Midterm: Flood -exploratory data analysis

Ruicheng Zhang

2023-12-01

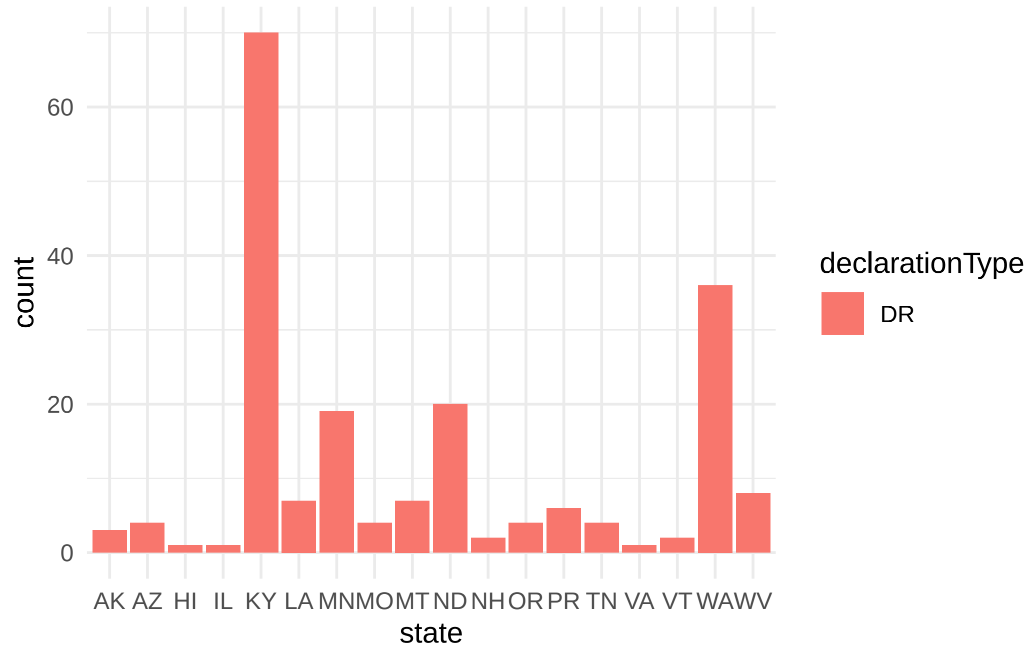
Initial questions

Using which data? What direction do we need to take our research, is there any relationship between the variables, and any meaningful output.

Data acquisition and assessment

```
# Read the data
disaster_data <- read.csv('~/Downloads/615mid/DisasterDeclarationsSummaries.csv')
```

```
# Filter rows where incidentType is 'Flood'
flood_data <- subset(disaster_data, incidentType == 'Flood')

# Drop unnecessary columns
columns_to_drop <- c('femaDeclarationString', 'declarationRequestNumber', 'ihProgramDeclar
                     'paProgramDeclared', 'hmProgramDeclared', 'tribalRequest', 'hash', 'i
                     'lastRefresh', 'disasterCloseOutDate',
                     'fipsStateCode', 'fipsCountyCode','incidentType')
flood_data_clean <- flood_data[, !(names(flood_data) %in% columns_to_drop)]

# Plot the number of flood disaster declarations by state

flood_data_clean %>%
 filter(incidentBeginDate >= "2020-01-25" & incidentBeginDate <= "2022-11-19") %>%
 ggplot() +
 aes(x = state, fill = declarationType) +
 geom_bar() +
 scale_fill_hue(direction = 1) +
 theme_minimal()
```



data analysis

#Find the time interval between each flood

```
# Convert incidentBeginDate and incidentEndDate to Date objects
flood_data_clean$incidentBeginDate <- as.Date(flood_data_clean$incidentBeginDate, format="
flood_data_clean$incidentEndDate <- as.Date(flood_data_clean$incidentEndDate, format="%Y-%

# Calculate the difference in months
flood_data_clean$durationInMonths <- NA
valid_dates <- !is.na(flood_data_clean$incidentBeginDate) & !is.na(flood_data_clean$incide
flood_data_clean$durationInMonths[valid_dates] <- round(as.numeric(difftime(flood_data_cle
                                                                          flood_data_clean$
                                                                          units = "weeks"))

# Display the first few rows of the dataframe
head(flood_data_clean[, c('incidentBeginDate', 'incidentEndDate', 'durationInMonths')])
```

```
    incidentBeginDate incidentEndDate durationInMonths
6          2023-06-08      2023-06-23             0.49
10         2023-06-08      2023-06-23             0.49
16         2023-06-08      2023-06-23             0.49
```

```
17          2023-06-08       2023-06-23                0.49
233         2023-08-03       2023-08-05                0.07
333         2023-07-07       2023-07-21                0.46
```

the same with declaration

```
# Convert declarationDate and disasterCloseoutDate to Date objects
flood_data_clean$declarationDate <- as.Date(flood_data_clean$declarationDate, format="%Y-%
flood_data_clean$disasterCloseoutDate <- as.Date(flood_data_clean$disasterCloseoutDate, fo

# Calculate the difference in months
flood_data_clean$declarationInMonths <- NA
valid_dates <- !is.na(flood_data_clean$declarationDate) & !is.na(flood_data_clean$disaster
flood_data_clean$declarationInMonths[valid_dates] <- round(as.numeric(difftime(flood_data_
                                                                       flood_data_clean$
                                                                       units = "weeks"))

# Display the first few rows of the dataframe
head(flood_data_clean[, c('declarationDate', 'disasterCloseoutDate', 'declarationInMonths'
```

```
    declarationDate disasterCloseoutDate declarationInMonths
6        2023-08-25                 <NA>                  NA
10       2023-08-25                 <NA>                  NA
16       2023-08-25                 <NA>                  NA
17       2023-08-25                 <NA>                  NA
233      2023-10-06                 <NA>                  NA
333      2023-07-14                 <NA>                  NA
```

