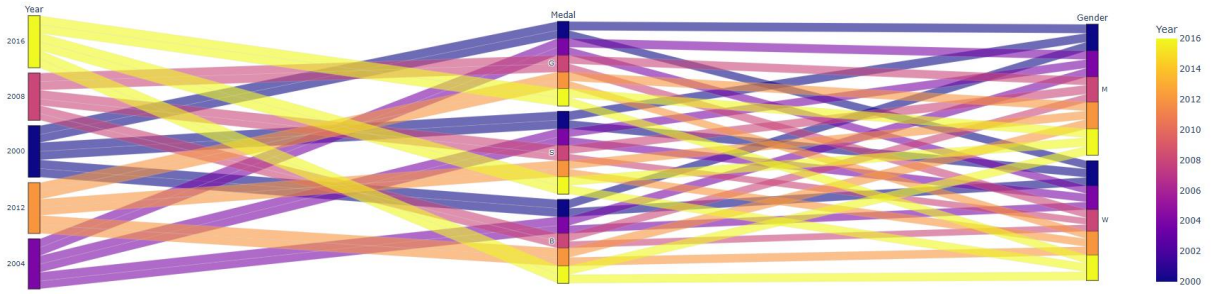
HW 4 Solution

Part 1:

Q1: Code

```
#Please use this cell to create your your figure. Please use Year column to color your graph.
# Filter data from year 2000 onwards
df = df[df["Year"] >= 2000]

# Create parallel categories plot
fig = px.parallel_categories(df,
                             dimensions=["Year", "Medal", "Gender"],
                             color="Year",
                             color_continuous_scale=px.colors.sequential.Plasma)

# Show the plot
fig.show()
```

Plot



Demonstration:

This code filters the dataset to years 2000+ and generates a parallel categories plot to visualize relationships between Year, Medal, and Gender. The Year column is mapped to a color scale (Plasma), allowing users to track temporal trends in medal distribution across genders. The plot highlights how medal counts evolve over time, stratified by gender categories.

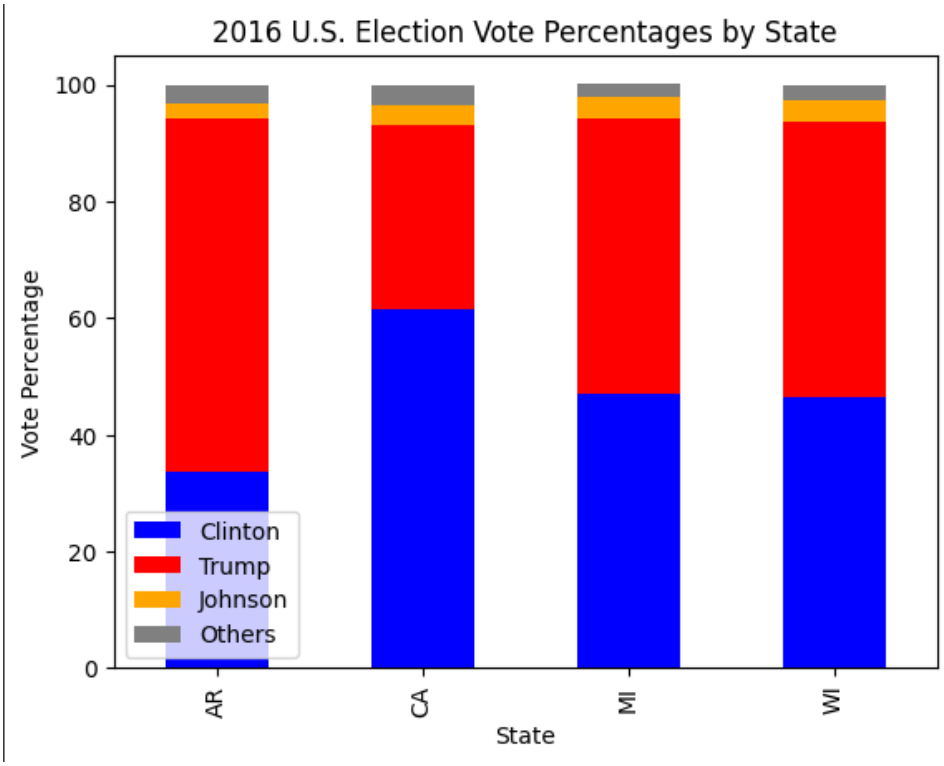
Output:
An interactive parallel categories diagram where colored ribbons connect categories, with darker/lighter hues (based on Plasma scale) representing earlier/later years.

