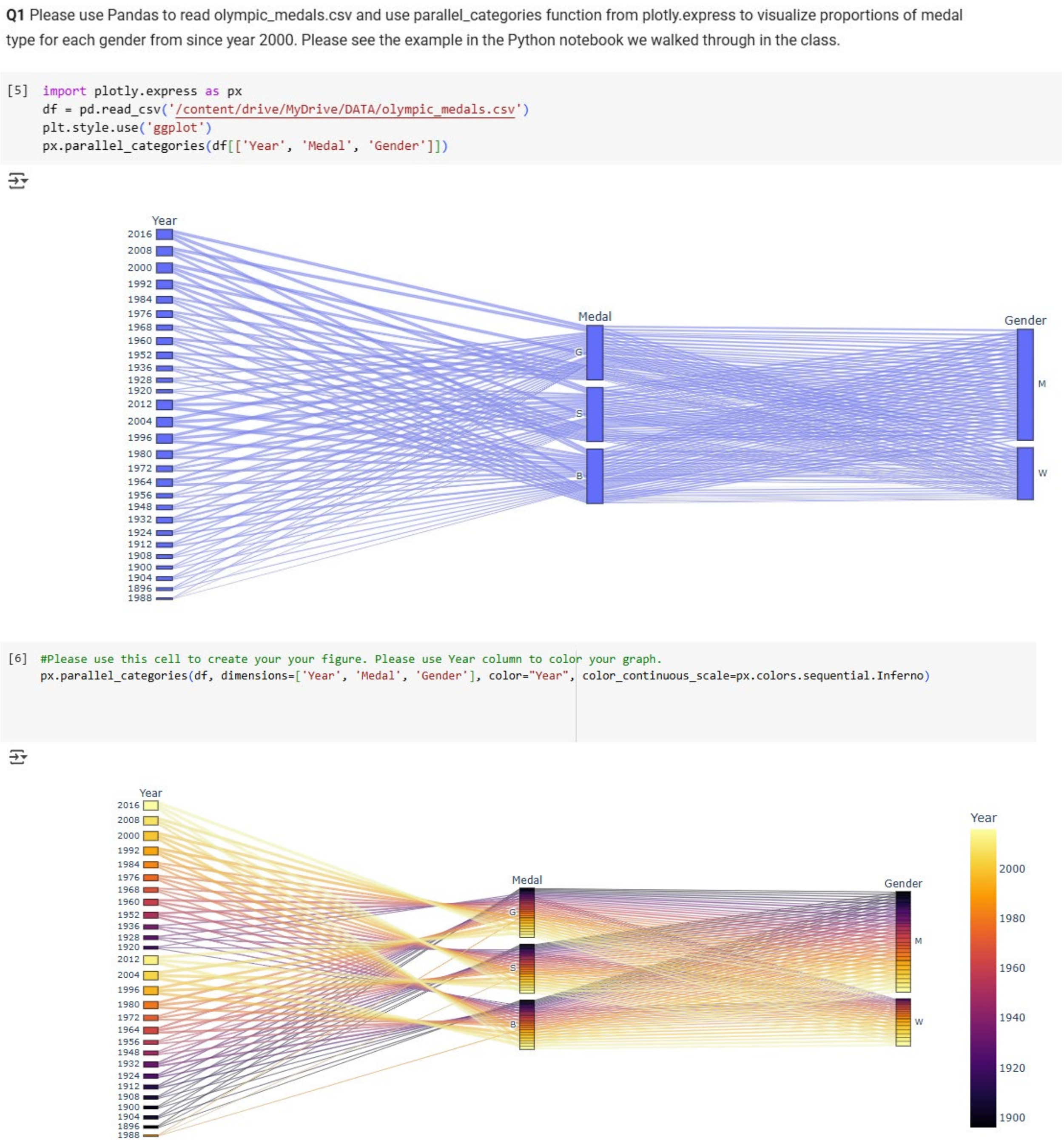
Posted on GitHub at https://github.com/1fastgranada/CSC302_HW4

1. Please use the Python notebook at https://colab.research.google.com/drive/1qnI9F4khh9EzhBLkjBl7_lkYcdppqoW2 to read olympic_medals.csv and use parallel_categories function from plotly.express to visualize proportions of medal type for each gender from since year 2000. Please see the example in the Python notebook we walked through in the class.

