Intermediate and Advanced Graphs in R

Welcome!

Jennifer Lin

PS 490: R Workshop

2022-02-25

Assumptions

1. You know how to install and load `ggplot`
2. You know the basic structure that goes into making a minimally acceptable `ggplot` graph
3. You are ready to have fun!

Last Quarter...

In the introductory version of this class, you might recall

```
ggplot(data layer)+
  graph layer +
  label layer +
  scale layer +
  theme layer +
  others
```

Today, we complicate!

…No I am just kidding…

BUT we get more into the weeds on plotting. We will consider…

1. Become familiar with different ways of integrating factor variables in graphs in R.
2. Become more comfortable in `ggplot2` by creating more complex graphs, including line plots and stacked bar charts.
3. Explore different ways to be creative with coding tools and streamline R code for making graphs

Bear in Mind...

Today's workshop is *still* not exhaustive on the many things you can do with `ggplot`. We will not cover...

1. Maps in R
2. Animated and Interactive plots
3. Social Network illustrations
4. Graphs from text analysis results

This list can go on...

A Review of the Basics

The Data -- ANES

- I cleaned up the ANES 2020 data and you can choose your own adventure.
- See handout for the specific variables, their definitions and the original variable name in the ANES data file

```
ANES <- read.csv("ANES_adv_ggplot.csv")
```

The Running Example

The ANES asks respondents a battery of items each year to assess where they would place themselves on a left-right scale for various issue positions. In my examples, I consider responses to the following variables:

- Spending and Services
- Defense Spending
- Government versus Private Medical Insurance
- Guaranteed Job/Income
- Environment-Business Tradeoff
- Government Assistance to Blacks

The Graphs

Recall that ggplot2 uses many rather intuitive names for graphs

Operator	Description
geom_line()	line graph
geom_point()	scatterplot
geom_bar()	bar plot
geom_histogram()	histogram
geom_boxplot()	boxplot
geom_violin()	violin plot

Theme Functions -- A Quick Digression

What? Why?

Recall the `theme_*()` class of functions that you can add to any `ggplot2` object to make your graph look nice.

Then recall all the beautiful `plot.*()`, `axis.*()` and `legend.*()` things you can add to the broader `theme()` function to further customize your plot.

Finally, remember that R is an object oriented programming language

We can leverage all of this to create a convienent theme object.

Theme Functions -- A Quick Digression

Writing the Function

You can store your favorite theme presets in a function.

```
theme_DEMO <- function(){
  theme_classic()+
  theme(
    plot.title          = element_text(hjust = 0.5, size = 20),
    plot.subtitle       = element_text(colour="black", face = "bold"),
    ...
  )
}
```

Now, you can construct plots with a theme like so:

```
ggplot()+
  ...
  theme_DEMO()
```

Theme Functions -- A Quick Digression

My Theme

```r
theme_rworkshop <- function() {
  theme_bw()+
  theme(
    plot.title         = element_text(hjust = 0.5, size = 20, colo
    plot.subtitle      = element_text(hjust = 0.5, size = 16, colo
    legend.title       = element_text(hjust = 0.5, size = 14, colo
    plot.caption       = element_text(size = 10, colour="black"),
    axis.title         = element_text(size = 14, colour="black"),
    axis.text.x        = element_text(size = 12, colour="black", a
    axis.text.y        = element_text(size = 12, colour="black"),
    legend.position    = 'bottom',
    legend.direction   = "horizontal",
    legend.text        = element_text(size = 12, colour="black")
  )
}
```

