STAA 551 Assignment 1

Matthew Stoebe

2024-08-29

# Assignment Description

Use R Markdown to complete your assignment.

All problem numbers refer to Regression and Other Stories by Gelman, Hill, and Vehtari Submission is a PDF file upload. There is only one attempt for this assignment, so make sure you upload the correct file

* Chapter 2: 2.3, 2.9
* Chapter 3: 3.6–3.9

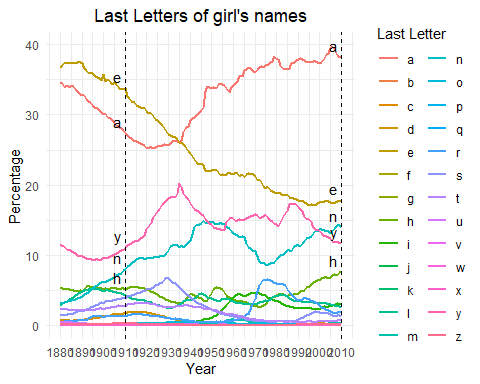
# Question 2.3

allnames <- read.csv("./ROS-Examples-master/Names/data/allnames\_clean.csv")  
  
female\_data <- filter(allnames, sex == "F")  
  
female\_data$last\_letter <- substr(female\_data$name, nchar(female\_data$name), nchar(female\_data$name))  
  
grouped\_data <- group\_by(female\_data, last\_letter)  
  
  
summarized\_data <- summarise(grouped\_data, across(matches("^X\\d"), sum, na.rm = TRUE))

## Warning: There was 1 warning in `summarise()`.  
## ℹ In argument: `across(matches("^X\\d"), sum, na.rm = TRUE)`.  
## ℹ In group 1: `last\_letter = "a"`.  
## Caused by warning:  
## ! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.  
## Supply arguments directly to `.fns` through an anonymous function instead.  
##   
## # Previously  
## across(a:b, mean, na.rm = TRUE)  
##   
## # Now  
## across(a:b, \(x) mean(x, na.rm = TRUE))

total\_sums <- colSums(summarized\_data[, -1]) # Exclude the first column (last\_letter)  
  
percentage\_data <- summarized\_data  
  
for (col in names(summarized\_data[-1])) {  
 percentage\_data[[col]] <- 100 \* (percentage\_data[[col]] / total\_sums[col])  
}  
  
  
percentage\_long <- pivot\_longer(percentage\_data, cols = starts\_with("X"),   
 names\_to = "year",   
 values\_to = "percentage")  
percentage\_long <- percentage\_long %>%  
 mutate(year = as.numeric(sub("^X", "", year)))  
  
  
top\_labels\_2010 <- percentage\_long %>%  
 filter(year == 2010) %>%  
 arrange(desc(percentage)) %>%  
 slice\_head(n = 5) # Get the top 3 lines based on percentage  
  
top\_labels\_1910 <- percentage\_long %>%  
 filter(year == 1910) %>%  
 arrange(desc(percentage)) %>%  
 slice\_head(n = 5) # Get the top 3 lines based on percentage  
  
  
  
ggplot(percentage\_long, aes(x = year, y = percentage, color = last\_letter, group = last\_letter)) +  
 geom\_line(size = 1) +  
 labs(title = "Last Letters of girl's names",  
 x = "Year",  
 y = "Percentage",  
 color = "Last Letter") +  
 scale\_x\_continuous(breaks = seq(1880, 2010, by = 10)) + # Show labels for every 10 years  
 theme\_minimal() +  
 theme(plot.title = element\_text(hjust = 0.5)) +  
  
 geom\_text(data = top\_labels\_2010,  
 aes(label = last\_letter),  
 vjust = -0.5, hjust = 1.5, color = "black") +  
   
 # Optional: Add custom annotations or vertical lines as needed  
 geom\_vline(xintercept = 2010, linetype = "dashed", color = "black") +  
  
 geom\_text(data = top\_labels\_1910,  
 aes(label = last\_letter),  
 vjust = -0.5, hjust = 1.5, color = "black") +  
   
 # Optional: Add custom annotations or vertical lines as needed  
 geom\_vline(xintercept = 1910, linetype = "dashed", color = "black")

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.  
## ℹ Please use `linewidth` instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.



