Mid term EDA

Yingmai Chen

2023-11-03

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.1     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- 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 become errors
```

```
library(ggplot2)
library(readxl)
library(stringr)
library(statebins)
library(RColorBrewer)
```

data clean for FEMA

```
disaster_summaries<-read.csv("DisasterDeclarationsSummaries.csv")
fema_summaries<-read.csv("FemaWebDisasterSummaries.csv")
flood_data <- disaster_summaries %>%
  filter(incidentType == "Flood", between(fyDeclared, 2020, 2021))
merged_data <- left_join(flood_data, fema_summaries, by = "disasterNumber")
glimpse(merged_data)
```

```
## Rows: 142
## Columns: 38
## $ femaDeclarationString      <chr> "DR-4621-VT", "DR-4621-VT", "DR-4620-AZ", "~
## $ disasterNumber             <int> 4621, 4621, 4620, 4620, 4620, 4609, 4609, 4~
## $ state                      <chr> "VT", "VT", "AZ", "AZ", "AZ", "TN", "TN", "~
## $ declarationType            <chr> "DR", "DR", "DR", "DR", "DR", "DR", "DR", "~
## $ declarationDate            <chr> "2021-09-29T00:00:00.000Z", "2021-09-29T00:~
## $ fyDeclared                 <int> 2021, 2021, 2021, 2021, 2021, 2021, 2021, 2~
## $ incidentType               <chr> "Flood", "Flood", "Flood", "Flood", "Flood"~
## $ declarationTitle           <chr> "SEVERE STORM AND FLOODING", "SEVERE STORM ~
```

```
## $ ihProgramDeclared          <int> 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1~
## $ iaProgramDeclared          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ paProgramDeclared          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1~
## $ hmProgramDeclared          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ incidentBeginDate          <chr> "2021-07-29T00:00:00.000Z", "2021-07-29T00:~
## $ incidentEndDate            <chr> "2021-07-30T00:00:00.000Z", "2021-07-30T00:~
## $ disasterCloseoutDate       <chr> "", "", "", "", "", "", "", "", "", "", "",~
## $ tribalRequest              <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ fipsStateCode              <int> 50, 50, 4, 4, 4, 47, 47, 47, 47, 22, 22, 22~
## $ fipsCountyCode             <int> 3, 25, 1, 5, 17, 43, 81, 83, 85, 7, 57, 55,~
## $ placeCode                  <int> 99003, 99025, 99001, 99005, 99017, 99043, 9~
## $ designatedArea             <chr> "Bennington (County)", "Windham (County)", ~
## $ declarationRequestNumber   <int> 21103, 21103, 21096, 21096, 21096, 21086, 2~
## $ lastIAFilingDate           <chr> "", "", "", "", "", "2021-10-25T00:00:00.00~
## $ lastRefresh.x              <chr> "2023-05-22T03:41:22.800Z", "2023-05-22T03:~
## $ hash.x                     <chr> "0fa835432c30b0e1f72f73fd5e49d97980681a79",~
## $ id.x                       <chr> "4a2dde3b-9597-4773-b92f-b08b0f015c5e", "0d~
## $ totalNumberIaApproved      <int> NA, NA, NA, NA, NA, 955, 955, 955, 955, 250~
## $ totalAmountIhpApproved     <dbl> NA, NA, NA, NA, NA, 8138894, 8138894, 81388~
## $ totalAmountHaApproved      <dbl> NA, NA, NA, NA, NA, 6301770, 6301770, 63017~
## $ totalAmountOnaApproved     <dbl> NA, NA, NA, NA, NA, 1837123, 1837123, 18371~
## $ totalObligatedAmountPa     <dbl> 7017657, 7017657, 5330186, 5330186, 5330186~
## $ totalObligatedAmountCatAb  <dbl> 219635.8, 219635.8, 117036.4, 117036.4, 117~
## $ totalObligatedAmountCatC2g <dbl> 6250266, 6250266, 5021941, 5021941, 5021941~
## $ paLoadDate                 <chr> "2023-10-30T00:00:00.000Z", "2023-10-30T00:~
## $ iaLoadDate                 <chr> "", "", "", "", "", "2023-10-30T00:00:00.00~
## $ totalObligatedAmountHmgp   <dbl> 276210.0, 276210.0, 0.0, 0.0, 0.0, 450000.0~
## $ hash.y                     <chr> "66f3d58cf4846dc2bc082fb9ca3a1db9a0dbc9b2",~
## $ lastRefresh.y              <chr> "2023-10-30T04:22:46.792Z", "2023-10-30T04:~
## $ id.y                       <chr> "b5b04ccf-5a3a-4f87-9151-5c21560cf210", "b5~
```

```
floods_by_state <- merged_data %>%
  group_by(state) %>%
  summarise(n = n(), .groups = 'drop')  # Ensuring that the resulting data frame will be ungrouped
```

I merge the data of FEMA.

data clean for census

```
total2020<-read_excel("total_2020.xls")
total2021<-read_excel("total_2021.xls")
total2020<- total2020 %>%
  mutate(Geography = str_sub(Geography, 1, -4))%>%
 mutate(`Geographic Area Name` = str_replace(`Geographic Area Name`, ".*?,\\s*", ""))
total2021<- total2021 %>%
  mutate(Geography = str_sub(Geography, 1, -4))%>%
mutate(`Geographic Area Name` = str_replace(`Geographic Area Name`, ".*?,\\s*", ""))
```

I just use the data of total population and poverty.

```r
total2020_processed <- total2020 %>%
  mutate(Geography = str_sub(Geography, 1, -4)) %>%
  mutate(`Geographic Area Name` = str_replace(`Geographic Area Name`, ".*?,\\s*", ""))

total2021_processed <- total2021 %>%
  mutate(Geography = str_sub(Geography, 1, -4)) %>%
  mutate(`Geographic Area Name` = str_replace(`Geographic Area Name`, ".*?,\\s*", ""))

# Assuming 'Total Population' and 'poverty' are column names in your data
# Combine the two data sets and calculate mean for 'Total Population' and 'poverty'
combined_data <- bind_rows(total2020_processed, total2021_processed, .id = "year") %>%
  group_by(`Geographic Area Name`) %>%
  summarise(
    Mean_Total_Population = mean(`Total population`, na.rm = TRUE),
    Mean_poverty = mean(poverty, na.rm = TRUE),
    .groups = 'drop'
  )
```

I calculate the average population of all cities in each state

```r
us_state_abbreviations <- data.frame(
  State = c("Alabama", "Alaska", "Arizona", "Arkansas", "California", "Colorado",
            "Connecticut", "Delaware", "Florida", "Georgia", "Hawaii", "Idaho",
            "Illinois", "Indiana", "Iowa", "Kansas", "Kentucky", "Louisiana",
            "Maine", "Maryland", "Massachusetts", "Michigan", "Minnesota",
            "Mississippi", "Missouri", "Montana", "Nebraska", "Nevada", "New Hampshire",
            "New Jersey", "New Mexico", "New York", "North Carolina", "North Dakota",
            "Ohio", "Oklahoma", "Oregon", "Pennsylvania", "Rhode Island",
            "South Carolina", "South Dakota", "Tennessee", "Texas", "Utah", "Vermont",
            "Virginia", "Washington", "West Virginia", "Wisconsin", "Wyoming"),
  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")
)

combined_data <- combined_data %>%
  left_join(us_state_abbreviations, by = c("Geographic Area Name" = "State")) %>%
  # Remove the full state name column if you don't need it
  select(-`Geographic Area Name`) %>%
  # Rename Abbreviation to State or another name if you prefer
  rename(State = Abbreviation)

f_data <- combined_data %>%
  filter(State %in% c("KY", "ND", "WA", "WV", "TX", "LA", "TN", "OR", "WI", "AZ", "VT", "PR", "HI"))
f_data <- f_data %>%
  rename(state = State)
state_summary <- merge(floods_by_state, f_data, by = "state")
```

I use the abbreviation of state instead of the full name of the state.

