Simpsons Tidy Tuesday - Khushi Choudhary

The Simpson's data!

Problems with the Dats: The script lines are of great interest, but it is a larger file, too big for Tidy Tuesday. We need to reduce the size of the file so we can use all the files together for a more robust analysis. Let's filter episodes down to the years 2010-2016, and then only select the script lines that correspond with those episodes.

```r
options(repos = c(CRAN = "https://cloud.r-project.org"))
install.packages("tidytuesdayR")
```

```
The downloaded binary packages are in
    /var/folders/n5/8vddkp6x6bz_2r5xz9nb_8pc0000gn/T//Rtmpk9kUi4/downloaded_packages
```

```r
tuesdata <- tidytuesdayR::tt_load('2025-02-04')

simpsons_characters <- tuesdata$simpsons_characters
simpsons_episodes <- tuesdata$simpsons_episodes
simpsons_locations <- tuesdata$simpsons_locations
simpsons_script_lines <- tuesdata$simpsons_script_lines

show_col_types = FALSE
```

Script to clean the data sourced from Kaggle

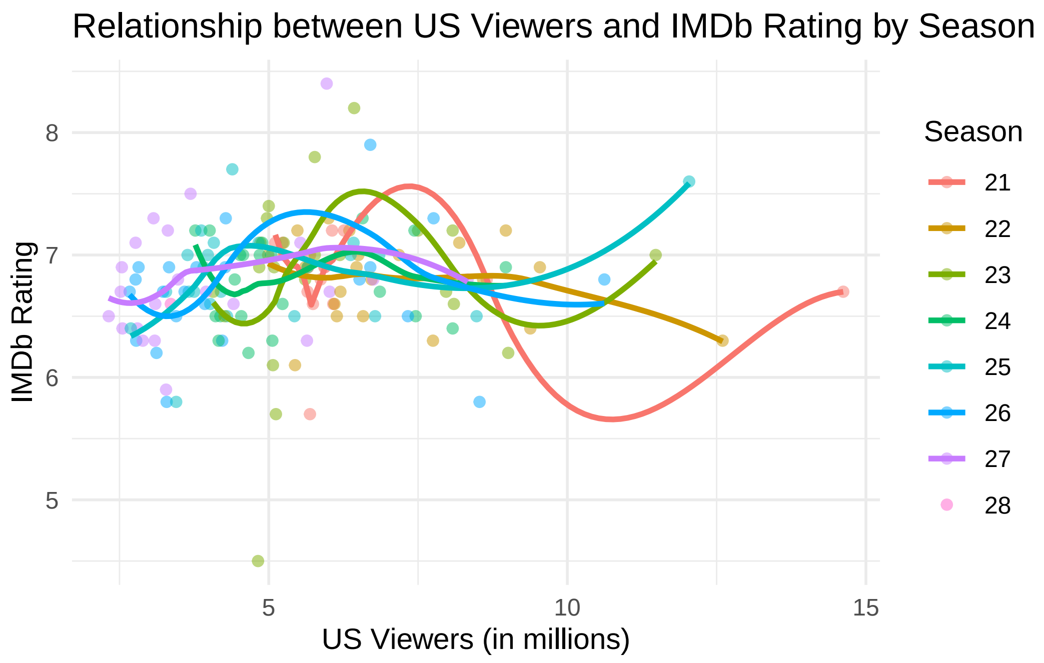
```r
# packages
library(httr)
library(tidyverse)
library(jsonlite)
library(withr)
```

```r
# Define the metadata URL and fetch it
metadata_url <- "www.kaggle.com/datasets/prashant111/the-simpsons-dataset/croissant/download"
response <- httr::GET(metadata_url)

# Ensure the request succeeded
if (httr::http_status(response)$category != "Success") {
  stop("Failed to fetch metadata.")
}

# Parse the metadata
metadata <- httr::content(response, as = "parsed", type = "application/json")

# Locate the ZIP file URL
distribution <- metadata$distribution
zip_url <- NULL

for (file in distribution) {
  if (file$encodingFormat == "application/zip") {
    zip_url <- file$contentUrl
    break
  }
}

if (is.null(zip_url)) {
  stop("No ZIP file URL found in the metadata.")
}

# Download the ZIP file. We'll use the withr package to make sure the downloaded
# files get cleaned up when we're done.
temp_file <- withr::local_tempfile(fileext = ".zip")
utils::download.file(zip_url, temp_file, mode = "wb")

# Unzip and read the CSV
unzip_dir <- withr::local_tempdir()
utils::unzip(temp_file, exdir = unzip_dir)

# Locate the CSV file within the extracted contents
csv_file <- list.files(unzip_dir, pattern = "\\.csv$", full.names = TRUE)

if (length(csv_file) == 0) {
  stop("No CSV file found in the unzipped contents.")
}
```

```r
# Read the CSV into a dataframe
simpsons_characters <- read_csv(csv_file[1])
simpsons_episodes <- read_csv(csv_file[2])
simpsons_locations <- read_csv(csv_file[3])
simpsons_script_lines <- read_csv(csv_file[4])


# filter episodes to include 2010+
simpsons_episodes <- simpsons_episodes |>
  dplyr::filter(original_air_year >= 2010)

# filter script lines to only include lines for these episodes
simpsons_script_lines <- simpsons_script_lines |>
  dplyr::semi_join(simpsons_episodes, by = c("episode_id" = "id"))
```