Please see Python notebook "HW4_Part1_ARoss.ipynb" on GitHub or linked here. Results have also been posted below for reference:



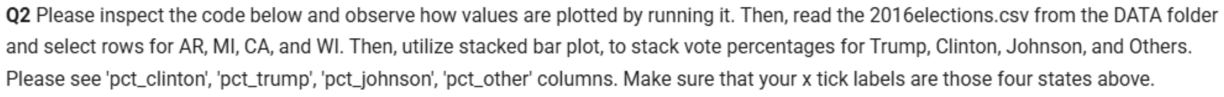
Q1 Please use Pandas to read olympic_medals.csv and use parallel_categories function from plotly.express to visualize proportions of medal type for each gender from since year 2000. Please see the example in the Python notebook we walked through in the class.

```
[5]  import plotly.express as px
     df = pd.read_csv('/content/drive/MyDrive/DATA/olympic_medals.csv')
     plt.style.use('ggplot')
     px.parallel_categories(df[['Year', 'Medal', 'Gender']])
```

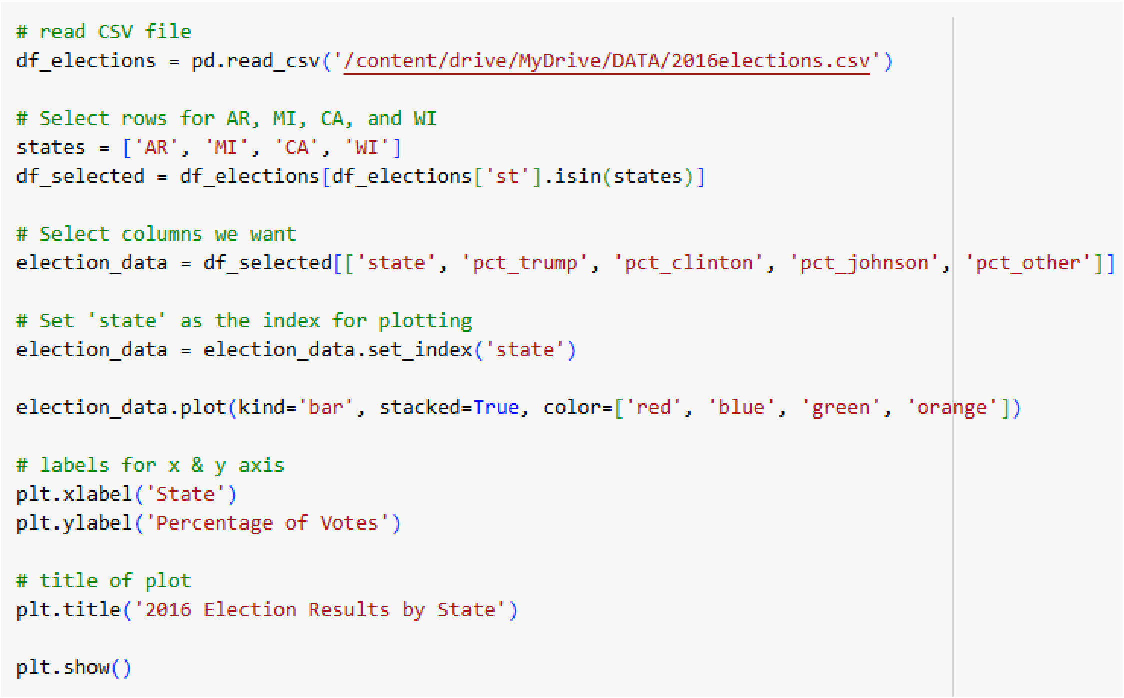
```
[6]  #Please use this cell to create your your figure. Please use Year column to color your graph.
     px.parallel_categories(df, dimensions=['Year', 'Medal', 'Gender'], color="Year", color_continuous_scale=px.colors.sequential.Inferno)
```