```r
# Assuming merged_data and f_data are already in your R environment

# Write merged_data to a CSV file
write.csv(merged_data, "merged_data.csv", row.names = FALSE)

# Write f_data to a CSV file (replace f_data with the actual name of your dataframe)
write.csv(f_data, "f_data.csv", row.names = FALSE)
```

data clean for storm

```r
df_2020 <- read_csv('2020storm.csv')
```

```
## Rows: 61279 Columns: 51
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (26): STATE, MONTH_NAME, EVENT_TYPE, CZ_TYPE, CZ_NAME, WFO, BEGIN_DATE_T...
## dbl (25): BEGIN_YEARMONTH, BEGIN_DAY, BEGIN_TIME, END_YEARMONTH, END_DAY, EN...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
df_2021 <- read_csv('2021storm.csv')
```

```
## Rows: 61389 Columns: 51
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (26): STATE, MONTH_NAME, EVENT_TYPE, CZ_TYPE, CZ_NAME, WFO, BEGIN_DATE_T...
## dbl (25): BEGIN_YEARMONTH, BEGIN_DAY, BEGIN_TIME, END_YEARMONTH, END_DAY, EN...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

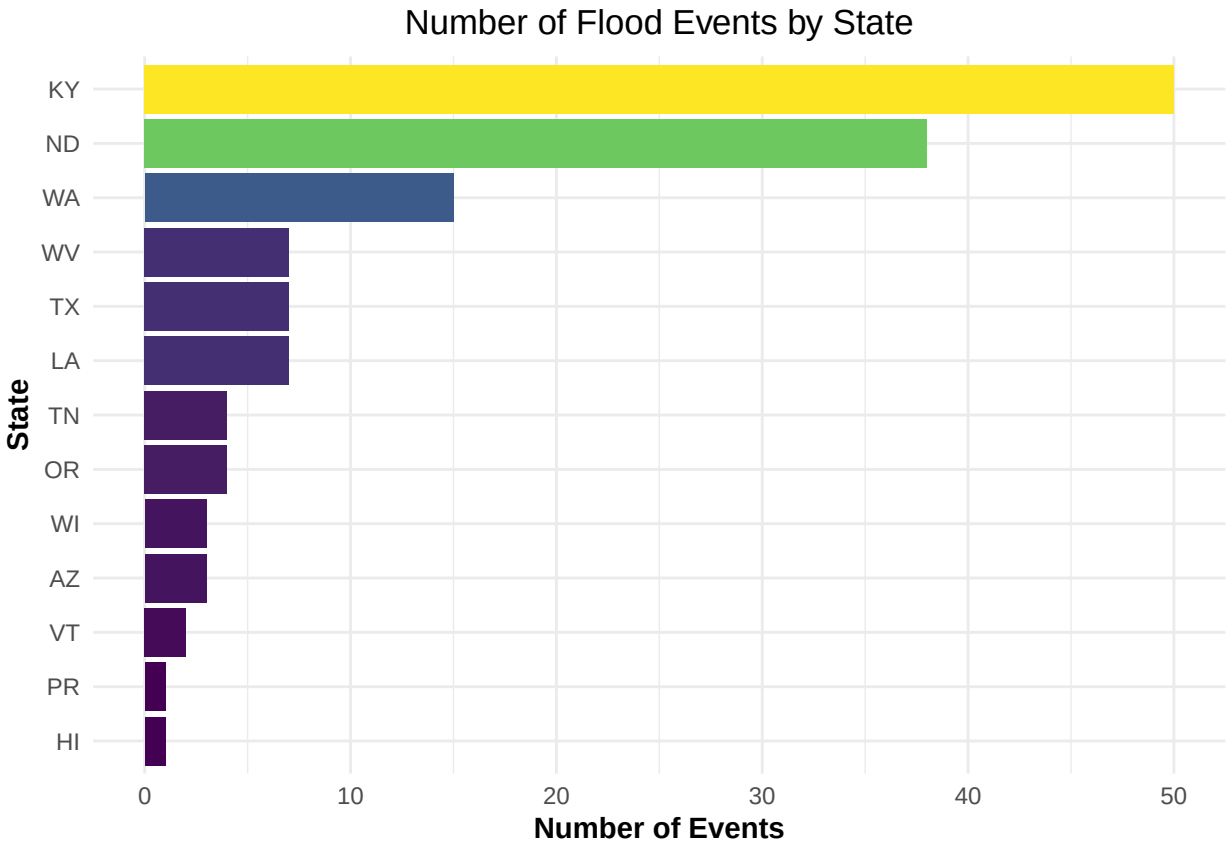
```r
merged_df <- full_join(df_2020, df_2021, by = intersect(names(df_2020), names(df_2021)))
flood_events <- merged_df %>% filter(grepl('Flood', EVENT_TYPE))%>%
  select(BEGIN_TIME,END_TIME,MONTH_NAME,STATE,DAMAGE_PROPERTY,DAMAGE_CROPS)
convert_damage <- function(damage) {
  damage <- gsub("K", "", damage)
  damage <- gsub("M", "", damage)
  numeric_damage <- as.numeric(damage)
  ifelse(grepl("K", damage), numeric_damage * 1e3,
         ifelse(grepl("M", damage), numeric_damage * 1e6,
                numeric_damage))
}
flood_events <- flood_events %>%
  mutate(DAMAGE_PROPERTY = convert_damage(DAMAGE_PROPERTY),
         DAMAGE_CROPS = convert_damage(DAMAGE_CROPS),
         totaldamage = DAMAGE_PROPERTY + DAMAGE_CROPS)
```

EDA - Exploratory Data Analysis

which states floods the most frequently?(among data of FEMA during 2020 and 2021) ?

Summary statistics for FEMA(by state)

```
# Now, you can plot the data
ggplot(floods_by_state, aes(x = reorder(state, n), y = n, fill = n)) +
  geom_bar(stat = "identity") +
  coord_flip() + # Flip coordinates for horizontal bars
  scale_fill_viridis_c() + # Use the viridis color scale
  labs(title = "Number of Flood Events by State", x = "State", y = "Number of Events") +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.title.x = element_text(face = "bold"),
    axis.title.y = element_text(face = "bold"),
    legend.title = element_blank(),
    legend.position = "bottom"
  ) +
  guides(fill = FALSE)  # Remove the color guide
```



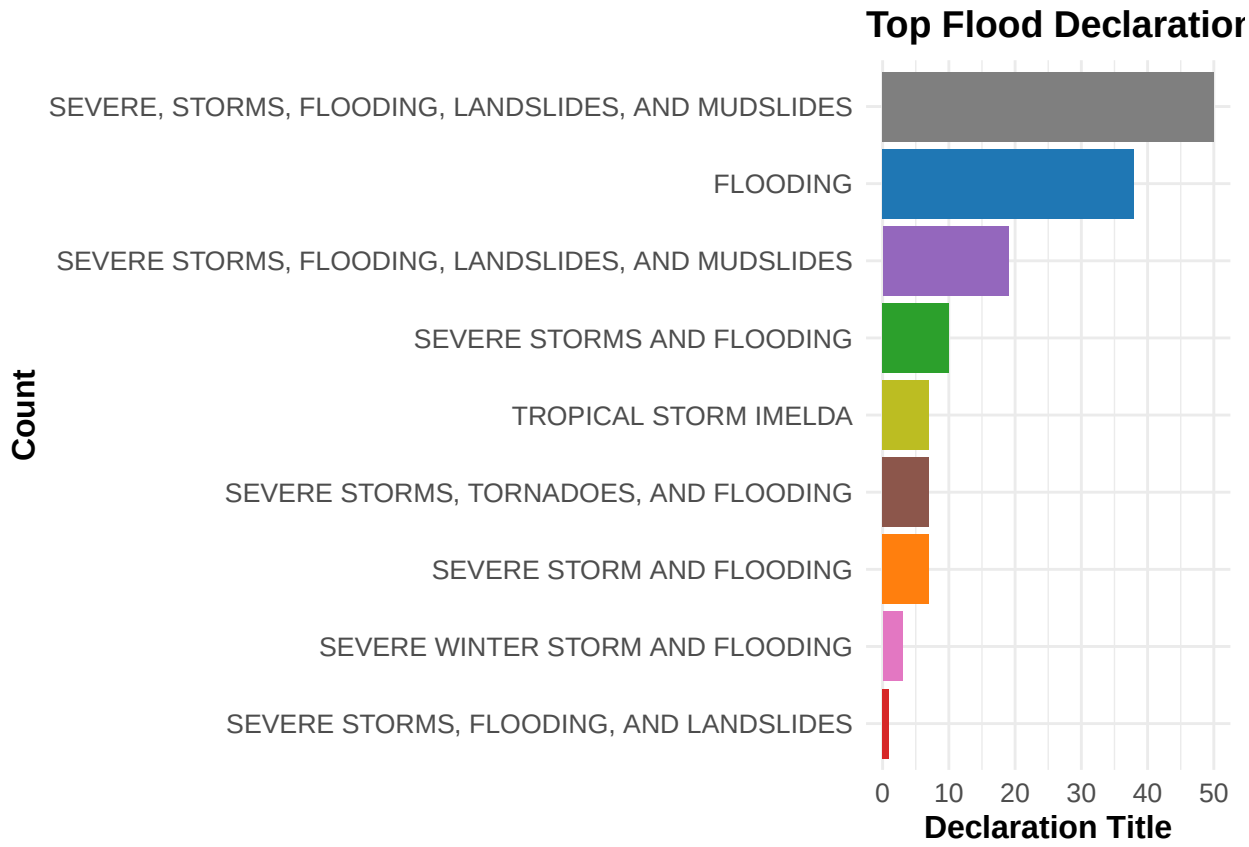
```r
# Save the enhanced plot
ggsave("number_of_floods_by_state_enhanced.png", width = 10, height = 8, dpi = 300)
```