Relationship between US Viewers and IMDb Rating by Season

```r
ggplot(simpsons_episodes, aes(x = us_viewers_in_millions, y = imdb_rating, color = factor(sea
  geom_point(alpha = 0.5) +
  geom_smooth(se = FALSE) +
  labs(
    title = "Relationship between US Viewers and IMDb Rating by Season",
    x = "US Viewers (in millions)",
    y = "IMDb Rating",
    color = "Season"
  ) +
  theme_minimal()
```



The visualization explores the relationship between US viewership and IMDb ratings for Simpsons episodes, broken down by season. Each point represents an episode, colored by its season. Smooth trend lines are overlaid to show how the relationship between viewership and rating varies across different seasons.

```
cor(simpsons_episodes$us_viewers_in_millions, simpsons_episodes$imdb_rating, use = "complete
```

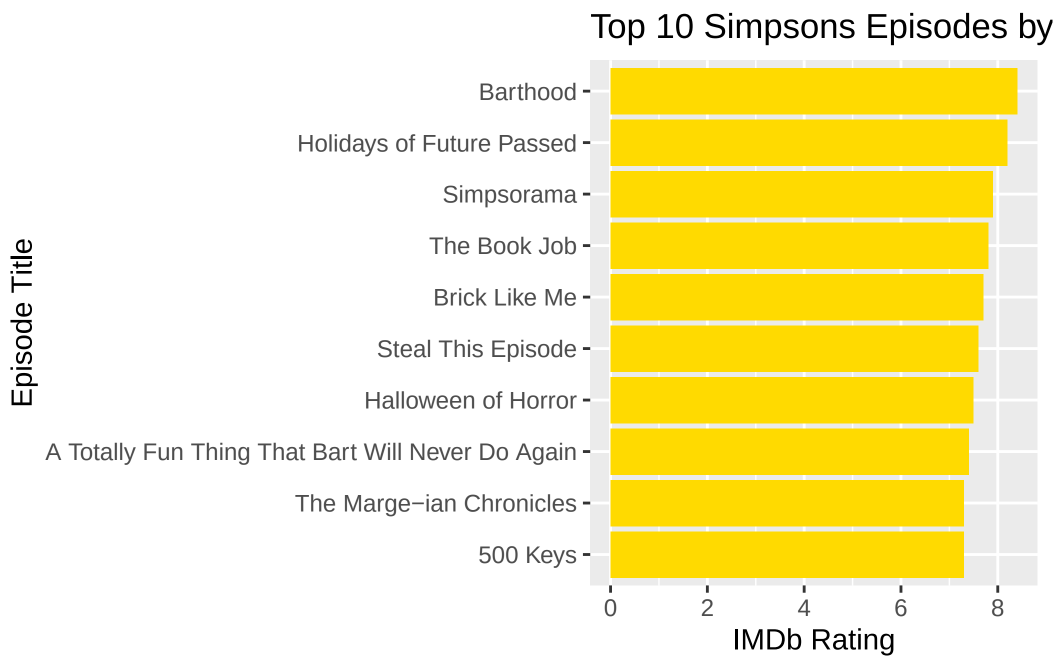
```
[1] 0.1119694
```

The correlation between US viewers and IMDb rating for Simpsons episodes is 0.11, indicating a very weak positive relationship. This suggests that there is little to no linear association between how many people watched an episode and how it was rated on IMDb.

