRUE))) %>%
  mutate(
    pay_grade = pay_grade |> stringr::str_to_upper() |> stringr::str_replace_all("[-\\s]", "
  ) %>%
  pivot_longer(-pay_grade, names_to = "col", values_to = "n_raw") %>%
  mutate(
    col  = stringr::str_squish(col),
    sex  = case_when(
      str_detect(col, regex("\\bMale\\b",   TRUE)) ~ "Male",
      str_detect(col, regex("\\bFemale\\b", TRUE)) ~ "Female",
      TRUE ~ "Total"
```

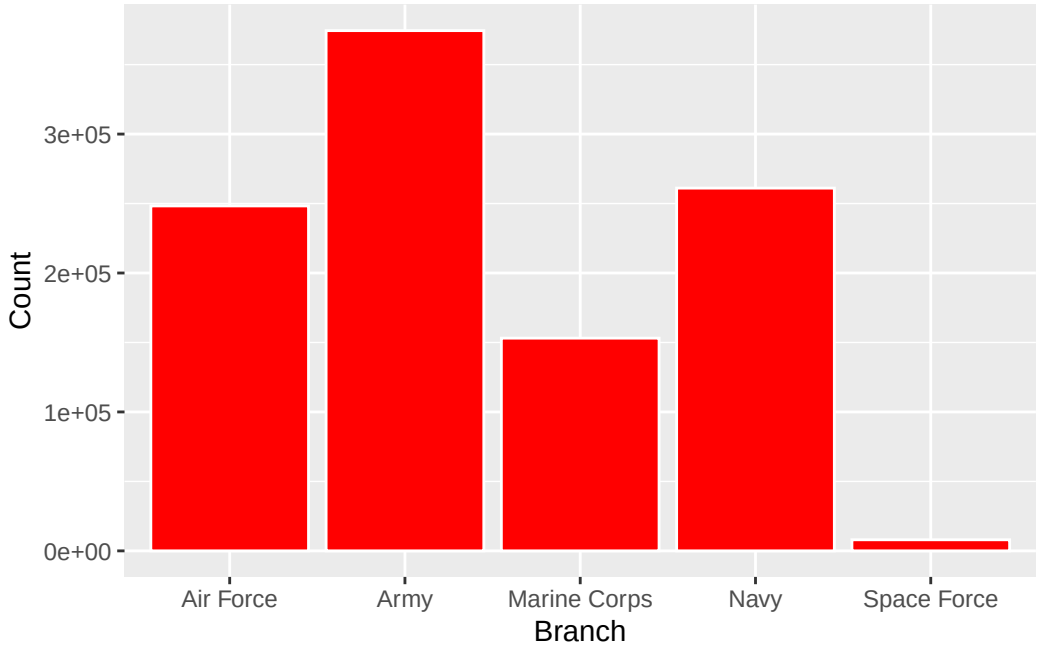
```
    ),
    branch = case_when(
      str_detect(col, regex("Coast\\s*Guard", TRUE)) ~ "Coast Guard",
      str_detect(col, regex("Space\\s*Force", TRUE)) ~ "Space Force",
      str_detect(col, regex("Air\\s*Force",   TRUE)) ~ "Air Force",
      str_detect(col, regex("Marine|USMC",    TRUE)) ~ "Marine Corps",
      str_detect(col, regex("Navy",           TRUE)) ~ "Navy",
      str_detect(col, regex("Army",           TRUE)) ~ "Army",
      TRUE ~ NA_character_
    ),
    n_raw = stringr::str_squish(n_raw),
    n = suppressWarnings(
      readr::parse_number(n_raw, na = c("Male","Female","Total","-","-",""))
    )
  ) %>%
  filter(!is.na(branch), !is.na(n), n > 0) %>%
  mutate(n = as.integer(n)) %>%
  select(branch, sex, pay_grade, n)