#Calculate Waiting Time

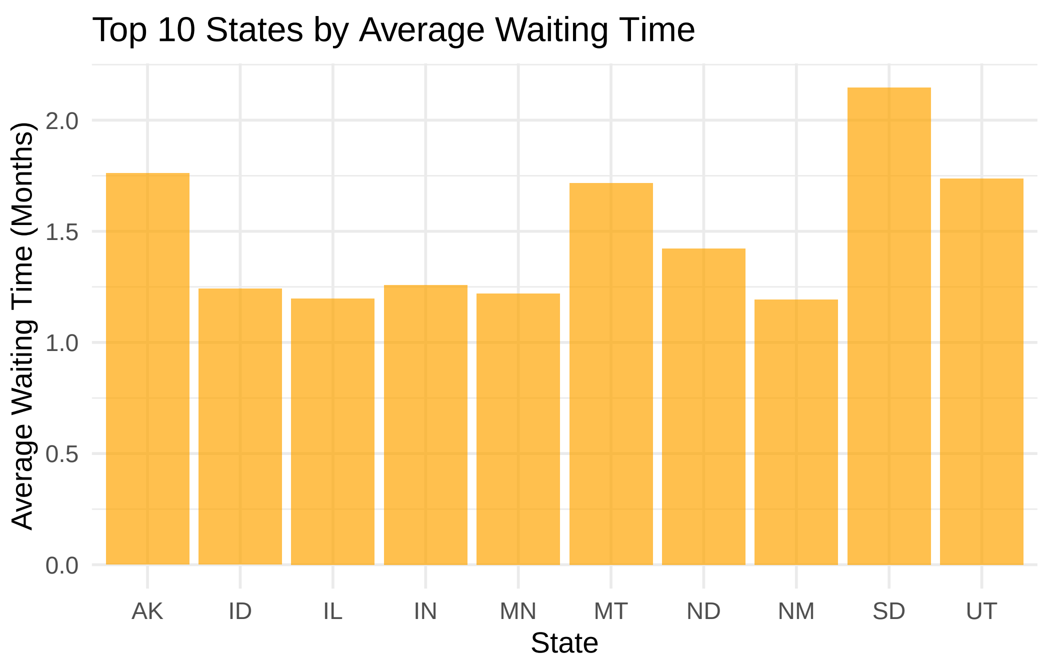
```
# Calculate the difference in months
flood_data_clean$waitingTime <- NA
valid_dates <- !is.na(flood_data_clean$incidentBeginDate) & !is.na(flood_data_clean$declar
flood_data_clean$waitingTime[valid_dates] <- round(as.numeric(difftime(flood_data_clean$de
                                                                       flood_data_clean$
                                                                       units = "weeks"))

# Display the first few rows of the dataframe
head(flood_data_clean[, c('incidentBeginDate', 'declarationDate', 'waitingTime')])
```

```
    incidentBeginDate declarationDate waitingTime
6          2023-06-08      2023-08-25        2.56
10         2023-06-08      2023-08-25        2.56
16         2023-06-08      2023-08-25        2.56
17         2023-06-08      2023-08-25        2.56
233        2023-08-03      2023-10-06        2.10
333        2023-07-07      2023-07-14        0.23
```

```r
columnr <- c('declarationDate','disasterCloseoutDate','incidentEndDate')

# Remove the column
data <- flood_data_clean[, !(names(flood_data_clean) %in% columnr)]
```

#Delay between disaster onset and declaration in top ten state

```r
# Calculate the average waiting time by state
avg_waiting_time_by_state <- data %>%
  group_by(state) %>%
  summarise(AvgWaitingTime = mean(waitingTime, na.rm = TRUE)) %>%
  arrange(desc(AvgWaitingTime))

# Plot the average waiting time for the top 10 states
top_states <- head(avg_waiting_time_by_state, 10)
ggplot(top_states, aes(x = state, y = AvgWaitingTime)) +
  geom_bar(stat = 'identity', fill = 'orange', alpha = 0.7) +
  theme_minimal() +
  labs(title = 'Top 10 States by Average Waiting Time',
       x = 'State',
       y = 'Average Waiting Time (Months)')
```