Find the top 10 episodes with the highest IMDb ratings

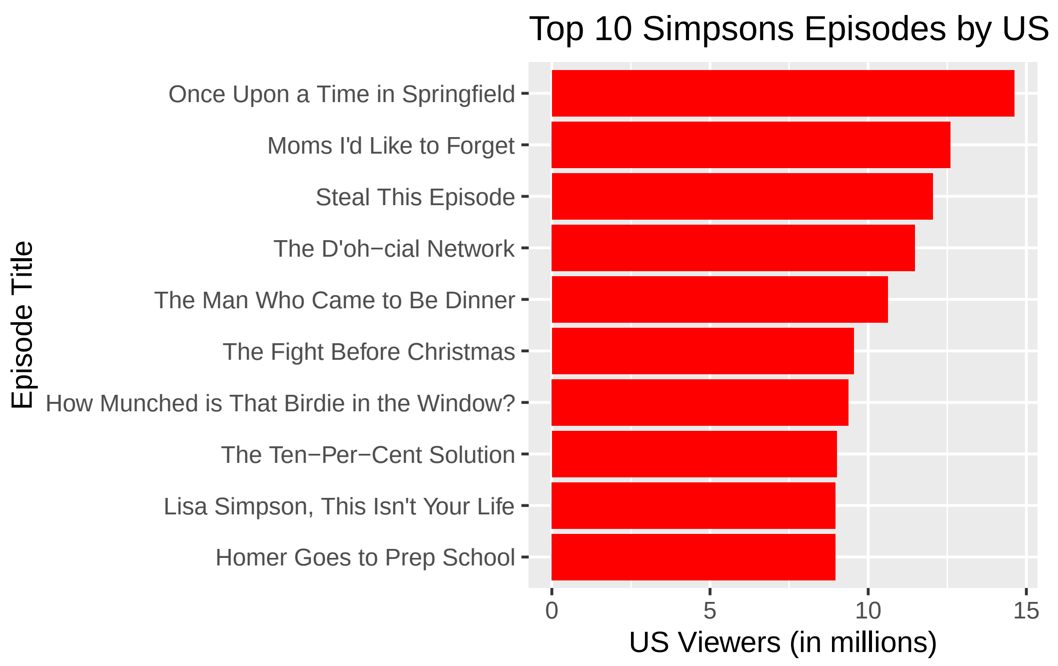
```
top_episodes_imdb <- simpsons_episodes |>
  arrange(desc(imdb_rating)) |>
  head(10)

ggplot(top_episodes_imdb, aes(x = reorder(title, imdb_rating), y = imdb_rating)) +
```

```r
  geom_col(fill = "#FFDA00") +
  coord_flip() +
  labs(
    title = "Top 10 Simpsons Episodes by IMDb Rating",
    x = "Episode Title",
    y = "IMDb Rating"
  )
```