Posted on GitHub at https://github.com/1fastgranada/CSC302_HW4

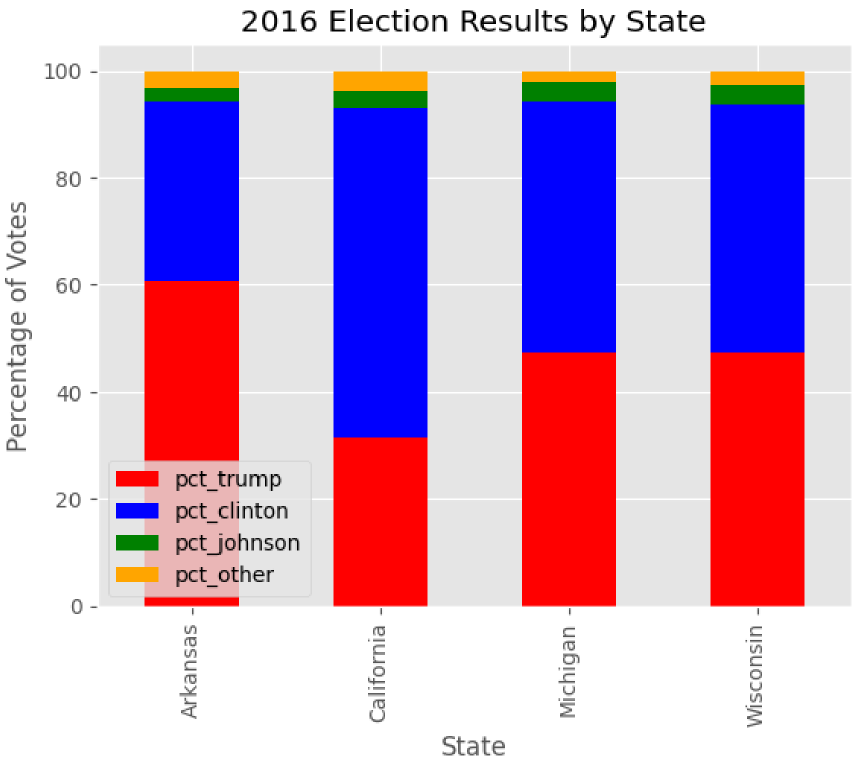
2. Please see the same notebook in the question above.



Q2 Please inspect the code below and observe how values are plotted by running it. Then, read the 2016elections.csv from the DATA folder and select rows for AR, MI, CA, and WI. Then, utilize stacked bar plot, to stack vote percentages for Trump, Clinton, Johnson, and Others. Please see 'pct_clinton', 'pct_trump', 'pct_johnson', 'pct_other' columns. Make sure that your x tick labels are those four states above.



```
# read CSV file
df_elections = pd.read_csv('/content/drive/MyDrive/DATA/2016elections.csv')

# Select rows for AR, MI, CA, and WI
states = ['AR', 'MI', 'CA', 'WI']
df_selected = df_elections[df_elections['st'].isin(states)]

# Select columns we want
election_data = df_selected[['state', 'pct_trump', 'pct_clinton', 'pct_johnson', 'pct_other']]

# Set 'state' as the index for plotting
election_data = election_data.set_index('state')

election_data.plot(kind='bar', stacked=True, color=['red', 'blue', 'green', 'orange'])

# labels for x & y axis
plt.xlabel('State')
plt.ylabel('Percentage of Votes')

# title of plot
plt.title('2016 Election Results by State')

plt.show()
```