# Question 2.9

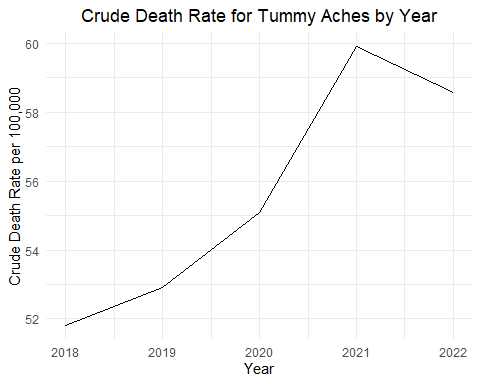
Graphing parallel time series: The mortality data in Section 2.4 are accessible from this site at the U.S. Centers for Disease Control and Prevention: wonder.cdc.gov. Download mortality data from this source but choose just one particular cause of death, and then make graphs similar to those in Section 2.4, breaking down trends in death rate by age, sex, and region of the country

data\_path <- "./Other Data/Mortality/Underlying\_Cause\_of\_Death\_2018\_2022\_Single\_Race.csv"  
  
# Initial Data Prep  
#data <- read.delim("C:\\Users\\mstoebe\\Downloads\\Underlying Cause of Death, 2018-2022, Single Race (1).txt", sep = "\t", header = TRUE)  
#filtered\_data <- data[!is.na(data$`Year.Code`) & data$`Year.Code` != "", ]  
#write.csv(filtered\_data, data\_path)  
  
data <- read.csv(data\_path)  
  
data$Crude.Rate <- as.numeric(data$Crude.Rate)

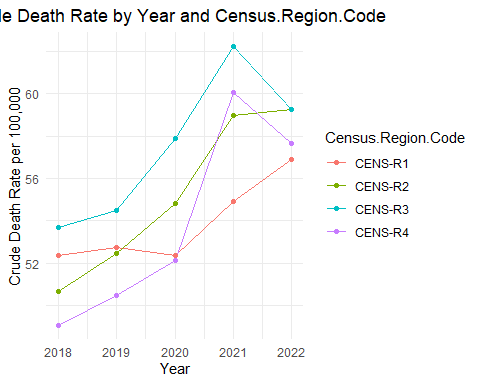
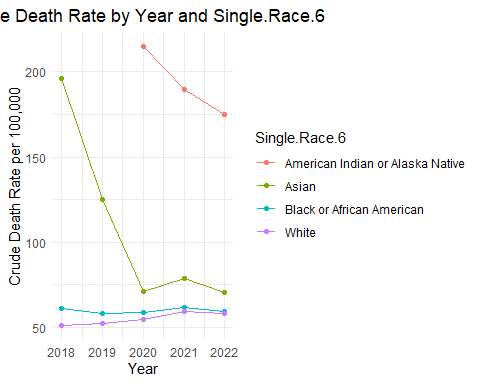
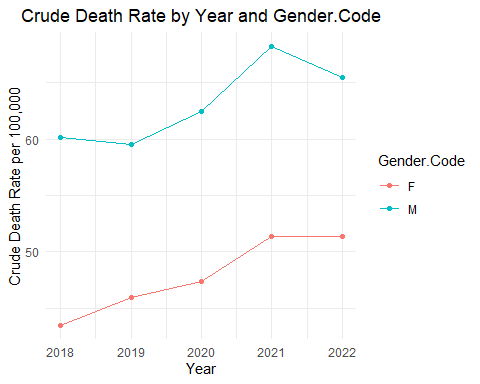
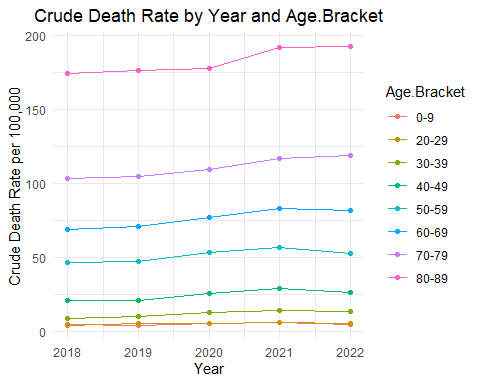
## Warning: NAs introduced by coercion

filtered\_data <- data[!is.na(data$Crude.Rate), ]  
  