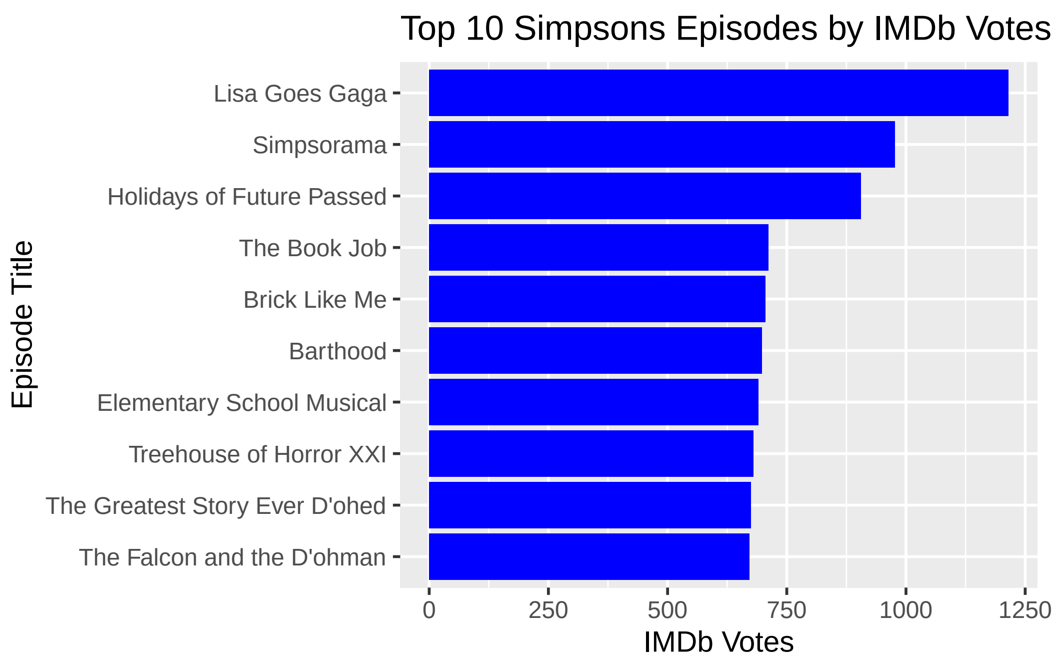
Find the top 10 episodes with the highest US viewership

```r
top_episodes_viewers <- simpsons_episodes |>
  arrange(desc(us_viewers_in_millions)) |>
  head(10)

ggplot(
  top_episodes_viewers,
  aes(x = reorder(title, us_viewers_in_millions), y = us_viewers_in_millions)
) +
  geom_col(fill = "red") +
  coord_flip() +
  labs(
    title = "Top 10 Simpsons Episodes by US Viewership (in millions)",
    x = "Episode Title",
    y = "US Viewers (in millions)"
  )
```