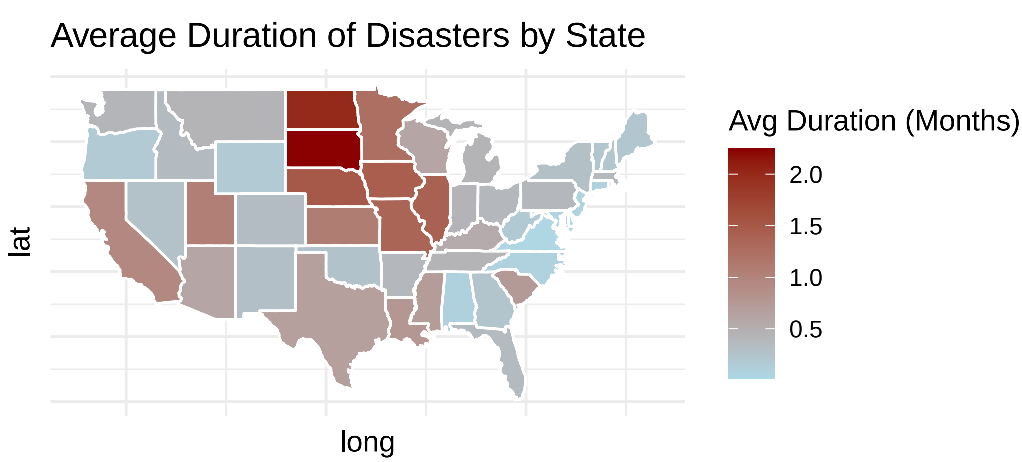



```
# Define a mapping from state abbreviations to full names
state_mapping <- data.frame(
  abbreviation = c('AL', 'AK', 'AZ', 'AR', 'CA', 'CO', 'CT', 'DE', 'FL', 'GA',
                   'HI', 'ID', 'IL', 'IN', 'IA', 'KS', 'KY', 'LA', 'ME', 'MD',
                   'MA', 'MI', 'MN', 'MS', 'MO', 'MT', 'NE', 'NV', 'NH', 'NJ',
                   'NM', 'NY', 'NC', 'ND', 'OH', 'OK', 'OR', 'PA', 'RI', 'SC',
                   'SD', 'TN', 'TX', 'UT', 'VT', 'VA', 'WA', 'WV', 'WI', 'WY'),
  full_name = c('Alabama', 'Alaska', 'Arizona', 'Arkansas', 'California', 'Colorado', 'Con
                'Delaware', 'Florida', 'Georgia', 'Hawaii', 'Idaho', 'Illinois', 'Indiana'
                'Iowa', 'Kansas', 'Kentucky', 'Louisiana', 'Maine', 'Maryland', 'Massachus
                'Michigan', 'Minnesota', 'Mississippi', 'Missouri', 'Montana', 'Nebraska',
                'Nevada', 'New Hampshire', 'New Jersey', 'New Mexico', 'New York', 'North
                'North Dakota', 'Ohio', 'Oklahoma', 'Oregon', 'Pennsylvania', 'Rhode Islan
                'South Carolina', 'South Dakota', 'Tennessee', 'Texas', 'Utah', 'Vermont',
                'Virginia', 'Washington', 'West Virginia', 'Wisconsin', 'Wyoming')
)

# Replace state abbreviations with full names
data$state <- state_mapping$full_name[match(data$state, state_mapping$abbreviation)]

# Calculate the average durationInMonths by state
avg_duration_by_state <- data %>%
```

```
  group_by(state) %>%
  summarise(AvgDuration = mean(durationInMonths, na.rm = TRUE))
```

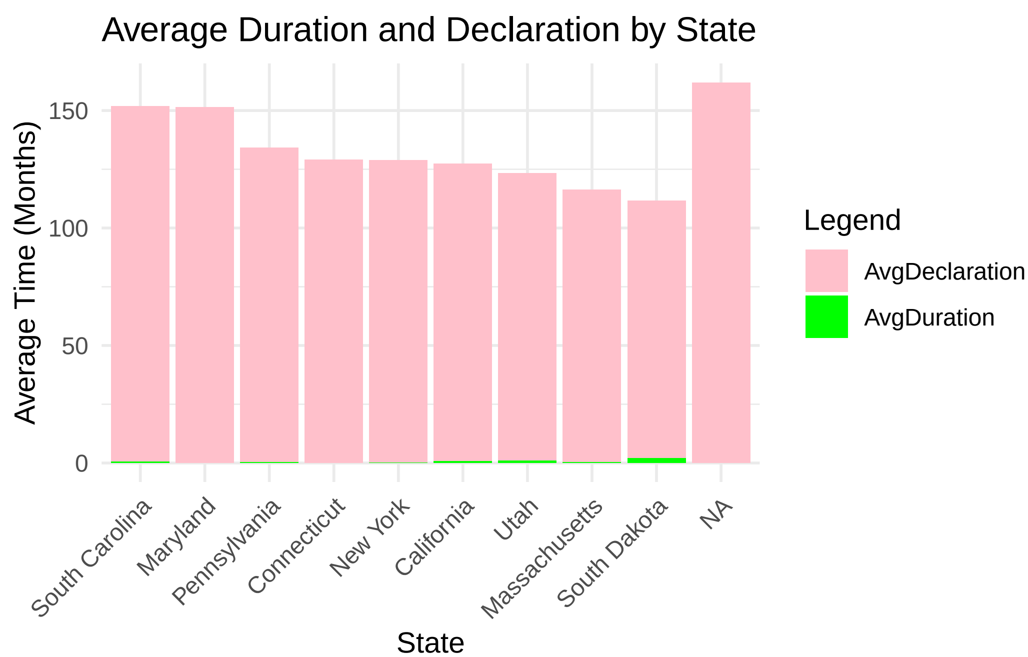
find the average duration of disasters for each state

```
# Create a map of the USA
usa_map <- map_data("state")
usa_map$state <- tools::toTitleCase(usa_map$region)

# Merge the map data with the average duration data
usa_map <- left_join(usa_map, avg_duration_by_state, by= "state")

# Plot the map
ggplot(data = usa_map, aes(x = long, y = lat, group = group, fill = AvgDuration)) +
  geom_polygon(color = "white") +
  coord_fixed(1.3) +
  scale_fill_gradient(low = "lightblue", high = "darkred", na.value = "grey90") +
  theme_minimal() +
  labs(title = "Average Duration of Disasters by State",
       fill = "Avg Duration (Months)") +
  theme(axis.text = element_blank(),
        axis.ticks = element_blank())
```