# Convert Population and Deaths to numeric, setting non-numeric values to NA  
filtered\_data$Population <- as.numeric(gsub(",", "", filtered\_data$Population))  
filtered\_data$Deaths <- as.numeric(gsub(",", "", filtered\_data$Deaths))  
filtered\_data$Single.Year.Ages <- as.numeric(gsub("[^0-9]", "", filtered\_data$`Single.Year.Ages`))  
filtered\_data$Age.Bracket <- floor(filtered\_data$Single.Year.Ages / 10) \* 10  
filtered\_data$Age.Bracket <- factor(filtered\_data$Age.Bracket,   
 levels = seq(0, 90, by = 10),   
 labels = paste(seq(0, 90, by = 10), seq(9, 99, by = 10), sep = "-"))  
  
  
summary\_by\_year <- aggregate(cbind(Deaths, Population) ~ Year, data = filtered\_data, sum, na.rm = TRUE)  
summary\_by\_year$Crude.Rate <- (summary\_by\_year$Deaths / summary\_by\_year$Population) \* 100000  
  
  
ggplot(summary\_by\_year, aes\_string(x = "Year", y = "Crude.Rate")) +  
 geom\_line() +   
 labs(title = "Crude Death Rate for Tummy Aches by Year",  
 x = "Year",  
 y = "Crude Death Rate per 100,000") +  
 theme\_minimal() +  
 theme(plot.title = element\_text(hjust = 0.5))

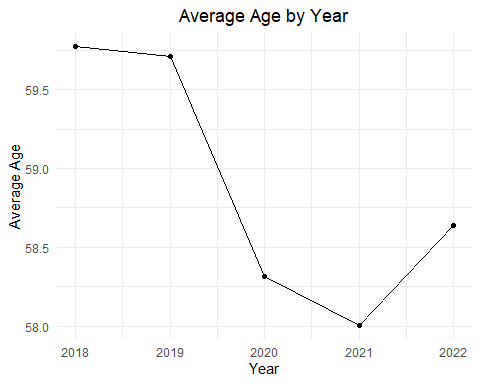
## Warning: `aes\_string()` was deprecated in ggplot2 3.0.0.  
## ℹ Please use tidy evaluation idioms with `aes()`.  
## ℹ See also `vignette("ggplot2-in-packages")` for more information.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.



features <- c("Age.Bracket","Gender.Code", "Single.Race.6", "Census.Region.Code" )  
  
  
for (feature in features) {  
 formula <- as.formula(paste("cbind(Deaths, Population) ~ Year +", feature))  
 summary\_by\_feature <- aggregate(formula, data = filtered\_data, sum, na.rm = TRUE)  
 summary\_by\_feature$Crude.Rate <- (summary\_by\_feature$Deaths / summary\_by\_feature$Population) \* 100000  
  
 p <- ggplot(summary\_by\_feature, aes\_string(x = "Year", y = "Crude.Rate", color = feature)) +  
 geom\_line() +  
 geom\_point() +  
 labs(title = paste("Crude Death Rate by Year and", feature),  
 x = "Year",  
 y = "Crude Death Rate per 100,000",  
 color = feature) +  
 theme\_minimal() +  
 theme(plot.title = element\_text(hjust = 0.5))  
  
   
 # Print the plot  
 print(p)   
  
}



average\_age\_by\_year <- aggregate(Single.Year.Ages ~ Year, data = filtered\_data, mean, na.rm = TRUE)  
ggplot(average\_age\_by\_year, aes(x = Year, y = Single.Year.Ages)) +  
 geom\_line() +   
 geom\_point() +  
 labs(title = "Average Age by Year",  
 x = "Year",  
 y = "Average Age") +  
 theme\_minimal() +  
 theme(plot.title = element\_text(hjust = 0.5))



#Question 3.6

Linear transformations: A test is graded from 0 to 50, with an average score of 35 and a standard deviation of 10. For comparison to other tests, it would be convenient to rescale to a mean of 100 and standard deviation of 15. (a) Labeling the original test scores as x and the desired rescaled test score as y, come up with a linear transformation, that is, values of a and b so that the rescaled scores y = a + bx have a mean of 100 and a standard deviation of 15. (b) What is the range of possible values of this rescaled score y? (c) Plot the line showing y vs. x