# Scrape rank titles

ranks_raw <- read_html(ranks_url) %>%
  html_elements("table") %>%
  html_table(fill = TRUE) %>%
  bind_rows()
```

```
New names:
* `` -> `...1`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...3`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...4`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...5`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...6`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...7`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...8`
```

```
ranks <- ranks_raw %>%
  rename(pay_grade = 1) %>%
  mutate(across(everything(), ~ .x |> as.character() |> stringr::str_squish())) %>%
  select(where(~ any(!is.na(.)))) %>%                          # drop fully-empty cols
  filter(stringr::str_detect(pay_grade, "^(?i)[EWO][- ]?\\d+")) %>%
  mutate(
```

```
    pay_grade = pay_grade |> stringr::str_to_upper() |> stringr::str_replace_all("[-\\s]", "
  ) %>%
  pivot_longer(-pay_grade, names_to = "branch_raw", values_to = "rank_title") %>%
  mutate(
    branch = case_when(                                          # normalize branches
      str_detect(branch_raw, regex("Coast\\s*Guard", TRUE)) ~ "Coast Guard",
      str_detect(branch_raw, regex("Space\\s*Force", TRUE)) ~ "Space Force",
      str_detect(branch_raw, regex("Air\\s*Force",   TRUE)) ~ "Air Force",
      str_detect(branch_raw, regex("Marine|USMC",    TRUE)) ~ "Marine Corps",
      str_detect(branch_raw, regex("Navy",           TRUE)) ~ "Navy",
      str_detect(branch_raw, regex("Army",           TRUE)) ~ "Army",
      TRUE ~ NA_character_
    ),
    rank_title = na_if(stringr::str_squish(rank_title), "")
  ) %>%
  filter(!is.na(branch), !is.na(rank_title)) %>%
  select(pay_grade, branch, rank_title)


# Join groups

df_groups <- groups %>%
  left_join(ranks, by = c("pay_grade","branch")) %>%
  relocate(rank_title, .after = pay_grade)

# Build individuals table

df_individuals <- df_groups %>% uncount(n, .remove = TRUE)
```

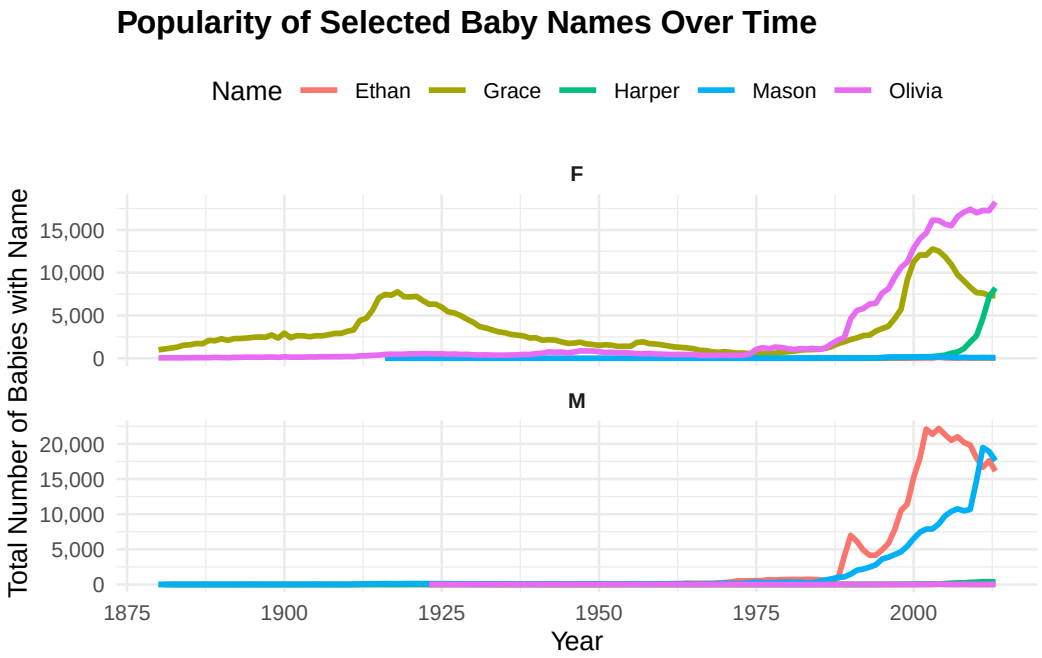
```
library(ggplot2)
ggplot(df_individuals, aes(x = factor(branch))) +
  geom_bar(fill = "red", color = "white") +
  labs(x = "Branch", y = "Count")
```



#2 Popularity of Baby Names