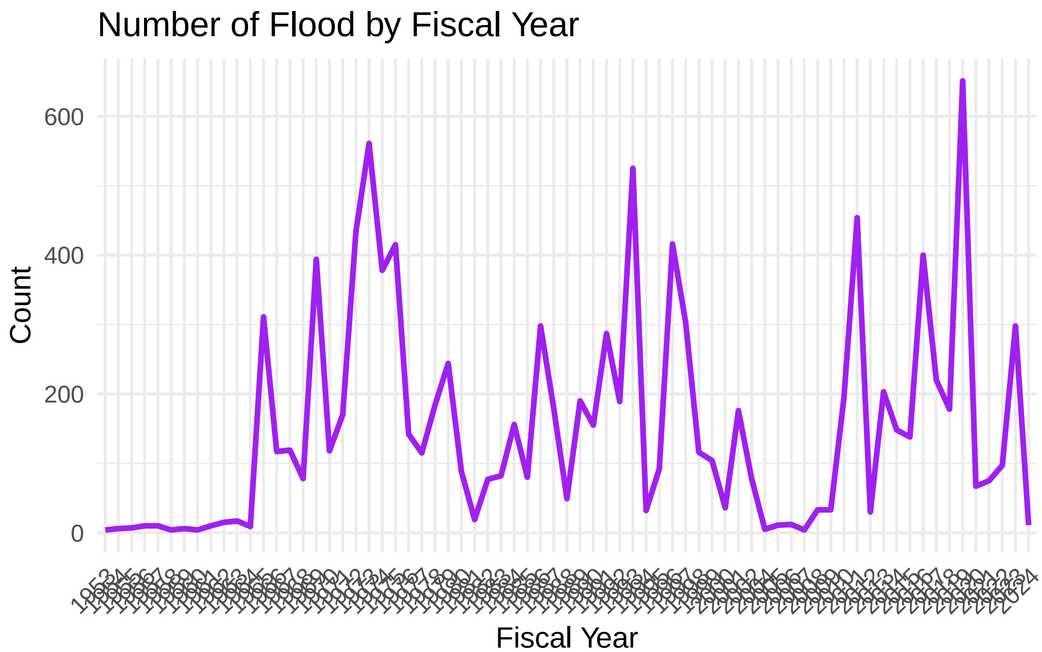



```
# Calculate the average durationInMonths and declarationInMonths by state
avg_duration_declaration_by_state <- data %>%
```

```r
  group_by(state) %>%
  summarise(AvgDuration = mean(durationInMonths, na.rm = TRUE),
            AvgDeclaration = mean(declarationInMonths, na.rm = TRUE))

# Sort the states by the sum of average durationInMonths and declarationInMonths
top_states <- avg_duration_declaration_by_state %>%
  mutate(Total = AvgDuration + AvgDeclaration) %>%
  arrange(desc(Total)) %>%
  head(10)

# Convert data to long format
top_states_long <- top_states %>%
  select(state, AvgDuration, AvgDeclaration) %>%
  gather(key = 'Type', value = 'Value', -state)

# Plot the stacked bar chart
ggplot(data = top_states_long, aes(x = reorder(state, -Value), y = Value, fill = Type)) +
  geom_bar(stat = 'identity', position = 'stack') +
  scale_fill_manual(values = c('pink', 'green')) +
  theme_minimal() +
  labs(title = 'Average Duration and Declaration by State',
       x = 'State',
       y = 'Average Time (Months)',
       fill = 'Legend') +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

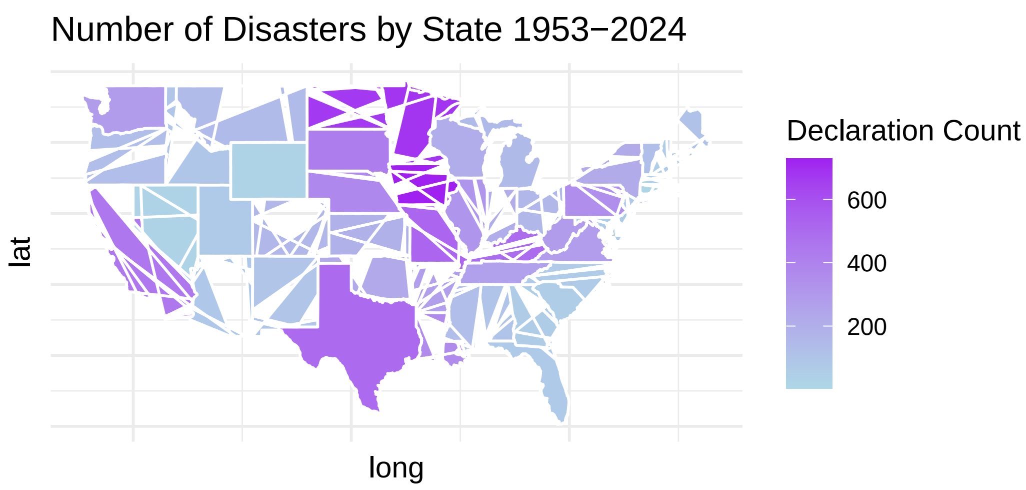