Based on data,we can find KY state has the Most flood accidents.

What are the main causes of flooding?

Summary statistics for FEMA(by reasons)

```r
# Group and summarise as before
grouped_data <- merged_data %>%
  group_by(declarationTitle) %>%
  summarise(Count = n(), .groups = 'drop')

# Filter for the top N declaration titles based on count
top_n_titles <- grouped_data %>%
  top_n(10, Count) # Adjust the number 10 to show more or fewer categories

custom_colors <- c("#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd", "#8c564b",
                   "#e377c2", "#7f7f7f", "#bcbd22", "#17becf")

ggplot(top_n_titles, aes(x = reorder(declarationTitle, Count), y = Count, fill = declarationTitle)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  scale_fill_manual(values = custom_colors) + # Use custom colors
  theme_minimal() +
  theme(legend.position = "none") +
  labs(title = "Top Flood Declarations by Title", x = "Count", y = "Declaration Title") +
  theme(plot.title = element_text(size = 14, face = "bold"),
        axis.title.x = element_text(size = 12, face = "bold"),
        axis.title.y = element_text(size = 12, face = "bold"),
        axis.text.x = element_text(size = 10),
        axis.text.y = element_text(size = 10))
```

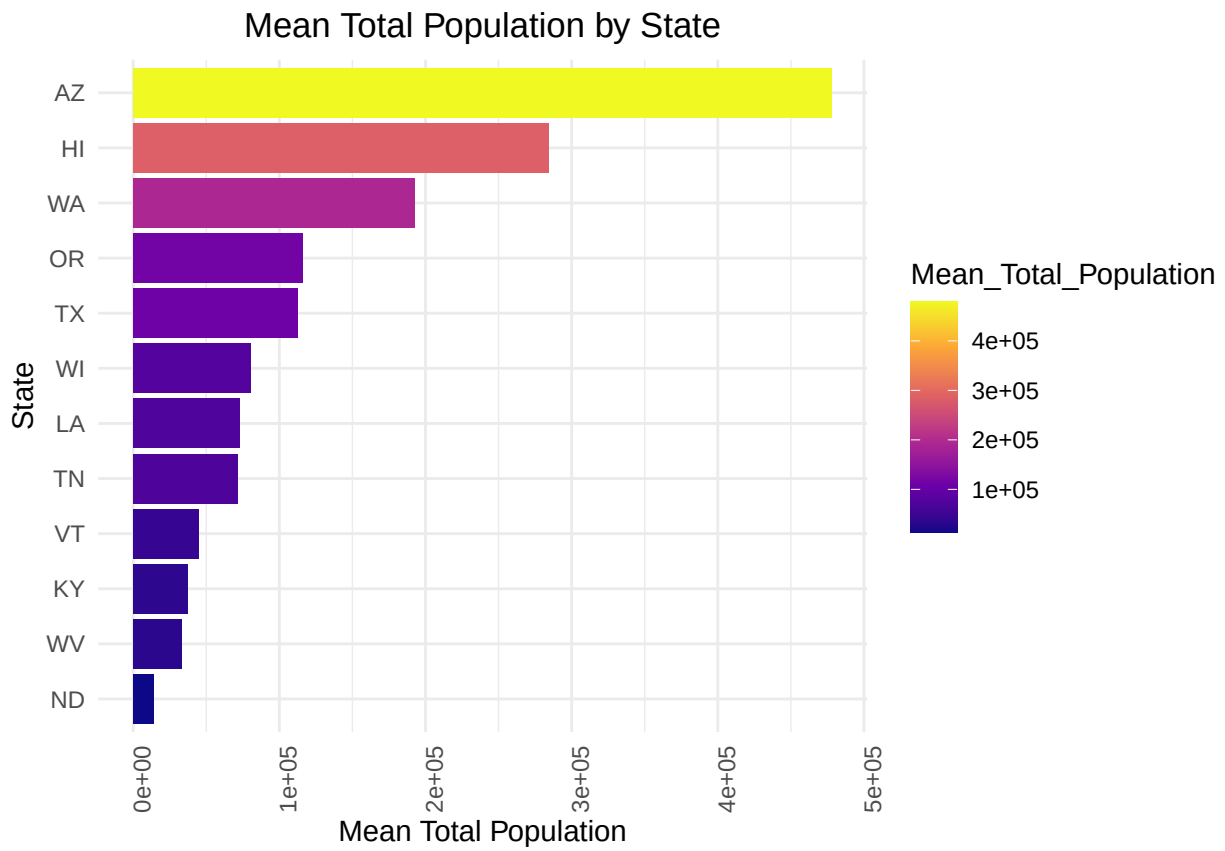


Finally, I found the reason of floods mostly Severe storm,flooding,landslides and mudslides,from the data given by FEMA.

Which state has the highest average population size?

summary for census

```
f_data_ordered_population <- f_data %>%
  arrange(desc(Mean_Total_Population))

ggplot(f_data_ordered_population, aes(x = reorder(state, Mean_Total_Population), y = Mean_Total_Populati
  geom_bar(stat = "identity") +
  coord_flip() + # Flip coordinates for horizontal bars
  scale_fill_viridis_c(option = "C") + # Use a viridis color scale for a visually appealing palette
  labs(title = "Mean Total Population by State",
       x = "State",
       y = "Mean Total Population") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        plot.title = element_text(hjust = 0.5))
```