```
library(tidyverse)
library(dcData)

chosen_names <- c("Mason", "Olivia", "Harper", "Ethan", "Grace")

baby <- BabyNames %>%
  filter(name %in% chosen_names) %>%
  group_by(year, sex, name) %>%
  summarise(count = sum(count, na.rm = TRUE), .groups = "drop")

ggplot(baby, aes(x = year, y = count, color = name)) +
  geom_line(linewidth = 1) +
  facet_wrap(~ sex, nrow = 2, scales = "free_y") +
  scale_y_continuous(labels = scales::comma) +
  labs(
    title = "Popularity of Selected Baby Names Over Time",
    x = "Year",
    y = "Total Number of Babies with Name",
    color = "Name"
```

```
  ) +
  theme_minimal(base_size = 10) +
  theme(
    legend.position = "top",
    plot.title = element_text(face = "bold"),
    strip.text = element_text(face = "bold")
  )
```



Why did I Choose These Names?

I wanted to have a set of names that included both boys and girls but also used my own prior knowledge to think of names that I believed to be popular throughout time. After creating the visualization I learned that Grace was the most populair historical name, but Mason is one of the most popular names today.

#3 Plotting a Mathematical Function

R Code to Solve the Volume of a Box

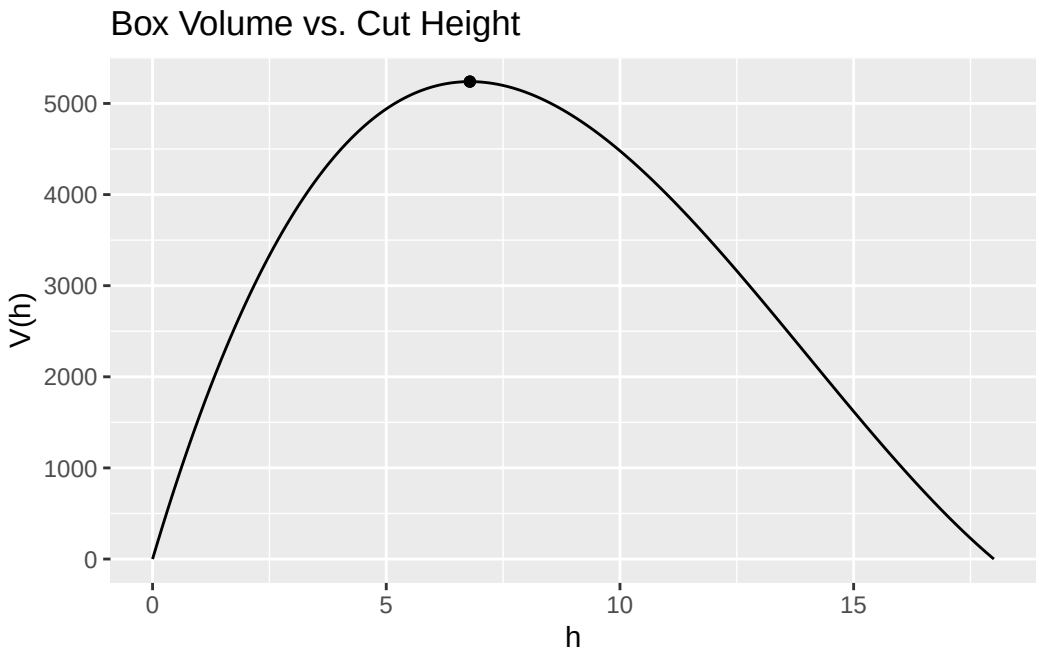
Volume Box Plot