```r
ggplot(data, aes(x = as.factor(fyDeclared))) +
  geom_line(stat = 'count', group = 1, color = 'purple', linewidth = 1) +
  theme_minimal() +
  labs(title = 'Number of Flood by Fiscal Year',
       x = 'Fiscal Year',
       y = 'Count') +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Looking into the number of disasters in past 70 years

```
declaration_count_by_state <- data %>%
  group_by(state) %>%
  summarise(Count = n())

um <- merge(usa_map, declaration_count_by_state,by= "state")


ggplot(data = um, aes(x = long, y = lat, group = group, fill = Count)) +
  geom_polygon(color = "white") +
  coord_fixed(1.3) +
  scale_fill_gradient(low = "lightblue", high = "purple", na.value = "grey90") +
  theme_minimal() +
  labs(title = "Number of Disasters by State 1953-2024",
       fill = "Declaration Count") +
  theme(axis.text = element_blank(),
        axis.ticks = element_blank())
```




```
data$placeCode <- paste0(flood_data$fipsStateCode,flood_data$fipsCountyCode)

data$placeCode <- sprintf("%05s", data$placeCode)
```

Reading census data

```
acs202 <- read.csv('~/Downloads/615mid/Census Download_2023-10-23T140133/ACSDP5Y2020.DP05-
acs202$placeCode <- substr(acs202$Geography, nchar(acs202$Geography)-4, nchar(acs202$Geogr
acs202<- as.data.frame(acs202)
acs0 <- acs202 %>% select(placeCode,Total)

data20 <- subset(data, fyDeclared == '2020')
data20 <- data20 %>% left_join(acs0 %>% select(placeCode, Total), by = "placeCode")
```

```
acs212 <- read.csv('~/Downloads/615mid/Census Download_2023-10-23T140133/ACSDP5Y2021.DP05-
acs212$placeCode <- substr(acs212$Geography, nchar(acs212$Geography)-4, nchar(acs212$Geogr
acs212<- as.data.frame(acs212)
acs1 <- acs212 %>% select(placeCode,Total)

data21 <- subset(data, fyDeclared == '2021')
data21 <- data21 %>% left_join(acs1 %>% select(placeCode, Total), by = "placeCode")
```

cleaning the census data into 2020 2021

```
result <- data20 %>%
  group_by(state) %>%
  summarise(AvgDurationInMonths = mean(durationInMonths, na.rm = TRUE),
            TotalSum = sum(Total, na.rm = TRUE))
head(result)
```

```
# A tibble: 5 x 3
  state        AvgDurationInMonths TotalSum
  <chr>                      <dbl>    <int>
1 North Dakota               0.681     7778
2 Oregon                     0.13         0
3 Texas                      0.2    5722456
4 Washington                 0.69         0
5 Wisconsin                  0.07    195859
```

```
result1 <- data21 %>%
  group_by(state) %>%
  summarise(AvgDurationInMonths = mean(durationInMonths, na.rm = TRUE),
            TotalSum = sum(Total, na.rm = TRUE))
head(result1)
```

```
# A tibble: 6 x 3
  state     AvgDurationInMonths TotalSum
  <chr>                   <dbl>    <int>
1 Arizona                  0.07        0
2 Hawaii                   0.33        0
3 Kentucky                 0.49   877641
4 Louisiana                0.13        0
5 Tennessee                0           0
6 Vermont                  0.03     7632
```