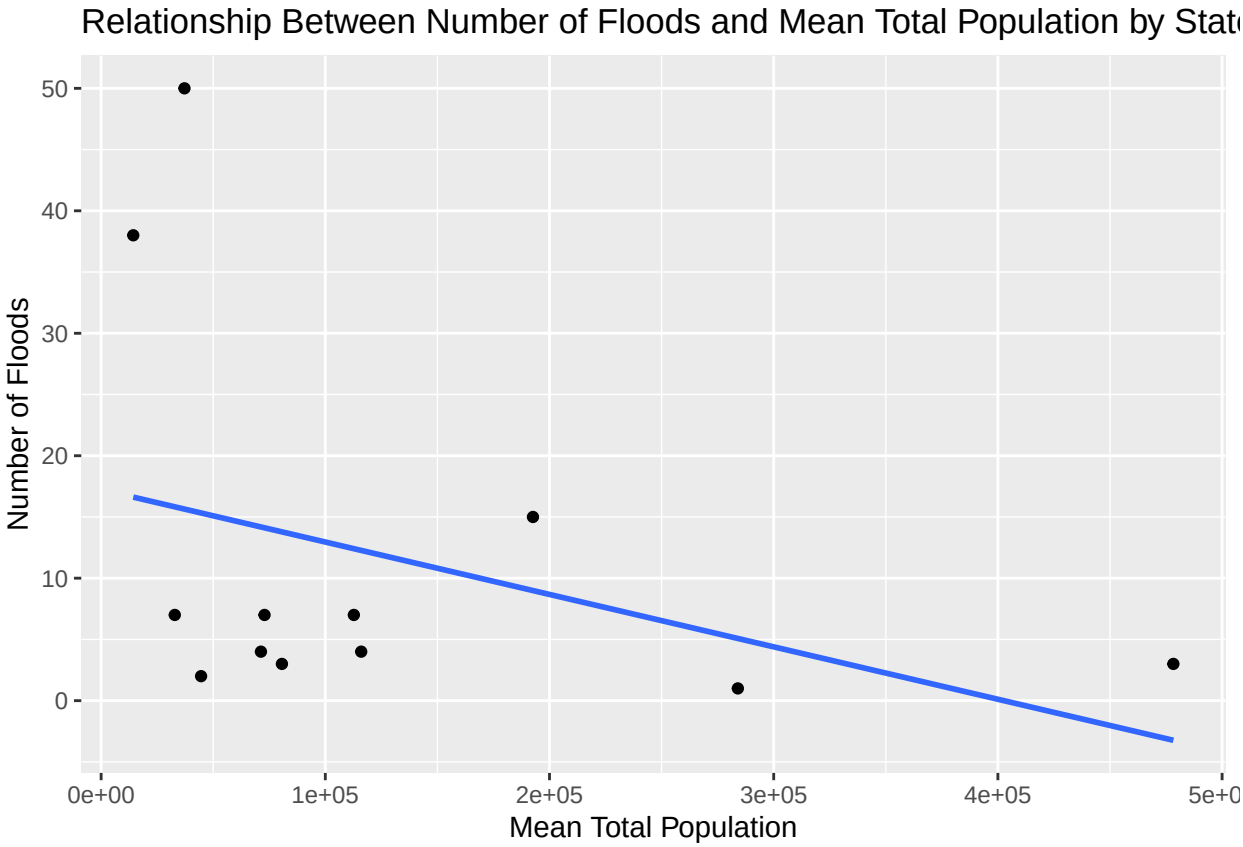


based on the data of total population,I find AZ has the highest mean value of total population by state. ## Is there any relationship between population and floods number group by state?

merge census and FEMA

```
ggplot(state_summary, aes(x = Mean_Total_Population, y = n)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Relationship Between Number of Floods and Mean Total Population by State",
       x = "Mean Total Population",
       y = "Number of Floods")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



So finally I don't find there has a relationship between mean total population and frequency of floods.For this part,maybe due to the dataset of FEMA is small after I merged,I don't explore the relationship between population with it,so for the next part,I believe there must exist a relationship between storm and flood,so I will use the data of storm2020 and 2021 to analysis.Before start the EDA for the next part,I just summarize what I find in these first 2 data,that is during 2020 and 2021,the main cause of flood is storm,and KY state is the most frequently one.

summary for storm

I found that the main cause of flood is storm,so I use the data of storm to do more eda. ## Which state has suffered the most damage due to flooding, and what is the damage like in each state?

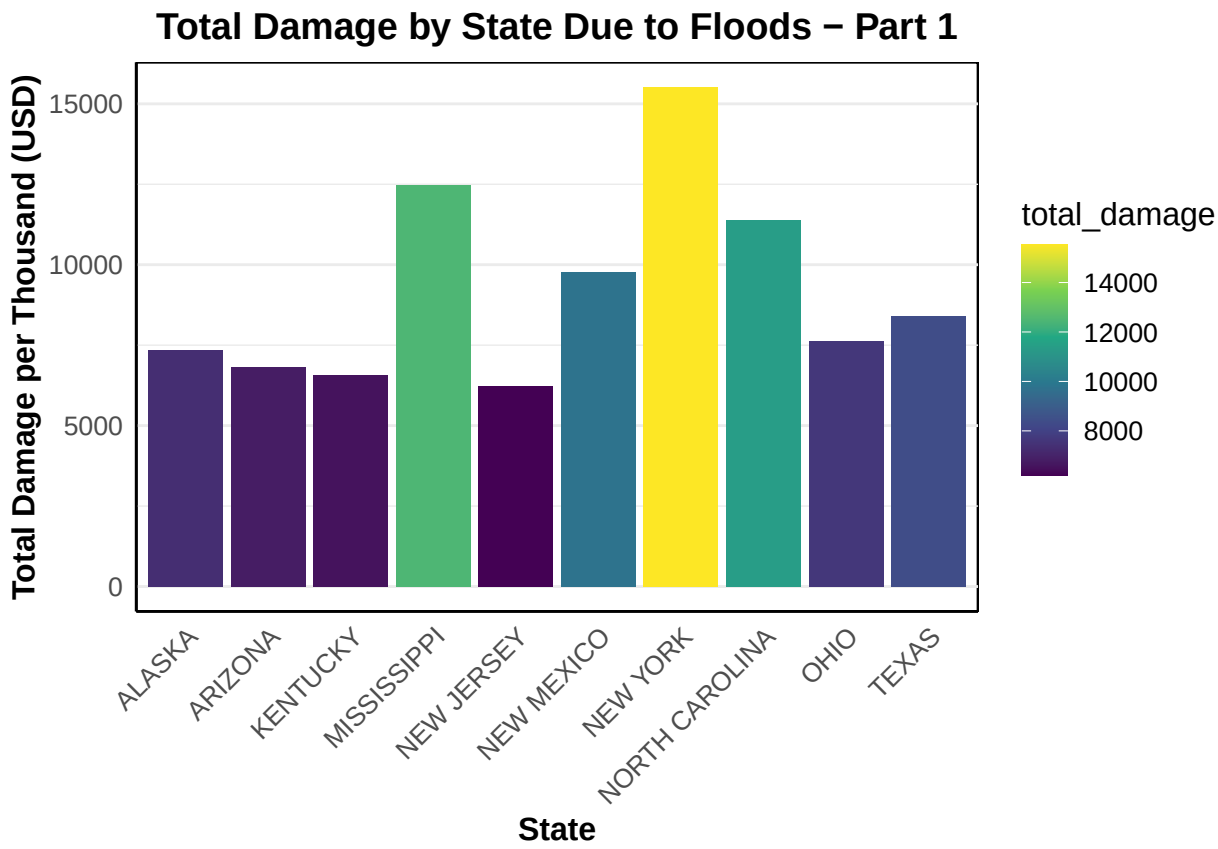
```
state_damage <- flood_events %>%
  group_by(STATE) %>%
  summarise(total_damage = sum(totaldamage, na.rm = TRUE)) %>%
  arrange(desc(total_damage))

# Calculate the number of plots needed (e.g., if there are too many states for one plot)
num_plots <- ceiling(nrow(state_damage) / 10) # Adjust the divisor as necessary for readability

# Create a list to hold the plot objects
plot_list <- vector("list", num_plots)

# Split the data into multiple parts and create a plot for each
for (i in 1:num_plots) {
```

```
  # Subset the data for the current plot
  subset_data <- state_damage[((i-1)*10 + 1):(i*10), ]

  # Generate the plot for the current subset
  plot_list[[i]] <- ggplot(subset_data, aes(x = STATE, y = total_damage, fill = total_damage)) +
    geom_bar(stat = "identity", position = "dodge") +
    scale_fill_viridis_c() + # Use a viridis color scale for the bars
    labs(x = "State", y = "Total Damage per Thousand (USD)", title = paste("Total Damage by State Due t
    theme_minimal() +
    theme(
      text = element_text(size = 12),
      axis.title.x = element_text(face = "bold", size = 12),
      axis.title.y = element_text(face = "bold", size = 12),
      plot.title = element_text(hjust = 0.5, face = "bold", size = 14),
      axis.text.x = element_text(angle = 45, hjust = 1), # Rotate x axis labels for better readability
      panel.grid.major.x = element_blank(), # Remove vertical grid lines for a cleaner look
      panel.grid.minor.x = element_blank(),
      panel.background = element_rect(fill = "white"),
      axis.line = element_line(colour = "black")
    )