```
volume_fn <- function(L = 36, W = 48) {
  function(h) (L - 2*h) * (W - 2*h) * h
}

volume_box <- function(L = 36, W = 48) {
  V <- function(h) (L - 2*h) * (W - 2*h) * h
  Hfinal <- optimize(V, interval = c(0, min(L, W)/2), maximum = TRUE)
  list(Max_height = Hfinal$maximum, Max_volume = Hfinal$objective)
}
volume_box()
```

```
$Max_height
[1] 6.788902

$Max_volume
[1] 5239.819
```

```
L <- 36
W <- 48
res <- volume_box(L, W)
Vh <- volume_fn(L, W)


ggplot(data.frame(x = c(0, min(L, W)/2)), aes(x = x)) +
  stat_function(fun = Vh, n = 1000) +
  geom_point(aes(x = res$Max_height, y = res$Max_volume)) +
  labs(x = "h", y = "V(h)", title = "Box Volume vs. Cut Height")
```

```
Warning in geom_point(aes(x = res$Max_height, y = res$Max_volume)): All aesthetics have leng
i Please consider using `annotate()` or provide this layer with data containing
  a single row.
```



#4 What I've Learned in This Class

I've learned to use not only basic R functions but also how to leverage existing R packages to make processes easier and more effective. Moreover, I've learned some of the best practices for creating visualizations, whether it be histograms, boxplots, or scatter plots. Also, I've learned how to work with data, which is something I enjoy, as in the real world, I will be working with data to try and solve problems.

Code Appendix

#1, Soldiers Wrangling

```
# Load Packages

library(tidyverse)
library(rvest)

# Load Data sources

sheet_url <- "https://docs.google.com/spreadsheets/d/19xQnI1cBh6Jkw7eP8YQuuicMlVDF7Gr-nXCb5q
```

```r
ranks_url <- "https://neilhatfield.github.io/Stat184_PayGradeRanks.html"

# Load the Sheet & build headers

raw <- readr::read_csv(sheet_url, col_names = FALSE, show_col_types = FALSE)
if (ncol(raw) < 2) raw <- readr::read_delim(sheet_url, delim = ",", col_names = FALSE, show_

row1 <- as.character(unlist(raw[1, ]))
row2 <- as.character(unlist(raw[2, ]))

blank    <- (row1 == "" | is.na(row1))
last_ix  <- cumsum(!blank)
row1fill <- row1
vals     <- row1[!blank]
pos      <- which(blank & last_ix > 0)
row1fill[pos] <- vals[last_ix[pos]]

sex2 <- ifelse(row2[-1] == "" | is.na(row2[-1]), "Total", row2[-1])
nm   <- c("pay_grade", paste(row1fill[-1], sex2, sep = " - ") |> stringr::str_squish())
suf  <- ave(seq_along(nm), nm, FUN = seq_along)
nm   <- ifelse(suf == 1, nm, paste0(nm, "_", suf))

df <- as.data.frame(raw[-c(1,2), ], stringsAsFactors = FALSE)
names(df) <- nm

# Begin Tidy data format

groups <- df %>%
  filter(!str_detect(pay_grade, regex("^Total", ignore_case = TRUE))) %>%
  mutate(
    pay_grade = pay_grade |> stringr::str_to_upper() |> stringr::str_replace_all("[-\\s]", "
  ) %>%
  pivot_longer(-pay_grade, names_to = "col", values_to = "n_raw") %>%
  mutate(
    col  = stringr::str_squish(col),
    sex  = case_when(
      str_detect(col, regex("\\bMale\\b",   TRUE)) ~ "Male",
      str_detect(col, regex("\\bFemale\\b", TRUE)) ~ "Female",
      TRUE ~ "Total"
    ),
    branch = case_when(
      str_detect(col, regex("Coast\\s*Guard", TRUE)) ~ "Coast Guard",
```