#reading the storm data

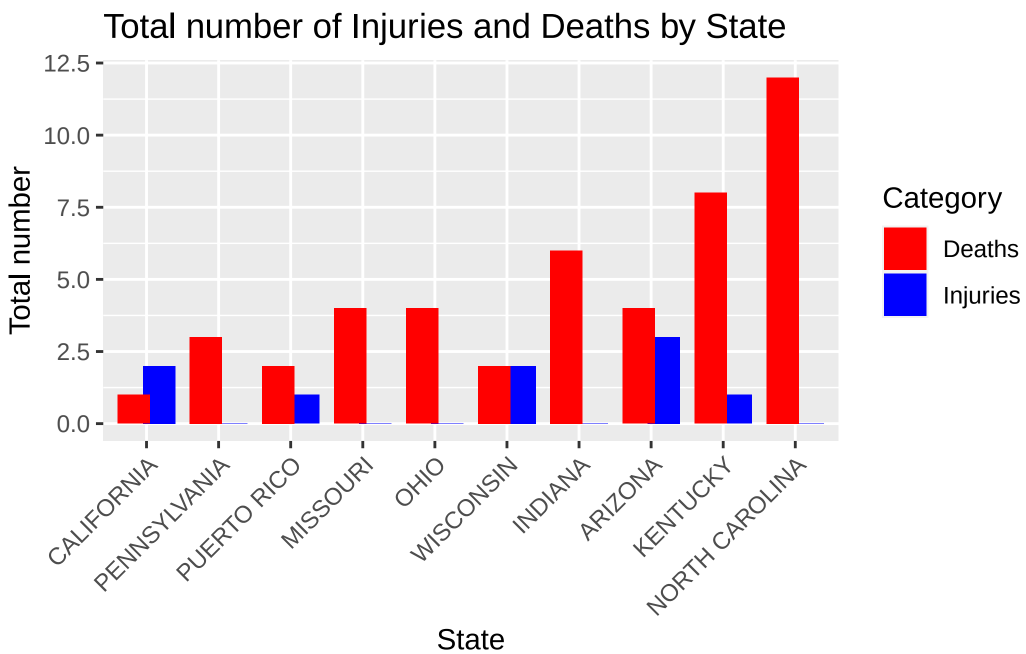
```
storm_data <- read.csv("~/Downloads/615mid/storm/2020/StormEvents_details-ftp_v1.0_d2020_c

# Select the specified columns
storm_data_selected <- storm_data %>%
```

```
  select(STATE, EVENT_TYPE, INJURIES_DIRECT, INJURIES_INDIRECT, DEATHS_DIRECT, DEATHS_INDI

# Filter rows where EVENT_TYPE contains 'Flood'
storm_data <- storm_data_selected %>%
  filter(grepl("Flood", EVENT_TYPE))

storm_data <- storm_data %>%
  mutate(Total_Injuries = INJURIES_DIRECT + INJURIES_INDIRECT) %>%
  select(-INJURIES_DIRECT, -INJURIES_INDIRECT)

storm_data <- storm_data %>%
  mutate(Total_DEATHS = DEATHS_DIRECT + DEATHS_INDIRECT) %>%
  select(-DEATHS_DIRECT, -DEATHS_INDIRECT)

storm_data <- storm_data %>%
  mutate(
    DAMAGE_PROPERTY = as.numeric(gsub("K", "", DAMAGE_PROPERTY, ignore.case = TRUE)) / ife
    DAMAGE_PROPERTY = as.numeric(gsub("M", "", DAMAGE_PROPERTY, ignore.case = TRUE)) * ife
    DAMAGE_CROPS = as.numeric(gsub("K", "", DAMAGE_CROPS, ignore.case = TRUE)) / ifelse(gr
    DAMAGE_CROPS = as.numeric(gsub("M", "", DAMAGE_CROPS, ignore.case = TRUE)) * ifelse(gr
  )

storm_data <- storm_data %>%
  mutate(Damage = DAMAGE_PROPERTY + DAMAGE_CROPS) %>%
  select(-DAMAGE_PROPERTY, -DAMAGE_CROPS)
```

Finding out the damage with each state

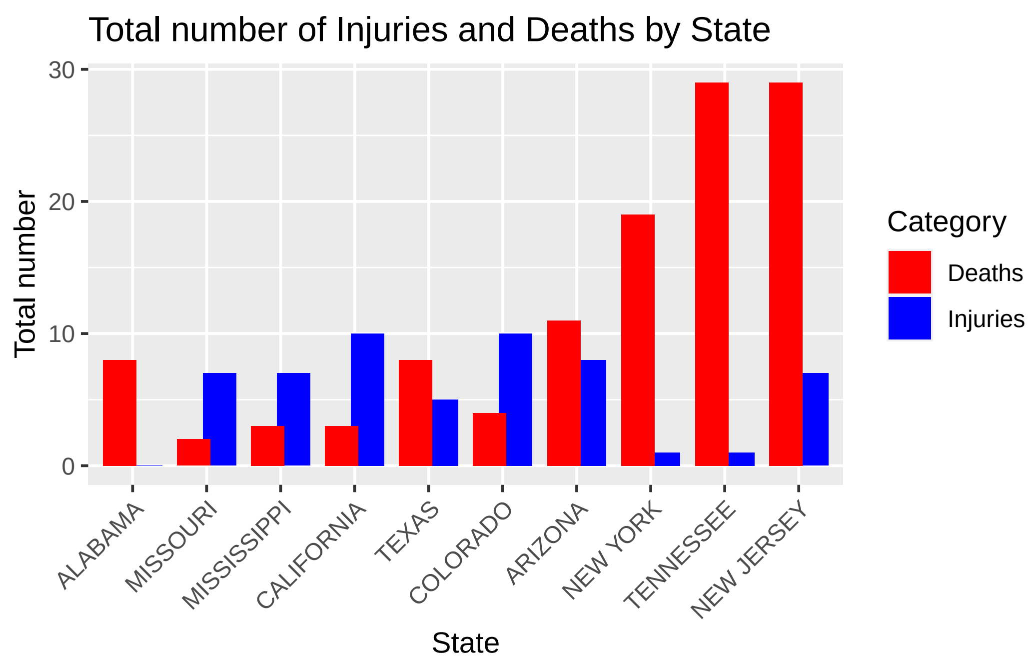
```
storm0 <- storm_data %>%
  group_by(STATE) %>%
  summarise(Injuries = sum(Total_Injuries, na.rm = TRUE),
            Deaths = sum(Total_DEATHS, na.rm = TRUE),
            Damage = sum(Damage, na.rm = TRUE))

storm00 <- storm0 %>%
  arrange(desc(Injuries + Deaths)) %>%
  slice(1:10)

df_long <- storm00 %>%
```

```r
  gather(key = "Category", value = "Value", -STATE,-Damage)

ggplot(df_long, aes(fill = Category, y = Value, x = reorder(STATE, Value))) +
  geom_bar(stat = "identity", position = position_dodge(width = 0.7)) +
  scale_fill_manual(values = c("Injuries" = "blue", "Deaths" = "red")) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(x = "State", y = "Total number", fill = "Category") +
  ggtitle("Total number of Injuries and Deaths by State")
```




```r
storm_data <- read.csv("~/Downloads/615mid/storm/2021/StormEvents_details-ftp_v1.0_d2021_c

# Select the specified columns
storm_data_selected <- storm_data %>%
  select(STATE, EVENT_TYPE, INJURIES_DIRECT, INJURIES_INDIRECT, DEATHS_DIRECT, DEATHS_INDI

# Filter rows where EVENT_TYPE contains 'Flood'
storm_data <- storm_data_selected %>%
  filter(grepl("Flood", EVENT_TYPE))

storm_data <- storm_data %>%
  mutate(Total_Injuries = INJURIES_DIRECT + INJURIES_INDIRECT) %>%
  select(-INJURIES_DIRECT, -INJURIES_INDIRECT)
```