  # Save the current plot to a file
  ggsave(paste("state_damage_plot_part", i, ".png", sep = ""), plot_list[[i]], width = 12, height = 7)
}
```

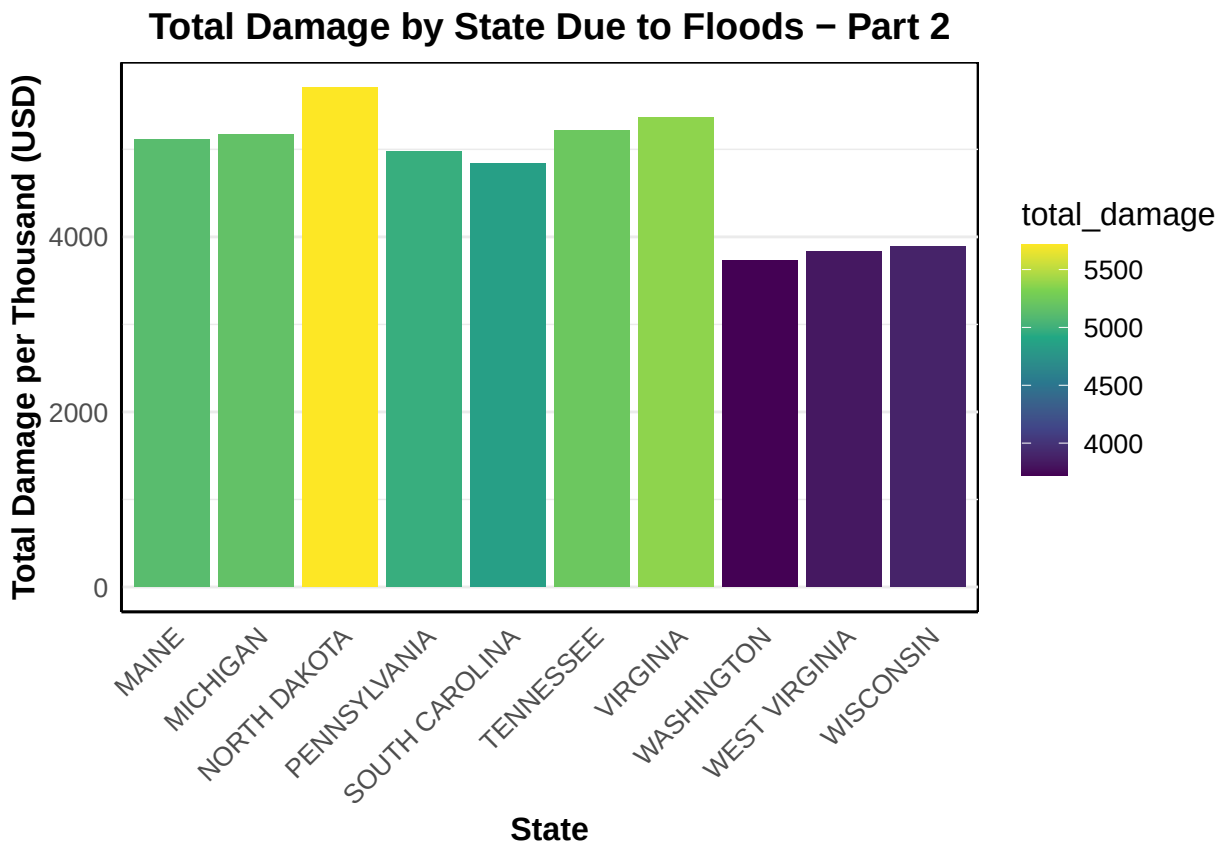
```
## Warning: Removed 5 rows containing missing values (`geom_bar()`).
```