Exercise 1

1. Look through the ANES data. PICK
   a. ONE FACTOR variable to use for your graph as a grouping variable
   b. ONE numeric variable for which to focus your graph
   c. ONE other variable -- your pick -- that would go well with the variable you picked in (a) and (b)
2. Generate a theme function for use for the rest of this workshop.
3. Generate a graph with everything you know about `ggplot`. Include proper labels, titles, colors, and themes (ideally using the function from (2)). Save as a PDF.

```r
ANES %>%
  filter(!is.na(PARTY)) %>%
  ggplot(aes(x = FT_rural, y = FT_BLM, color = PARTY))+
    geom_point(position = position_jitter(1, 1), alpha = .5)+
  xlab("Feelings towards Rural Americans")+
  ylab("Feelings towards the BLM Movement")+
  labs(
    color = "Political Party",
    title = "Feelings towards Rural Americans and
    the BLM Movement",
    subtitle = "Analysis from the American National Elections Stuc
    caption = "Data: ANES 2020
    Author: Jennifer Lin"
  )+
  scale_color_manual(
    name = "Political\nParty",
    breaks = c("Democrat",  "Republican", "Independent"),
    values = c("Democrat" = "#3182bd", "Republican" = "#de2d26", '
  )+
  scale_x_continuous(
    breaks = seq(0, 100, 10),
    limits = c(0, 100)
  )+
  scale_y_continuous(
    breaks = seq(0, 100, 10),
    limits = c(0, 100)
  )+
```

Stacked Bar Graphs

First, Why?

- Regular bar graphs often have one x- and one y-axis
- Y-axis generally displays the number of responses for each category on the x-axis
- Stacked bar graphs give us a new layer of data that we can use to understand responses.
- We can graph the number of respondents by party for the ANES survey and plot the number of people from each party who responds on each level of the Services and Spending 7-point scale

Creating a Stacked Bar Chart

Making the Data Table

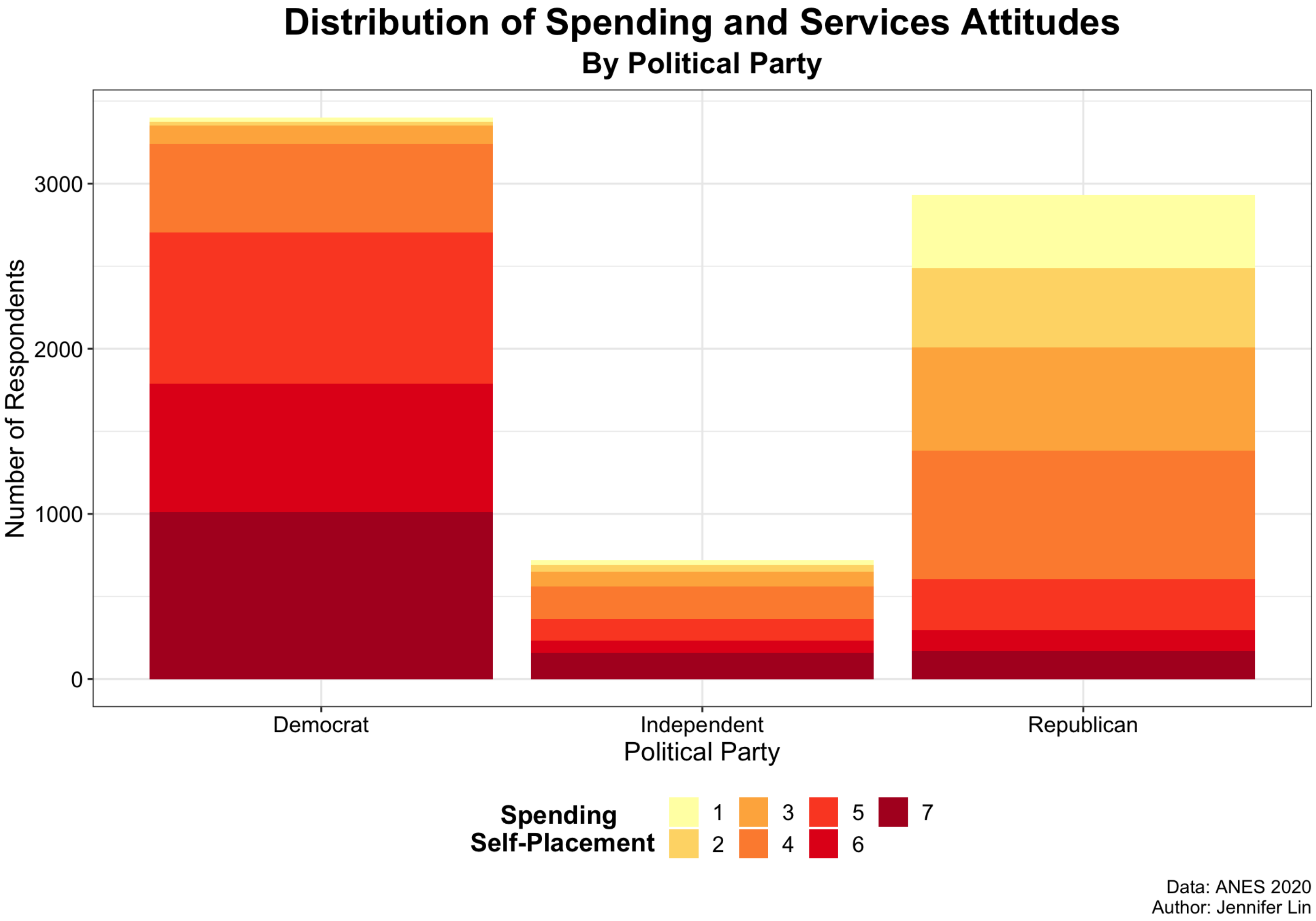
Like a bar graph, we will need to generate a table of summary statistics for the variables that we are interested in plotting

```r
Spending <- ANES %>%
  group_by(PARTY, self_spending) %>%
  summarise(
    n = n(),
    .groups = 'keep'
    ) %>%
  filter(!is.na(PARTY) & self_spending != "NaN")
```