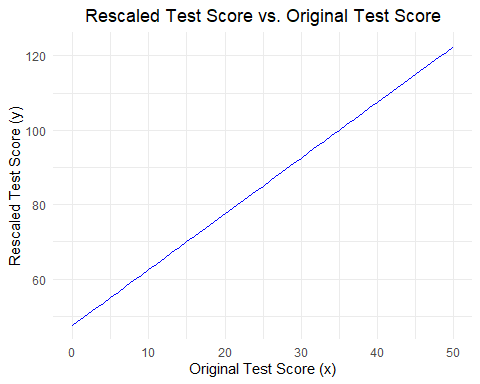
# Solution (a): Determine the linear transformation  
x\_mean <- 35  
x\_stdev <- 10  
y\_mean <- 100  
y\_stdev <- 15  
  
  
b <- y\_stdev / x\_stdev  
a <- y\_mean - b \* x\_mean  
  
  
cat("The linear transformation is: y =", a, "+", b, "\* x\n")

## The linear transformation is: y = 47.5 + 1.5 \* x

# Solution (b): Determine the range of the new test scores  
x\_min <- 0  
x\_max <- 50  
  
y\_min <- a + b \* x\_min  
y\_max <- a + b \* x\_max  
  
cat("The range of the rescaled test scores is from", y\_min, "to", y\_max, "\n")

## The range of the rescaled test scores is from 47.5 to 122.5

# Solution (c): Plot y vs. x  
data <- data.frame(x = seq(0, 50, by = 1))  
  
# Apply the linear transformation to get y values  
data$y <- a + b \* data$x  
  
# Plot y vs. x with centered title  
ggplot(data, aes(x = x, y = y)) +  
 geom\_line(color = "blue") +  
 labs(title = "Rescaled Test Score vs. Original Test Score",  
 x = "Original Test Score (x)",  
 y = "Rescaled Test Score (y)") +  
 theme\_minimal() +  
 theme(plot.title = element\_text(hjust = 0.5))



#Question 3.7

## Solution

You could also use -1.5 for your slope because the equation for slope is y\_stdev = abs(b) \* x\_stdev

In 3.6, we assumed that the transformation would be postive, but it is also possible to do a negative transformation. The primary issue with this is that the interpretation of the scores changes as the lowest test scores now become the highest and the highest test scores become the lowest.

#Question 3.8

We know the correlation means and standard deviation which means that we can get the summed mean and standard deviation.

The new mean = (x\_mean + y\_mean) /2 = (69.1 + 63.7) /2 = 66.4

New STDEV =

mean\_x <- 69.1 # Mean height of husbands  
mean\_y <- 63.7 # Mean height of wives  
sd\_x <- 2.9 # Standard deviation of husbands' heights  
sd\_y <- 2.7 # Standard deviation of wives' heights  
rho <- 0.3 # Correlation between husbands' and wives' heights  
  
# Mean of the average height  
new\_mean <- (mean\_x + mean\_y) / 2  
  
var <- sd\_x^2 + sd\_y^2 + 2\*rho\*sd\_x\*sd\_y  
  
new\_sd = sqrt(var)  
  
cat("Average height for husbands and wives:", new\_mean, "inches\n")

## Average height for husbands and wives: 66.4 inches

cat("Standard deviation of the average height for husbands and wives:", new\_sd, "inches\n")

## Standard deviation of the average height for husbands and wives: 4.516415 inches

#Question 3.9

Answered using the study below on how price controlls affected price volitility <https://link.springer.com/article/10.1007/s00181-020-01953-w>

# Load the necessary library  
library(ggplot2)  
  
# Define parameters  
control\_mean <- 100  
treated\_mean <- 100  
control\_sd <- 10  
treated\_sd <- control\_sd \* 0.6 # 60% reduction in variability for liberalized prices  
inflation\_impact <- 0  
  
# Adjust control SD for inflation  
control\_sd\_inflation <- control\_sd + inflation\_impact  
  
# Generate sequences of prices  
x <- seq(50, 150, length.out = 100)  
  
# Calculate the normal distribution for both groups  
control\_density <- dnorm(x, mean = control\_mean, sd = control\_sd\_inflation)  
treated\_density <- dnorm(x, mean = treated\_mean, sd = treated\_sd)  
  
# Create a data frame for plotting  
data <- data.frame(  
 x = c(x, x),  
 density = c(control\_density, treated\_density),  
 group = factor(rep(c("Price Control", "Liberalized Prices"), each = 100))  
)  
  
# Plot the distributions  
ggplot(data, aes(x = x, y = density, color = group)) +  
 geom\_line(size = 1) +  
 labs(title = "Impact of Price Controls on Price Variability",  
 x = "Price",  
 y = "Density") +  
 theme\_minimal() +  
 theme(plot.title = element\_text(hjust = 0.5))