```
# Return the list of plots to display in R environment
plot_list
```

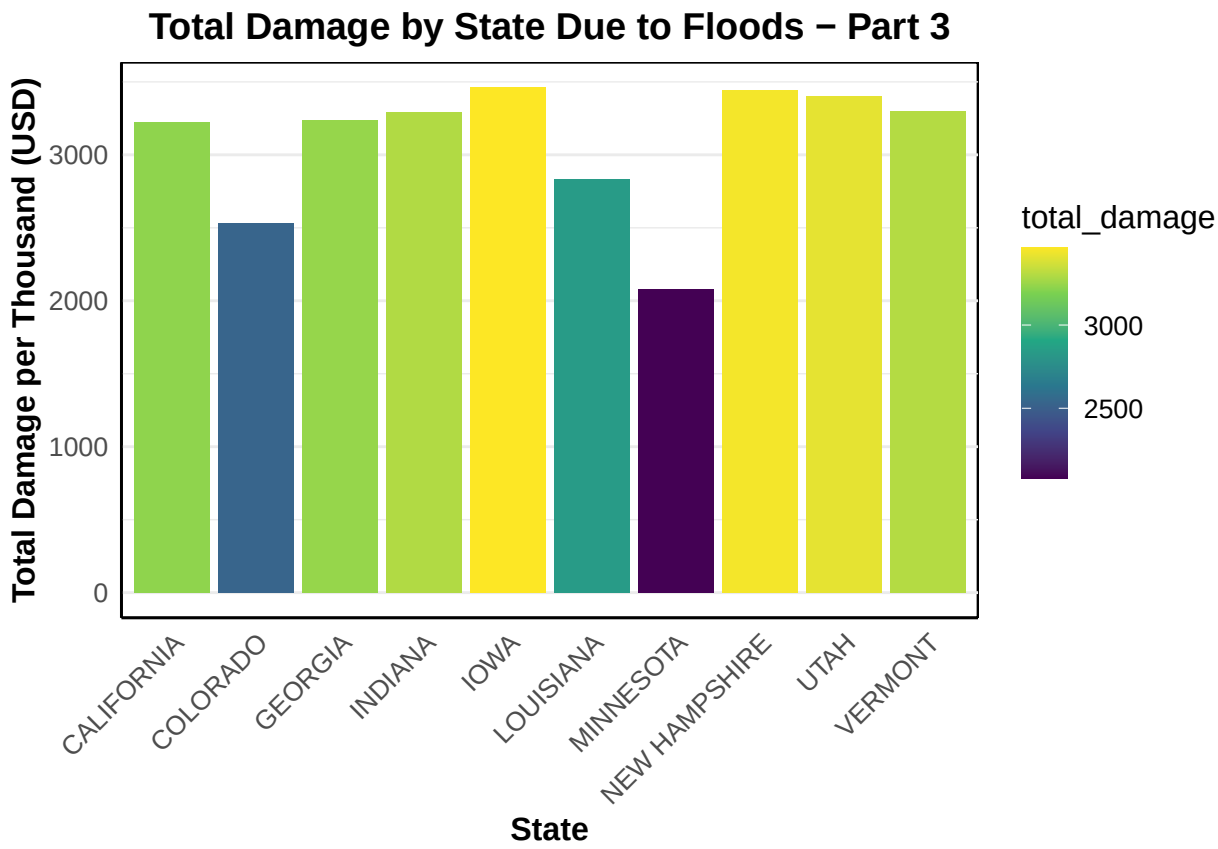
```
## [[1]]
```


Total Damage by State Due to Floods − Part 1
Total Damage per Thousand (USD)
15000
10000
5000
0
ALASKA
ARIZONA
KENTUCKY
MISSISSIPPI
NEW JERSEY
NEW MEXICO
NEW YORK
NORTH CAROLINA
OHIO
TEXAS
State
total_damage
14000
12000
10000
8000

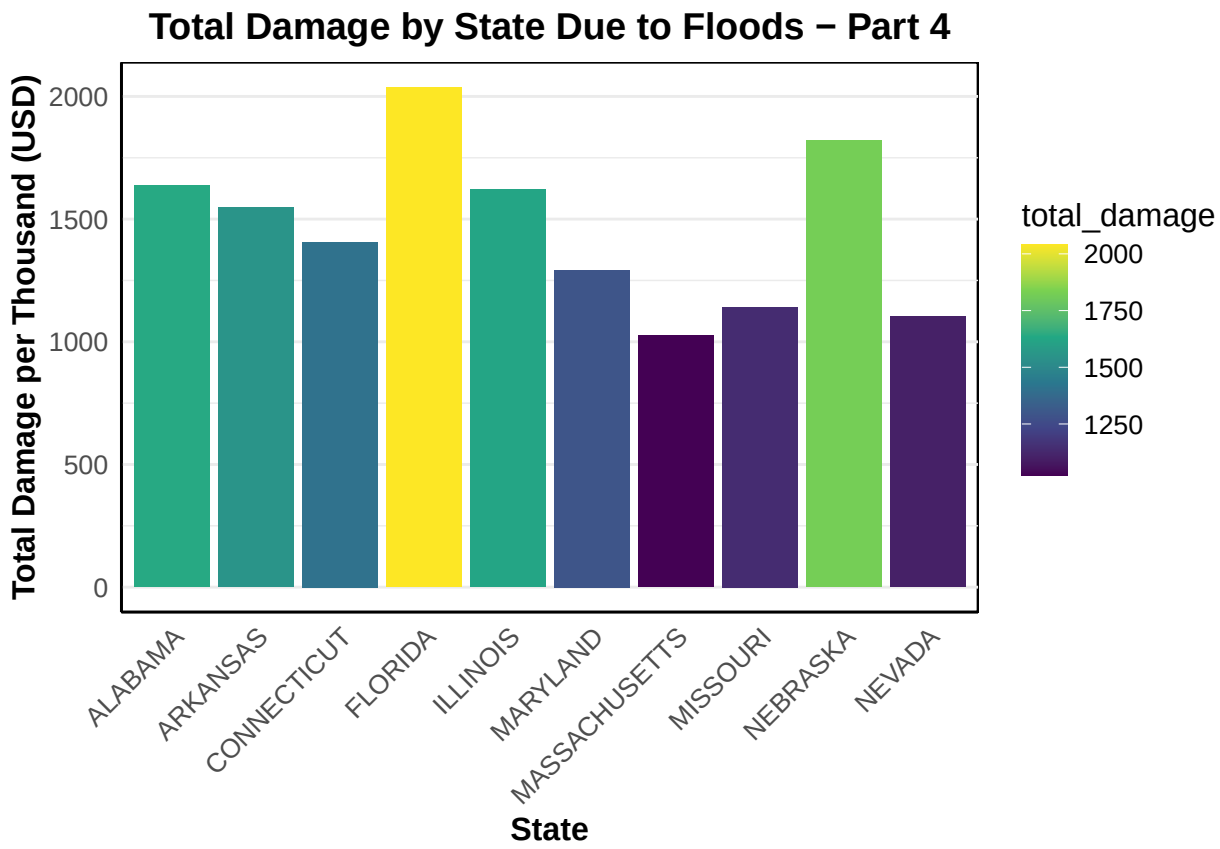
```
##
## [[2]]
```


Total Damage by State Due to Floods − Part 2
Total Damage per Thousand (USD)
0
2000
4000
MAINE
MICHIGAN
NORTH DAKOTA
PENNSYLVANIA
SOUTH CAROLINA
TENNESSEE
VIRGINIA
WASHINGTON
WEST VIRGINIA
WISCONSIN
State
total_damage
5500
5000
4500
4000

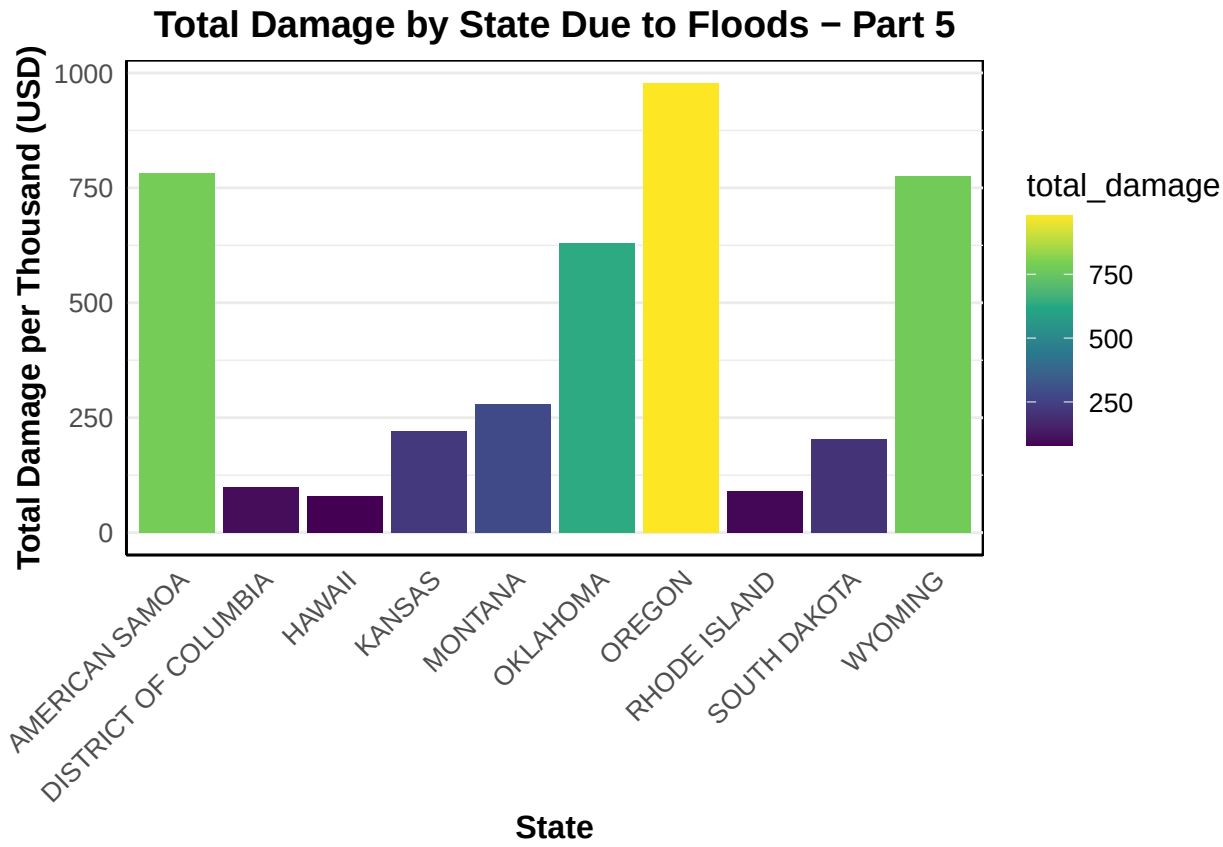
```
##
## [[3]]
```


Total Damage by State Due to Floods − Part 3
Total Damage per Thousand (USD)
3000
2000
1000
0
CALIFORNIA
COLORADO
GEORGIA
INDIANA
IOWA
LOUISIANA
MINNESOTA
NEW HAMPSHIRE
UTAH
VERMONT
State
total_damage
3000
2500

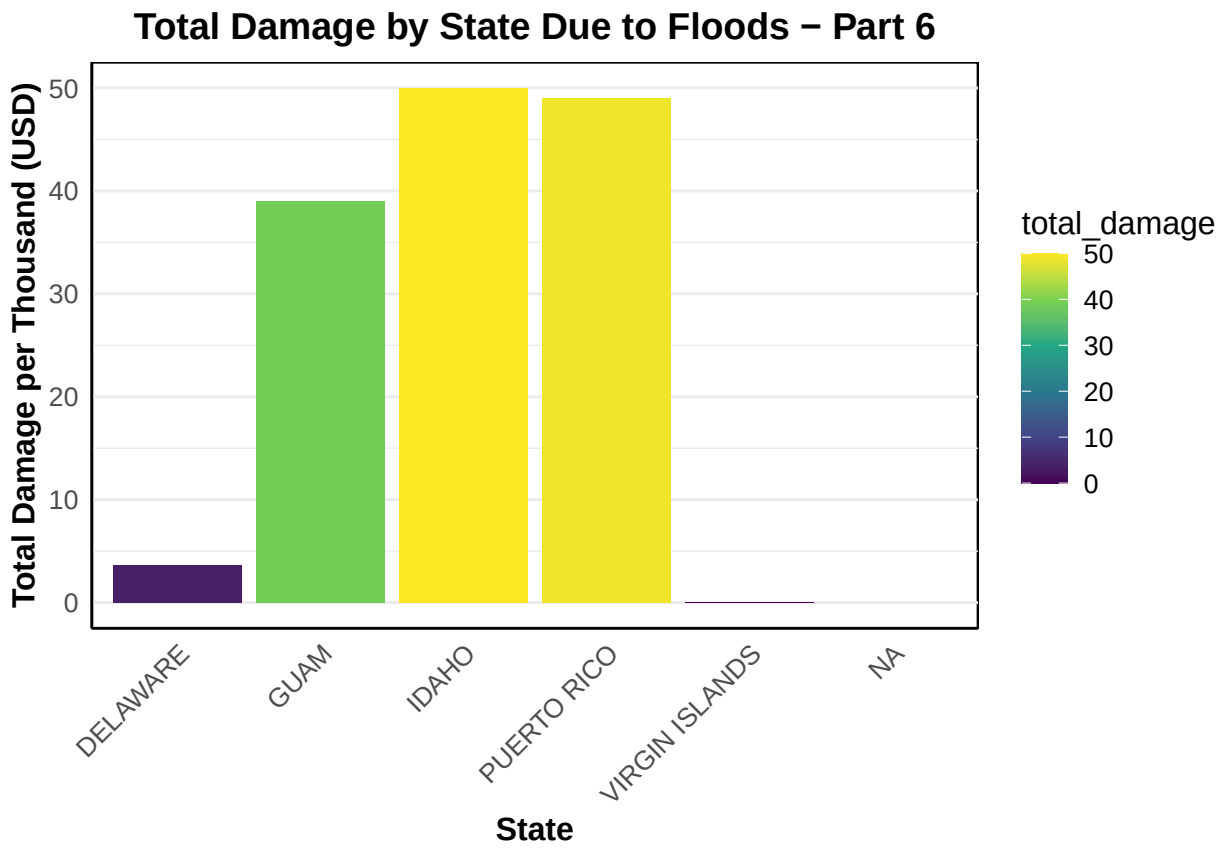
```
##
## [[4]]
```


Total Damage by State Due to Floods − Part 4
Total Damage per Thousand (USD)
2000
1500
1000
500
0
ALABAMA
ARKANSAS
CONNECTICUT
FLORIDA
ILLINOIS
MARYLAND
MASSACHUSETTS
MISSOURI
NEBRASKA
NEVADA
State
total_damage
2000
1750
1500
1250