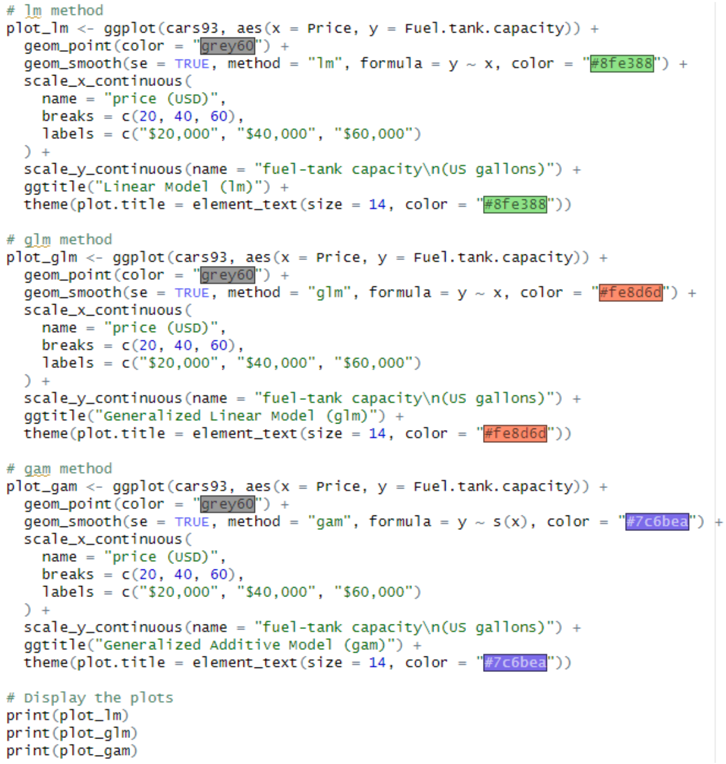


Posted on GitHub at https://github.com/1fastgranada/CSC302_HW4

3. Please run the following code and observe how the parameters are used in the resulting figure. Then perform the steps in the following bullet points. Notice that we are getting the data frame, Cars93, from MASS module. So, we're not reading a dataset for this question.

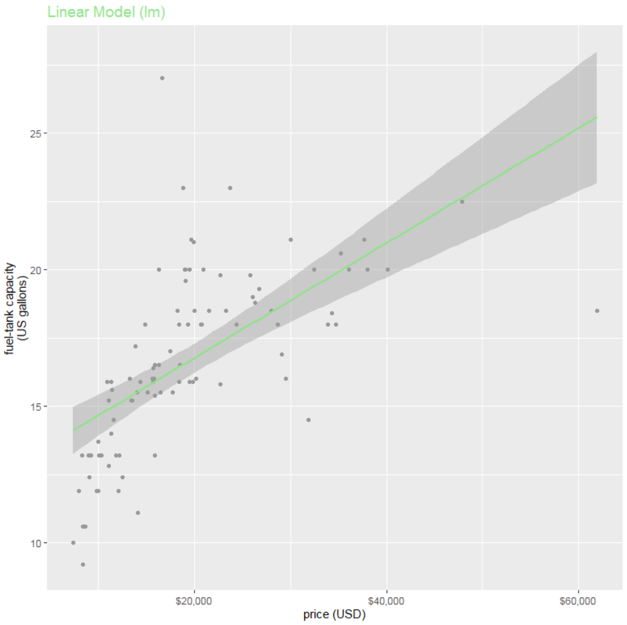
```
cars93 <- MASS::Cars93
ggplot(cars93, aes(x = Price, y = Fuel.tank.capacity)) +
  geom_point(color = "grey60") +
  geom_smooth(se = FALSE, method = "loess", formula = y ~ x, color = "#0072B2") +
  scale_x_continuous(
    name = "price (USD)",
    breaks = c(20, 40, 60),
    labels = c("$20,000", "$40,000", "$60,000")
  ) +
  scale_y_continuous(name = "fuel-tank capacity\n(US gallons)")
```

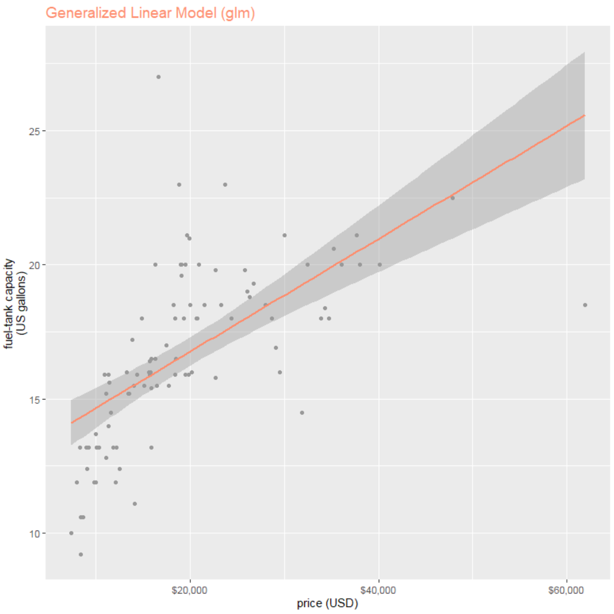
   a. Use "lm", "glm", "gam" methods in the **geom_smooth()** function to create three figures.
   b. Set the **se** parameter to TRUE to show the standard error (shaded area around the fitted line)
   c. For every method above change the color of the line with the following color codes: #8fe388, #fe8d6d, #7c6bea
   d. Please search for the method to add a title to your ggplot figure and add titles for each figure to indicate the method that you used for smoothing.
   e. Please search for the **theme()** function for ggplot and change the font size of the titles to 14 and match their colors with the line colors you used above.