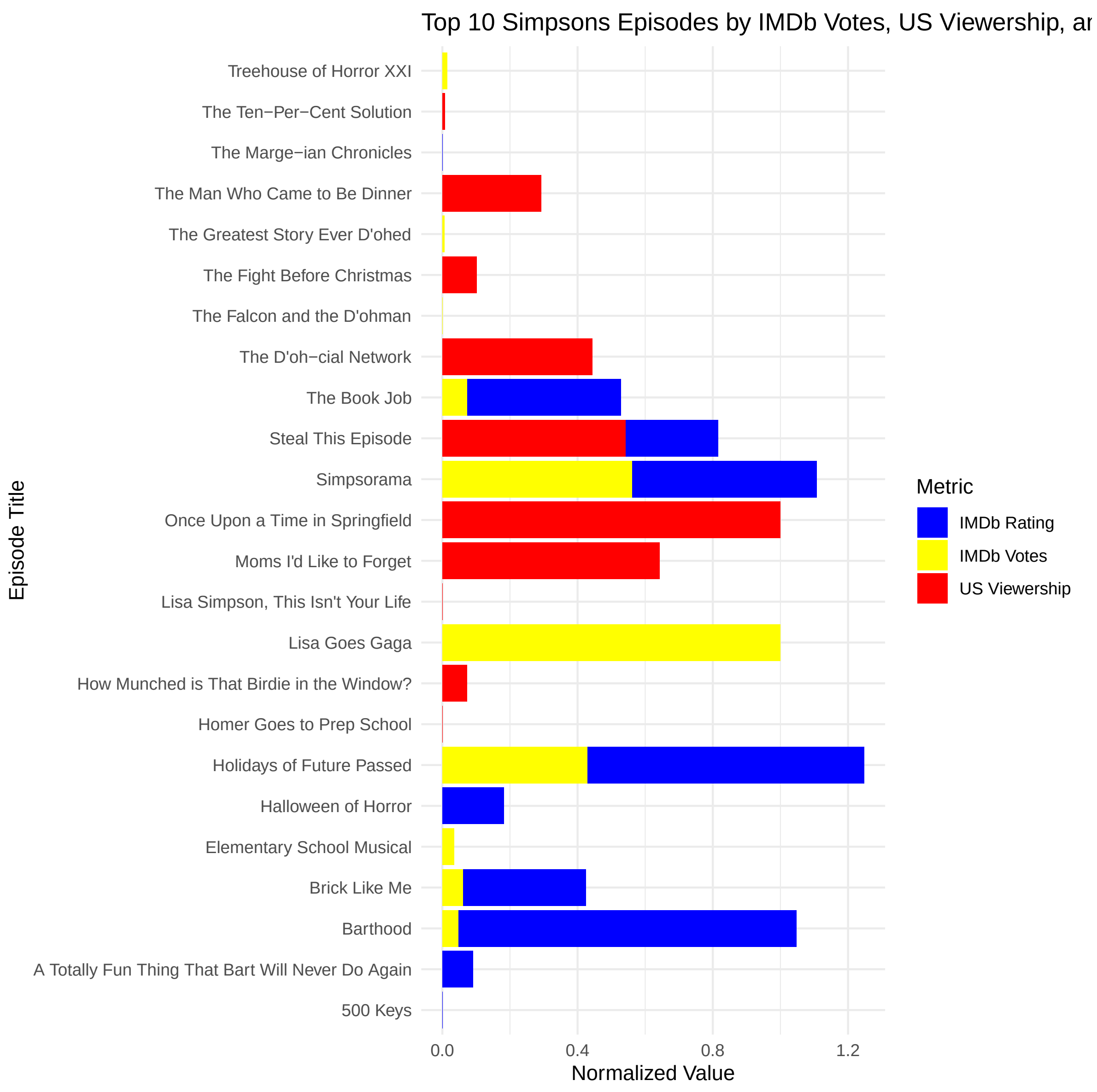
Top 10 episodes with the highest IMDb votes

```r
top_episodes_votes <- simpsons_episodes |>
  arrange(desc(imdb_votes)) |>
  head(10)

# Create a bar chart of the top episodes
ggplot(
  top_episodes_votes,
  aes(x = reorder(title, imdb_votes), y = imdb_votes)
) +
  geom_col(fill = "blue") +
  coord_flip() +
  labs(
    title = "Top 10 Simpsons Episodes by IMDb Votes",
    x = "Episode Title",
    y = "IMDb Votes"
  )
```



```
# Define Simpsons colors
simpsons_colors <- c("IMDb Votes" = "yellow", "US Viewership" = "red", "IMDb Rating" = "blue

# Function to get top 10 episodes by a given metric
get_top_10 <- function(data, metric, name) {
  data |>
    arrange(desc(.data[[metric]])) |>
    head(10) |>
    mutate(metric_name = name, value = .data[[metric]]) |>
    select(title, metric_name, value)
}

# Get top 10 episodes for each metric
top_votes <- get_top_10(simpsons_episodes, "imdb_votes", "IMDb Votes")
top_viewers <- get_top_10(simpsons_episodes, "us_viewers_in_millions", "US Viewership")
top_ratings <- get_top_10(simpsons_episodes, "imdb_rating", "IMDb Rating")
```

```r
# Combine the data
combined_data <- bind_rows(top_votes, top_viewers, top_ratings)

# Normalize the values
combined_data <- combined_data |>
  group_by(metric_name) |>
  mutate(normalized_value = (value - min(value)) / (max(value) - min(value))) |>
  ungroup()

# Create the stacked bar chart
ggplot(combined_data, aes(x = title, y = normalized_value, fill = metric_name)) +
  geom_col(position = "stack") +
  scale_fill_manual(values = simpsons_colors) +
  coord_flip() +
  labs(
    title = "Top 10 Simpsons Episodes by IMDb Votes, US Viewership, and IMDb Rating",
    x = "Episode Title",
    y = "Normalized Value",
    fill = "Metric"
  ) +
  theme_minimal()
```