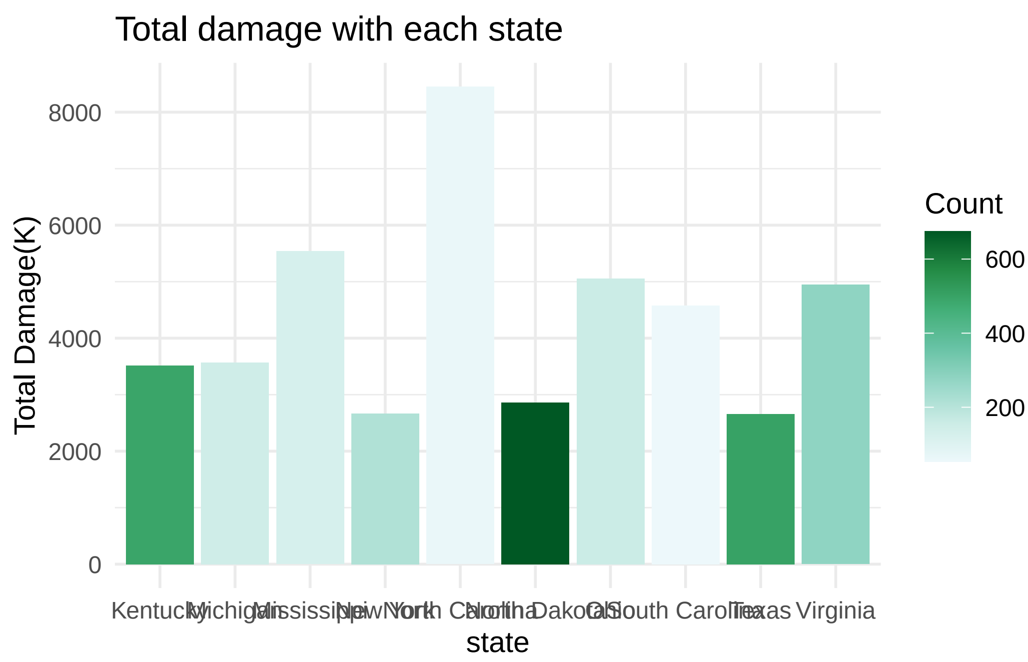
```r
storm_data <- storm_data %>%
  mutate(Total_DEATHS = DEATHS_DIRECT + DEATHS_INDIRECT) %>%
  select(-DEATHS_DIRECT, -DEATHS_INDIRECT)

storm_data <- storm_data %>%
  mutate(
    DAMAGE_PROPERTY = as.numeric(gsub("K", "", DAMAGE_PROPERTY, ignore.case = TRUE)) / ife
    DAMAGE_PROPERTY = as.numeric(gsub("M", "", DAMAGE_PROPERTY, ignore.case = TRUE)) * ife
    DAMAGE_CROPS = as.numeric(gsub("K", "", DAMAGE_CROPS, ignore.case = TRUE)) / ifelse(gr
    DAMAGE_CROPS = as.numeric(gsub("M", "", DAMAGE_CROPS, ignore.case = TRUE)) * ifelse(gr
  )

storm_data <- storm_data %>%
  mutate(Damage = DAMAGE_PROPERTY + DAMAGE_CROPS) %>%
  select(-DAMAGE_PROPERTY, -DAMAGE_CROPS)

storm1 <- storm_data %>%
  group_by(STATE) %>%
  summarise(Injuries = sum(Total_Injuries, na.rm = TRUE),
            Deaths = sum(Total_DEATHS, na.rm = TRUE),
            Damage = sum(Damage, na.rm = TRUE))

storm11 <- storm1 %>%
  arrange(desc(Injuries + Deaths)) %>%
  slice(1:10)

df_long <- storm11 %>%
  gather(key = "Category", value = "Value", -STATE,-Damage)

ggplot(df_long, aes(fill = Category, y = Value, x = reorder(STATE, Value))) +
  geom_bar(stat = "identity", position = position_dodge(width = 0.7)) +
  scale_fill_manual(values = c("Injuries" = "blue", "Deaths" = "red")) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(x = "State", y = "Total number", fill = "Category") +
  ggtitle("Total number of Injuries and Deaths by State")
```




```
capitalize_first <- function(string) {
  paste0(tolower(substr(string, 1, nchar(string))))
}

storm0$state<-sapply(storm0$STATE, capitalize_first)
storm0$state <- tools::toTitleCase(storm0$state)
storm0 <-storm0 %>%
     select(-STATE)

storm0 <- left_join(storm0,declaration_count_by_state, by='state')
```

```
s0 <- storm0 %>%
  arrange(desc(Damage)) %>%
  slice(1:10)

s0 %>%
 ggplot() +
 aes(x = state, fill = Count, weight = Damage) +
 geom_bar() +
 scale_fill_distiller(palette = "BuGn",
 direction = 1) +
 labs(y = "Total Damage(K)", title = "Total damage with each state") +
```