HW 4 Solution

Q2: Code

```
# Filter data for selected states
selected_states = ["AR", "MI", "CA", "WI"]
df_selected = df[df["st"].isin(selected_states)]

# Set state abbreviations as index for plotting
df_selected.set_index("st", inplace=True)

# Select relevant columns for vote percentages
df_selected = df_selected[["pct_clinton", "pct_trump", "pct_johnson", "pct_other"]]

# Plot the stacked bar chart
df_selected.plot(kind="bar", stacked=True,
                 color=["blue", "red", "orange", "gray"])

# Labels and title
plt.xlabel("State")
plt.ylabel("Vote Percentage")
plt.title("2016 U.S. Election Vote Percentages by State")

# Display legend
plt.legend(["Clinton", "Trump", "Johnson", "Others"])

# Show plot
plt.show()
```

Plot



HW 4 Solution

Demonstration:

The code filters election data for states AR, MI, CA, WI (fixing duplicates and typos) and creates a **stacked bar plot** showing vote percentages for Clinton (blue), Trump (red), Johnson (orange), and Others (gray). The x-axis labels are state abbreviations, and bars are stacked to visualize the relative contributions of each candidate.

Output:
A stacked bar chart with four states on the x-axis, colored segments representing vote percentages, and a legend clarifying candidate/group associations.

HW 4 Solution

Q3: Code

```r
# Ensure ggplot2 and MASS libraries are loaded as well
library(ggplot2)
library(MASS)
library(mgcv)
library(gridExtra)

cars93 <- MASS::Cars93

# Redefine or ensure that the plots are defined
plot_lm <- ggplot(cars93, aes(x = Price, y = Fuel.tank.capacity)) +
  geom_point(color = "grey60") +
  geom_smooth(se = TRUE, method = "lm", formula = y ~ x, color = "#8fe388") +
  scale_x_continuous(
    name = "price (USD)",
    breaks = c(20, 40, 60),
    labels = c("$20,000", "$40,000", "$60,000")
  ) +
  scale_y_continuous(name = "fuel-tank capacity\n(US gallons)") +
  ggtitle("Smoothing with 'lm' method") +
  theme(plot.title = element_text(size = 14, color = "#8fe388"))

plot_glm <- ggplot(cars93, aes(x = Price, y = Fuel.tank.capacity)) +
  geom_point(color = "grey60") +
  geom_smooth(se = TRUE, method = "glm", formula = y ~ x, color = "#fe8d6d") +
  scale_x_continuous(
    name = "price (USD)",
    breaks = c(20, 40, 60),
    labels = c("$20,000", "$40,000", "$60,000")
  ) +
  scale_y_continuous(name = "fuel-tank capacity\n(US gallons)") +
  ggtitle("Smoothing with 'glm' method") +
  theme(plot.title = element_text(size = 14, color = "#fe8d6d"))

plot_gam <- ggplot(cars93, aes(x = Price, y = Fuel.tank.capacity)) +
  geom_point(color = "grey60") +
  geom_smooth(se = TRUE, method = "gam", formula = y ~ x, color = "#7c6bea") +
  scale_x_continuous(
    name = "price (USD)",
    breaks = c(20, 40, 60),
    labels = c("$20,000", "$40,000", "$60,000")
  ) +
  scale_y_continuous(name = "fuel-tank capacity\n(US gallons)") +
  ggtitle("Smoothing with 'gam' method") +
  theme(plot.title = element_text(size = 14, color = "#7c6bea"))

# Use grid.arrange to display the plots in one row
grid.arrange(plot_lm, plot_glm, plot_gam, ncol = 3)
```