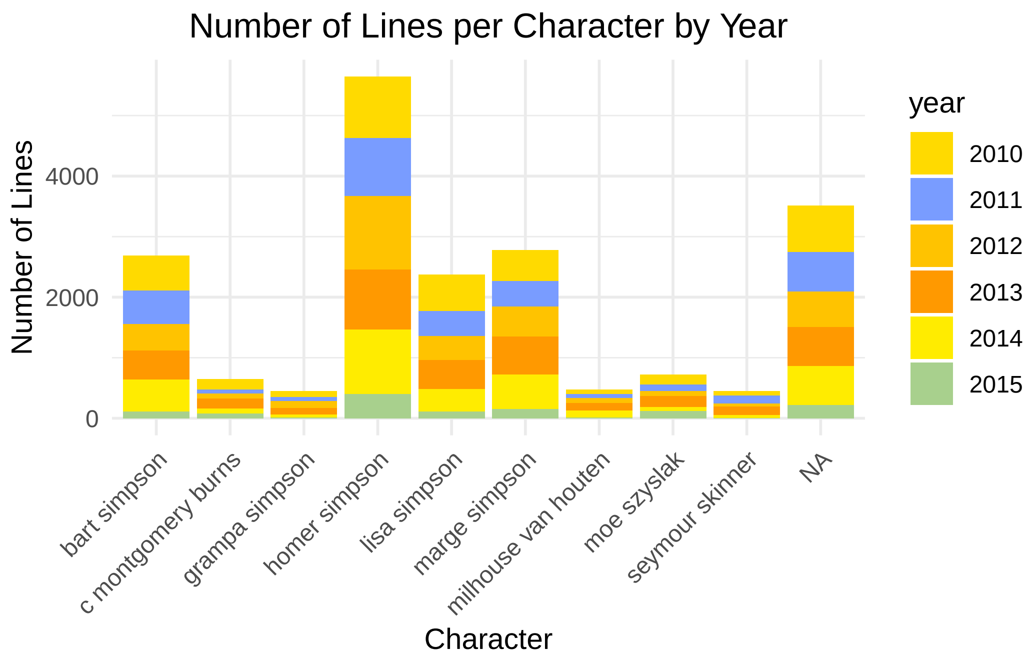
Based on this visualization, "Simpsorama" appears to be a strong contender for the "best" Simpsons episode. It has a relatively long bar and a good distribution of colors, suggesting it performed well in all three areas: critical acclaim (IMDb rating), popularity (US viewership), and engagement (IMDb votes). This suggests that "Simpsorama" is a well-rounded episode that appeals to both critics and general audiences.

Bonus

A plot in Simpsons Colours.

```r
library(tidyverse)

# Define Simpsons color palette (for seasons - approximate)
simpsons_season_colors <- c(
  "2010" = "#FFDA00",  # Homer Yellow
  "2011" = "#799CFF",  # Marge Blue
  "2012" = "#FFC300",  # Bart Orange
  "2013" = "#FF9900",  # Lisa Orange-Yellow
  "2014" = "#FFEC00",  # Maggie Yellow
  "2015" = "#A8D08D",  # "Other" Greenish
  "2016" = "#D3B043"   # A goldish color
)

# Prepare the data
# Calculate total lines per character
character_lines <- simpsons_script_lines |>
  left_join(simpsons_characters, by = c("character_id" = "id")) |>
  group_by(normalized_name) |>
  summarize(total_lines = n(), .groups = "drop")

# Get top N characters
top_characters <- character_lines |>
  slice_max(order_by = total_lines, n = 10) |>
  pull(normalized_name)

# Calculate lines per character per year for top characters
character_lines_by_season <- simpsons_script_lines |>
  left_join(simpsons_characters, by = c("character_id" = "id")) |>
  left_join(simpsons_episodes, by = c("episode_id" = "id")) |>
  filter(normalized_name %in% top_characters) |>
  group_by(normalized_name, original_air_year) |>
  summarize(line_count = n(), .groups = "drop") |>
  rename(year = original_air_year)

# Ensure all years are present in the data
character_lines_by_season <- character_lines_by_season |>
  complete(normalized_name, year, fill = list(line_count = 0))

# Convert year to factor
```

```r
character_lines_by_season <- character_lines_by_season |>
  mutate(year = factor(year))

# Create the vertical bar chart
ggplot(
  character_lines_by_season,
  aes(x = normalized_name, y = line_count, fill = year)
) +
  geom_col(position = "stack") +
  scale_fill_manual(values = simpsons_season_colors, na.value = "grey50") +
  labs(
    title = "Number of Lines per Character by Year",
    x = "Character",
    y = "Number of Lines"
  ) +
  theme_minimal() +
  theme(
    axis.text.x = element_text(
      angle = 45,
      hjust = 1
    ),
    plot.title = element_text(hjust = 0.5)
  )
```


Number of Lines per Character by Year
Number of Lines
0
2000
4000
Character
bart simpson
c montgomery burns
grampa simpson
homer simpson
lisa simpson
marge simpson
milhouse van houten
moe szyslak
seymour skinner
NA
year
2010
2011
2012
2013
2014
2015