```
##
## [[5]]
```



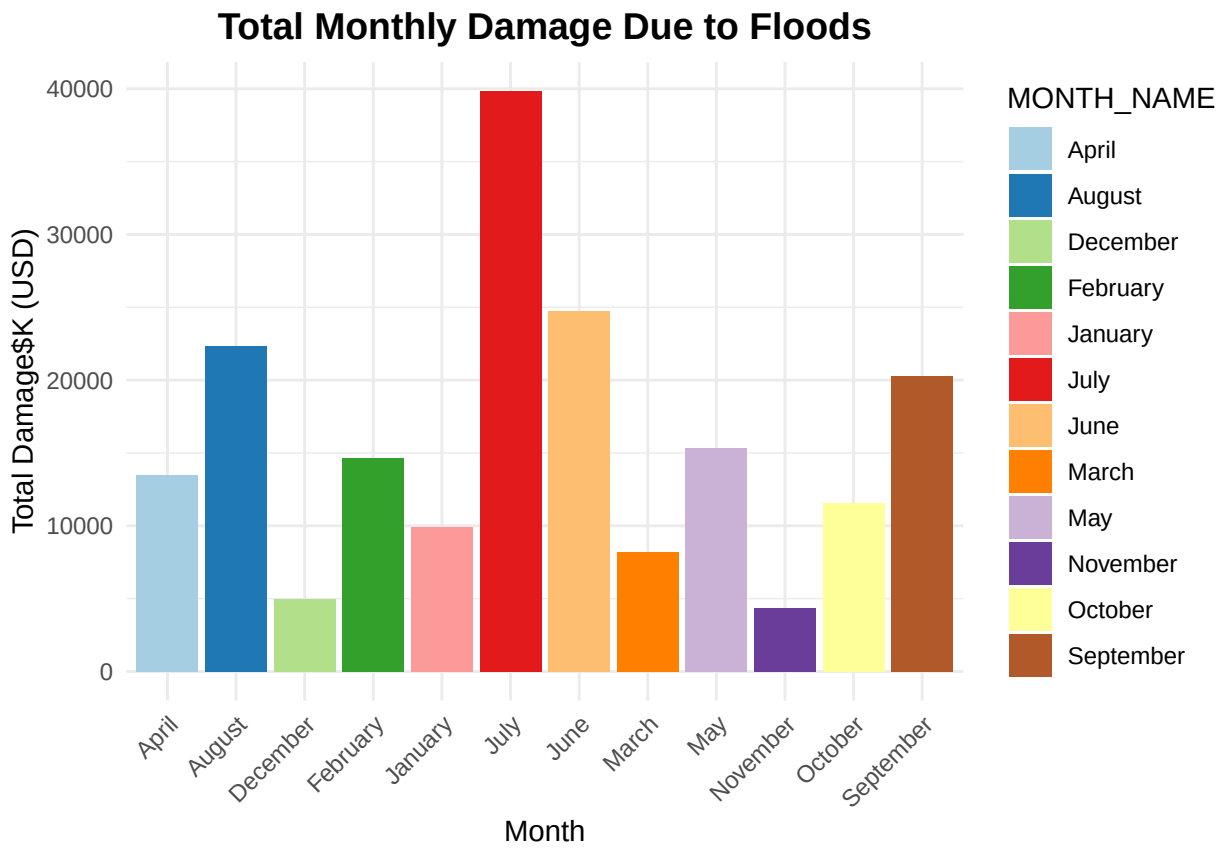

```
##
## [[6]]

## Warning: Removed 5 rows containing missing values (‘geom_bar()‘).
```



I think the New York state suffer the most damage due to flood during2020 to 2021,and I just seperate state by the amount of damage for 6 part to make it more visualization.So,for the next part,I want to find which month has the most damage due to floods.

Which month has suffered the most damage due to flooding, and what is the damage like in each month?

```
monthly_damage <- flood_events %>%
  group_by(MONTH_NAME) %>%
  summarise(total_damage = sum(totaldamage, na.rm = TRUE)) %>%
  arrange(match(MONTH_NAME, month.name)) # Arrange by month order
ggplot(monthly_damage, aes(x = MONTH_NAME, y = total_damage, fill = MONTH_NAME)) +
  geom_bar(stat = "identity") +
  scale_fill_brewer(palette = "Paired") + # Use a color scale that is distinct for each month
  labs(x = "Month", y = "Total Damage$K (USD)", title = "Total Monthly Damage Due to Floods") +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1), # Angle the month names for better readability
    plot.title = element_text(size = 14, face = "bold", hjust = 0.5) # Bold and center the plot title
  )
```



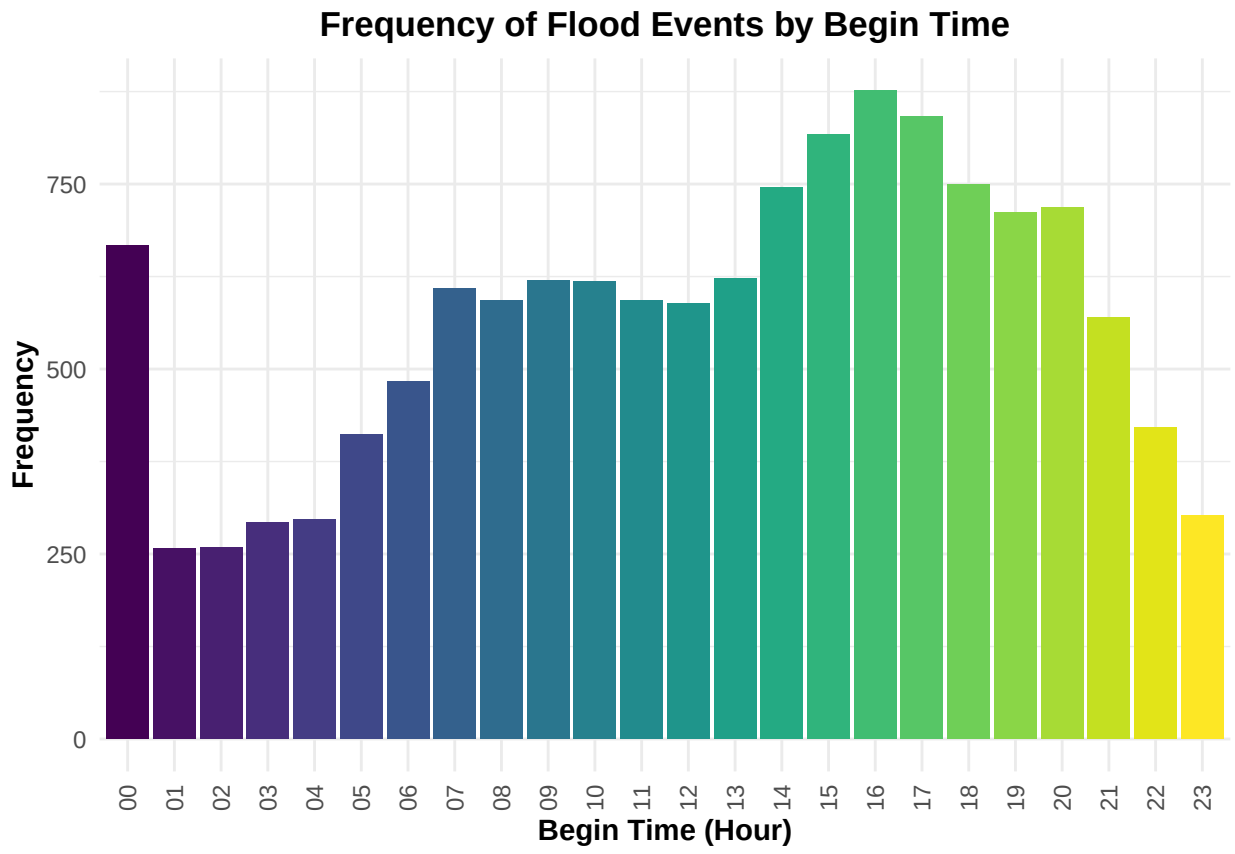
I find that the month with the most losses due to floods is July,and June,August,September also suffers from a lot of losses,so it meets my expectations,the loss of flood disaster casused by rainy seanson is relatively large.So,for the next part,I want to find which time occurs most frequently during the day.

which time occurs most frequently during the day?