```
      str_detect(col, regex("Space\\s*Force", TRUE)) ~ "Space Force",
      str_detect(col, regex("Air\\s*Force",   TRUE)) ~ "Air Force",
      str_detect(col, regex("Marine|USMC",    TRUE)) ~ "Marine Corps",
      str_detect(col, regex("Navy",           TRUE)) ~ "Navy",
      str_detect(col, regex("Army",           TRUE)) ~ "Army",
      TRUE ~ NA_character_
    ),
    n_raw = stringr::str_squish(n_raw),
    n = suppressWarnings(
      readr::parse_number(n_raw, na = c("Male","Female","Total","-","-",""))
    )
  ) %>%
  filter(!is.na(branch), !is.na(n), n > 0) %>%
  mutate(n = as.integer(n)) %>%
  select(branch, sex, pay_grade, n)

# Scrape rank titles

ranks_raw <- read_html(ranks_url) %>%
  html_elements("table") %>%
  html_table(fill = TRUE) %>%
  bind_rows()
```

```
New names:
* `` -> `...1`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...3`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...4`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...5`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...6`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...7`
* `Ranks by Branch of Service` -> `Ranks by Branch of Service...8`
```

```
ranks <- ranks_raw %>%
  rename(pay_grade = 1) %>%
  mutate(across(everything(), ~ .x |> as.character() |> stringr::str_squish())) %>%
  select(where(~ any(!is.na(.)))) %>%                              # drop fully-empty cols
  filter(stringr::str_detect(pay_grade, "^(?i)[EWO][- ]?\\d+")) %>%
  mutate(
    pay_grade = pay_grade |> stringr::str_to_upper() |> stringr::str_replace_all("[-\\s]", "
  ) %>%
  pivot_longer(-pay_grade, names_to = "branch_raw", values_to = "rank_title") %>%
```

```
  mutate(
    branch = case_when(                                          # normalize branches
      str_detect(branch_raw, regex("Coast\\s*Guard", TRUE)) ~ "Coast Guard",
      str_detect(branch_raw, regex("Space\\s*Force", TRUE)) ~ "Space Force",
      str_detect(branch_raw, regex("Air\\s*Force",   TRUE)) ~ "Air Force",
      str_detect(branch_raw, regex("Marine|USMC",    TRUE)) ~ "Marine Corps",
      str_detect(branch_raw, regex("Navy",           TRUE)) ~ "Navy",
      str_detect(branch_raw, regex("Army",           TRUE)) ~ "Army",
      TRUE ~ NA_character_
    ),
    rank_title = na_if(stringr::str_squish(rank_title), "")
  ) %>%
  filter(!is.na(branch), !is.na(rank_title)) %>%
  select(pay_grade, branch, rank_title)


# Join groups

df_groups <- groups %>%
  left_join(ranks, by = c("pay_grade","branch")) %>%
  relocate(rank_title, .after = pay_grade)

# Build individuals table

df_individuals <- df_groups %>% uncount(n, .remove = TRUE)
```

#3, Boxplot Volume

```
volume_fn <- function(L = 36, W = 48) {
  function(h) (L - 2*h) * (W - 2*h) * h
}

volume_box <- function(L = 36, W = 48) {
  V <- function(h) (L - 2*h) * (W - 2*h) * h
  Hfinal <- optimize(V, interval = c(0, min(L, W)/2), maximum = TRUE)
  list(Max_height = Hfinal$maximum, Max_volume = Hfinal$objective)
}
volume_box()
```

```
$Max_height
```

```
[1] 6.788902

$Max_volume
[1] 5239.819
```