Creating a Stacked Bar Chart

Making the Graph Itself

```
ggplot(Spending,
       aes(x = PARTY, y = n, fill = factor(self_spending)))+
  geom_bar(stat = 'identity', position=position_stack())+
  scale_fill_brewer(palette = 'YlOrRd')+
  labs(
    fill = "Spending \nSelf-Placement",
    title = "Distribution of Spending and Services Attitudes",
    subtitle = "By Political Party",
    caption = "Data: ANES 2020
    Author: Jennifer Lin"
  )+
  xlab("Political Party")+
  ylab("Number of Respondents")+
  theme_rworkshop()
```



Distribution of Spending and Services Attitudes

By Political Party

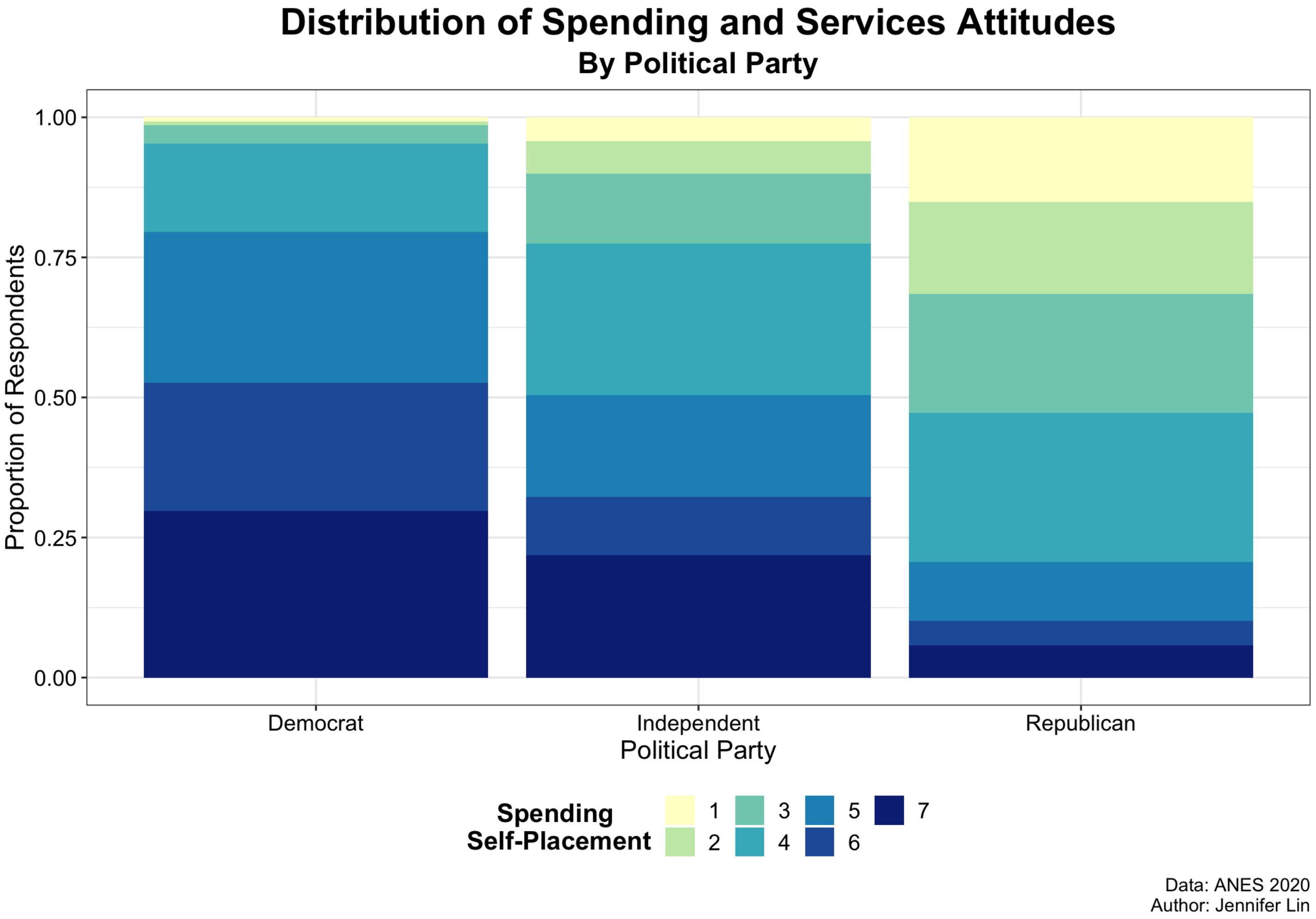
Data: ANES 2020
Author: Jennifer Lin

Proportional Representations

In the previous example, we plotted the raw number of respondents per party that responded on each level of the Spending question. Now, what if we want proportions?

```
ggplot(Spending,
       aes(x = PARTY, y = n, fill = factor(self_spending)))+
  geom_bar(stat = 'identity', position=position_fill())+
  scale_fill_brewer(palette = 'YlGnBu')+
  labs(
    fill = "Spending \nSelf-Placement",
    title = "Distribution of Spending and Services Attitudes",