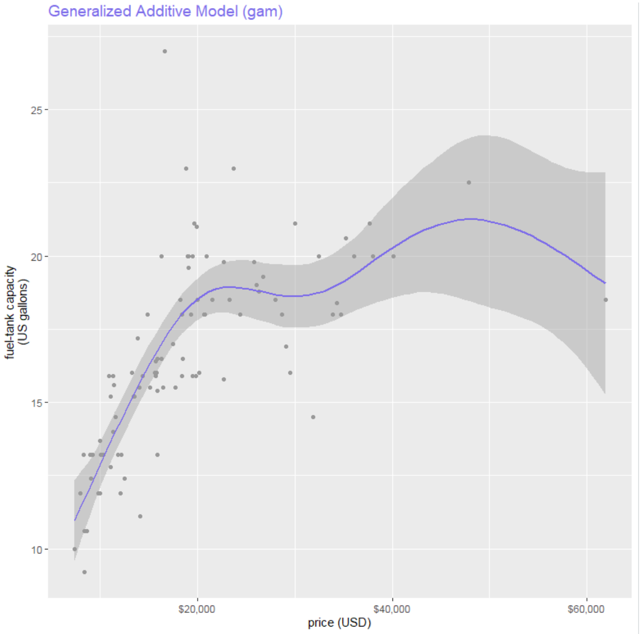


```
# lm method
plot_lm <- ggplot(cars93, aes(x = Price, y = Fuel.tank.capacity)) +
  geom_point(color = "grey60") +
  geom_smooth(se = TRUE, method = "lm", formula = y ~ x, color = "#8fe388") +
  scale_x_continuous(
    name = "price (USD)",
    breaks = c(20, 40, 60),
    labels = c("$20,000", "$40,000",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

# glm method
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("Generalized Linear Model (glm)") +
  theme(plot.title = element_text(size = 14, color = "#fe8d6d"))

# gam method
plot_gam <- ggplot(cars93, aes(x = Price, y = Fuel.tank.capacity)) +
  geom_point(color = "grey60") +
  geom_smooth(se = TRUE, method = "gam", formula = y ~ s(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("Generalized Additive Model (gam)") +
  theme(plot.title = element_text(size = 14, color = "#7c6bea"))

# Display the plots
print(plot_lm)
print(plot_glm)
print(plot_gam)
```

Posted on GitHub at https://github.com/1fastgranada/CSC302_HW4


Linear Model (lm)

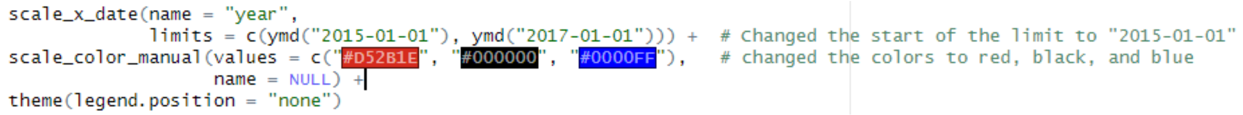

Generalized Linear Model (glm)


Generalized Additive Model (gam)