```
flood_events <- flood_events %>%
  mutate(BEGIN_TIME = sprintf("%04d", as.integer(BEGIN_TIME)))
flood_events <- flood_events %>%
  mutate(BEGIN_TIME = substr(as.character(BEGIN_TIME), 1, 2))

# Count the number of occurrences for each truncated begin time
begin_time_counts <- flood_events %>%
  group_by(BEGIN_TIME) %>%
  summarise(count = n())
```

```
frequency_plot <- ggplot(begin_time_counts, aes(x = BEGIN_TIME, y = count, fill = BEGIN_TIME)) +
  geom_bar(stat = "identity") +
  scale_fill_viridis_d() + # Use a discrete viridis color scale for aesthetic reasons
  labs(x = "Begin Time (Hour)", y = "Frequency", title = "Frequency of Flood Events by Begin Time") +
  theme_minimal() +
  theme(
    plot.title = element_text(face = "bold", hjust = 0.5),
    axis.title.x = element_text(face = "bold"),
```

```
    axis.title.y = element_text(face = "bold"),
    axis.text.x = element_text(angle = 90, vjust = 0.5),
    legend.position = "none" # Remove the legend since the fill is not informative
  )


# Print the plot to view in the R environment
print(frequency_plot)
```

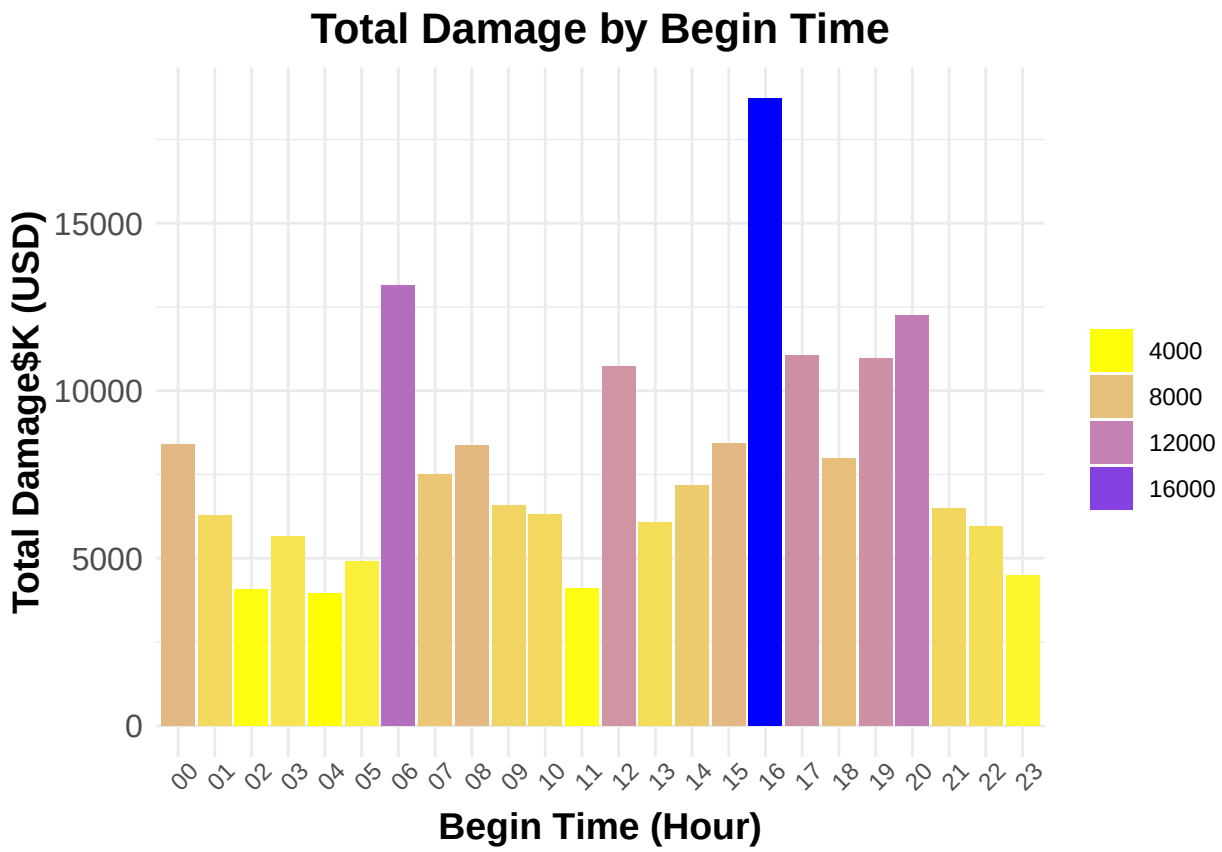


In general,it is most frequent in the afternoon.

which time occurs most damage during the day?

```
begin_time_damage <- flood_events %>%
  group_by(BEGIN_TIME) %>%
  summarise(total_damage = sum(totaldamage, na.rm = TRUE)) %>%
  arrange(desc(total_damage))
ggplot(begin_time_damage, aes(x = BEGIN_TIME, y = total_damage)) +
  geom_bar(stat = "identity", aes(fill = total_damage)) +
  scale_fill_gradient(low = "yellow",high="blue") + # Beautiful gradient from low (blue) to high (red)
  labs(x = "Begin Time (Hour)", y = "Total Damage$K (USD)", title = "Total Damage by Begin Time") +
  theme_minimal() +
  theme(
    plot.title = element_text(face = "bold", size = 16, hjust = 0.5),
```

```
    axis.title.x = element_text(face = "bold", size = 14),
    axis.title.y = element_text(face = "bold", size = 14),
    axis.text.x = element_text(angle = 45, vjust = 1),
    axis.text.y = element_text(size = 12),
    legend.title = element_blank(), # Remove legend title for a cleaner look
    legend.position = "right"
  ) +
  guides(fill = guide_legend(title.position = "top", title.hjust = 0.5))
```



I found that during the day time,16h has the highest total damage. ### summarize

In general,I do the analysis for floods during 2020 and 2021 to the FEMA and CENCUS part,I find that KY state has the highest frequency among the state,and find the most cause of the floods is storm,there don't exist some relationship between populations and floods frequency,then,I use the data of storm to make a deeper understanding of data,I use storm to analysis and find New York state has the highest amount of damage,and then I find among all the months,July has the highest amount of damage,during the day time,16 o'clock has the highest amount of damage and 16 o'clock is also the most frequency time.