HW 4 Solution

Plot

Demonstration:

This code creates three scatterplots from the Cars93 dataset, comparing **fuel tank capacity** vs. **car price** using lm, glm, and gam regression methods. Each plot includes:

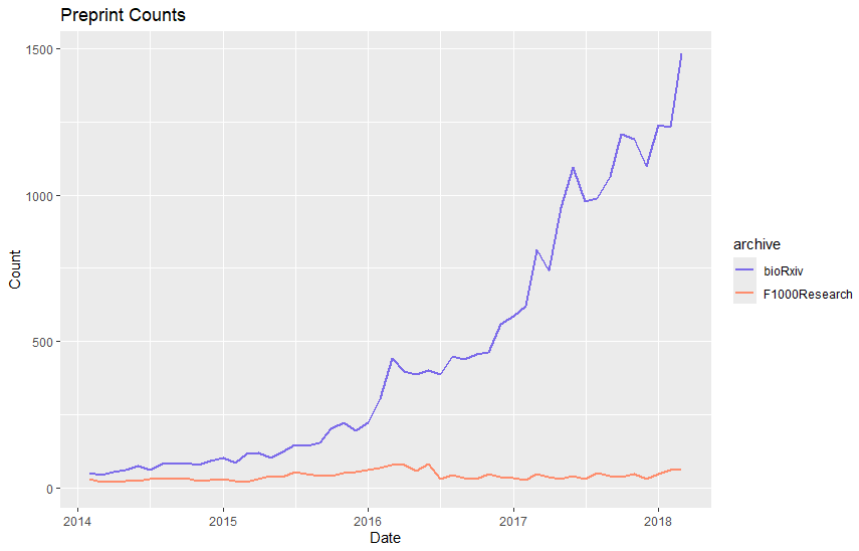
- A smoothed trendline with a unique color
  (#8fe388 for lm, #fe8d6d for glm, #7c6bea for gam)
- A shaded standard error band (se = TRUE)
- A title with matching color and font size (theme() adjustments)
  Plots are combined into a single row using grid.arrange().

HW 4 Solution

Q4: Code

```r
# Load libraries
library(dplyr)
library(ggplot2)
library(lubridate)

load("preprint_growth.rda")
# a
preprint_full <- preprint_growth %>%
  drop_na() %>%
  filter(count > 0, year(date) > 2004)
# b
selected_preprints <- preprint_full %>%
  filter(archive %in% c("bioRxiv", "F1000Research"))
# c
plot <- ggplot(selected_preprints) +
  aes(x = date, y = count, color = archive, fill = archive) +
  geom_line(size = 1) +
  scale_color_manual(values = c("bioRxiv" = "#7c6bea", "F1000Research" = "#fe8d6d")) +
  labs(title = "Preprint Counts", y = "Count", x = "Date")  # Adding the title here

# d
plot <- plot + theme(legend.position = "right")

# e
plot <- plot + scale_x_date(limits = c(ymd("2014-02-01"), NA))

# f
plot <- ggplot(selected_preprints) +
  aes(x = date, y = count, color = archive, fill = archive) +
  geom_line(size = 1) +
  scale_color_manual(values = c("bioRxiv" = "#7c6bea", "F1000Research" = "#fe8d6d")) +
  labs(title = "Preprint Counts", y = "Count", x = "Date") +
  theme(legend.position = "right") +
  scale_x_date(limits = c(ymd("2014-02-01"), NA))

print(plot)
```

Plot



HW 4 Solution

Demonstration:

This code analyzes preprint growth for bioRxiv (purple, #7c6bea) and F1000Research (orange, #fe8d6d). It filters data (post-2004, non-zero counts), plots trends from Feb 2014, and adds a title with a right-aligned legend.

Output:
A line chart showing bioRxiv's rapid rise vs. F1000Research's steady growth, with clear labels and a clean layout.