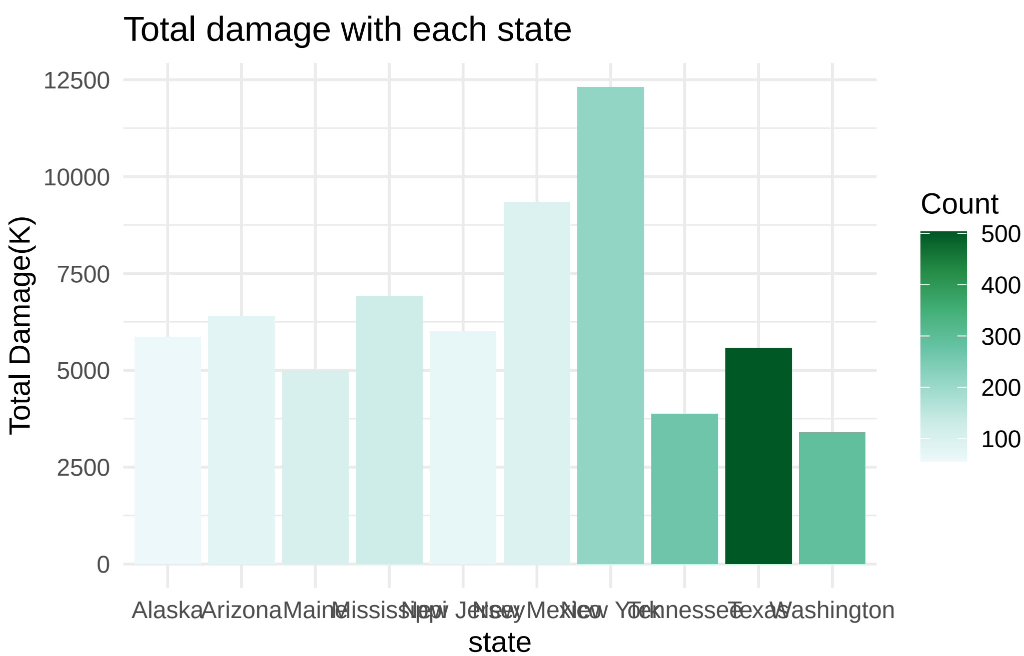
```
 theme_minimal()
```




```
capitalize_first <- function(string) {
  paste0(tolower(substr(string, 1, nchar(string))))
}

storm1$state<-sapply(storm1$STATE, capitalize_first)
storm1$state <- tools::toTitleCase(storm1$state)
storm1 <-storm1 %>%
     select(-STATE)

storm1 <- left_join(storm1,declaration_count_by_state, by='state')
```

```
s1 <- storm1 %>%
  arrange(desc(Damage)) %>%
  slice(1:10)
s1 %>%
 ggplot() +
 aes(x = state, fill = Count, weight = Damage) +
 geom_bar() +
 scale_fill_distiller(palette = "BuGn",
```

```
direction = 1) +
labs(y = "Total Damage(K)", title = "Total damage with each state") +
theme_minimal()
```



We can find that New York, Mexico, New Jersey, Texas, Tennessee and Mississipp are the most affected by flooding disasters, where the frequency of disasters, casualties, property damage The most affected by the flood disaster, in which the frequency of disasters, casualties, property damage is our main object of observation.

Combining Census, Flood, and Storm Data

```
com0 <- left_join(result,storm0, by='state')
com1 <- left_join(result1,storm1, by='state')
head(com0)
```

```
# A tibble: 5 x 7
  state        AvgDurationInMonths TotalSum Injuries Deaths Damage Count
  <chr>                      <dbl>    <int>    <int>  <int>  <dbl> <int>
1 North Dakota               0.681     7778        0      0   2860   674
2 Oregon                     0.13         0        0      1    879   127
```

```
3 Texas                        0.2    5722456        0       3  2657.   503
4 Washington                   0.69         0        0       0     0    291
5 Wisconsin                    0.07    195859        2       2  2328.   206
```

```
head(com1)
```

```
# A tibble: 6 x 7
  state      AvgDurationInMonths TotalSum Injuries Deaths Damage Count
  <chr>                    <dbl>    <int>    <int>  <int>  <dbl> <int>
1 Arizona                   0.07        0        8     11  6404.    82
2 Hawaii                    0.33        0        1      0     0     14
3 Kentucky                  0.49   877641        0      3  3044.   494
4 Louisiana                 0.13        0        0      5  1449    363
5 Tennessee                 0           0        1     29  3871.   268
6 Vermont                   0.03     7632        1      0  3255    124
```

Throughout the processing of the data, we can see that the more densely populated the area, the higher the casualty rate.

References

https://www.ncei.noaa.gov/pub/data/swdi/stormevents/csvfiles/Storm-Data-Export-Format.pdf https://www.fema.gov/about/openfema/data-sets