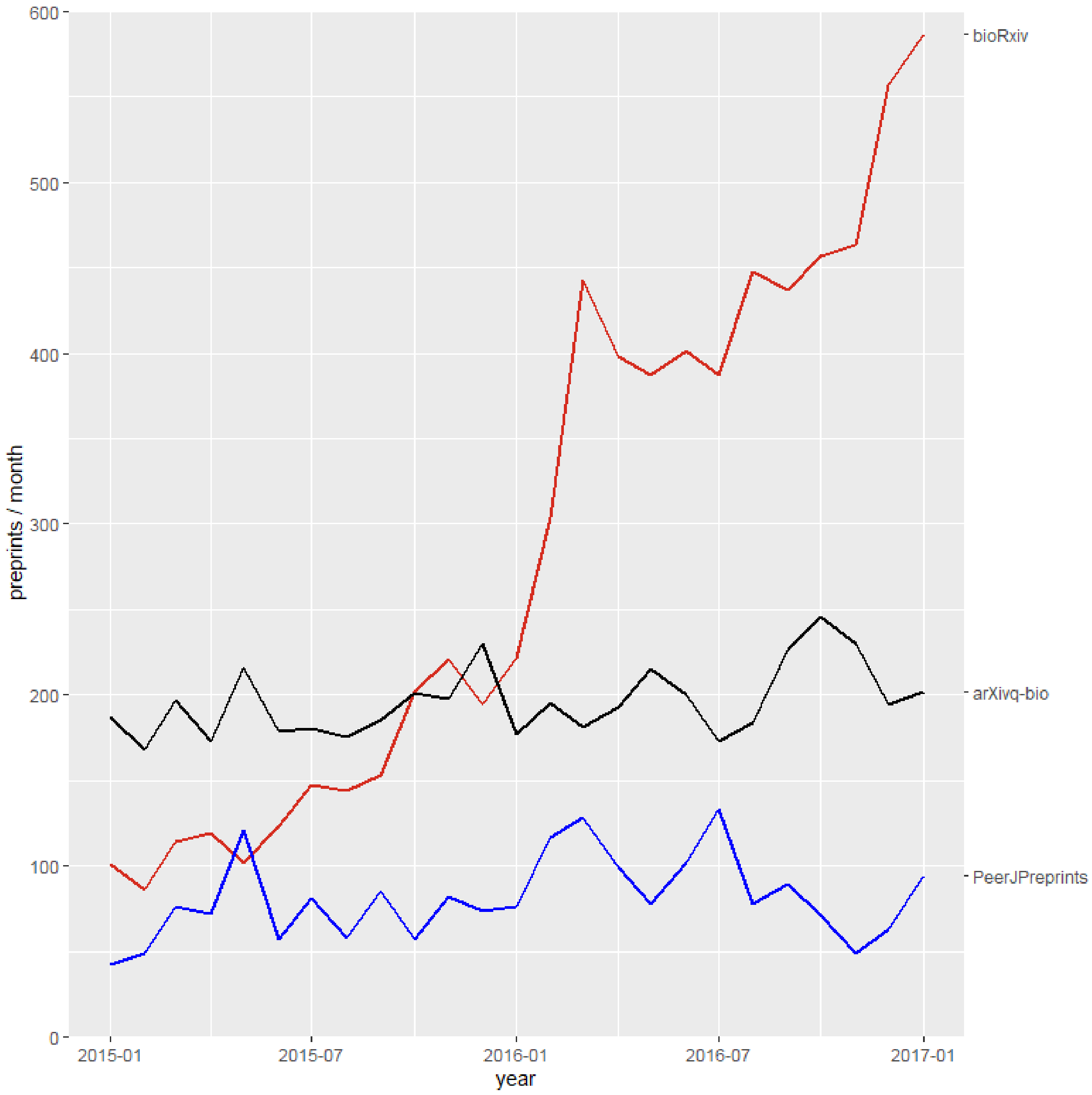
Posted on GitHub at https://github.com/1fastgranada/CSC302_HW4

4. Please inspect the following code which can be also found in TimeSeries_Trends.R and try to run how it generates three time series in a single plot. Then, modify the start date and the manual coloring as you want to get a different version of the chart. Please indicate what you changed and submit the figure you created as a response to this question.

I changed the follow 2 lines (commented) to get the requested results:



```
scale_x_date(name = "year",
             limits = c(ymd("2015-01-01"), ymd("2017-01-01"))) +  # Changed the start of the limit to "2015-01-01"
scale_color_manual(values = c("#D52B1E", "#000000", "#0000FF"),   # changed the colors to red, black, and blue
                   name = NULL) +
theme(legend.position = "none")
```


preprints / month
year
bioRxiv
arXivq-bio
PeerJPreprints