    subtitle = "By Political Party",
    caption = "Data: ANES 2020
    Author: Jennifer Lin"
  )+
  xlab("Political Party")+
  ylab("Proportion of Respondents")+
  theme_rworkshop()
```

What changed in the code?



Distribution of Spending and Services Attitudes

By Political Party

Data: ANES 2020
Author: Jennifer Lin

Exercise 2

From your variables in Exercise 1, generate a stacked bar chart with any combination of the three variables such that it tells a story about the relationship of the variables. Include proper labels, titles, colors and themes

Line Plots with Factors and Facets

Generating Line Plots

```
Spending_Line <- ANES %>%
  mutate(
    PARTY7          = case_when(
      pid7 == 1 ~ "Strong Democrat",
      pid7 == 2 ~ "Democrat",
      pid7 == 3 ~ "Lean Democrat",
      pid7 == 4 ~ "Independent",
      pid7 == 5 ~ "Lean Republican",
      pid7 == 6 ~ "Republican",
      pid7 == 7 ~ "Strong Republican"
    ),
    PARTY7          = factor(
      PARTY7,
      levels = c(
        "Strong Democrat",
        "Democrat","Lean Democrat",
        "Independent","Lean Republican",
        "Republican","Strong Republican"
      ),
      ordered = TRUE
    )
  )
```

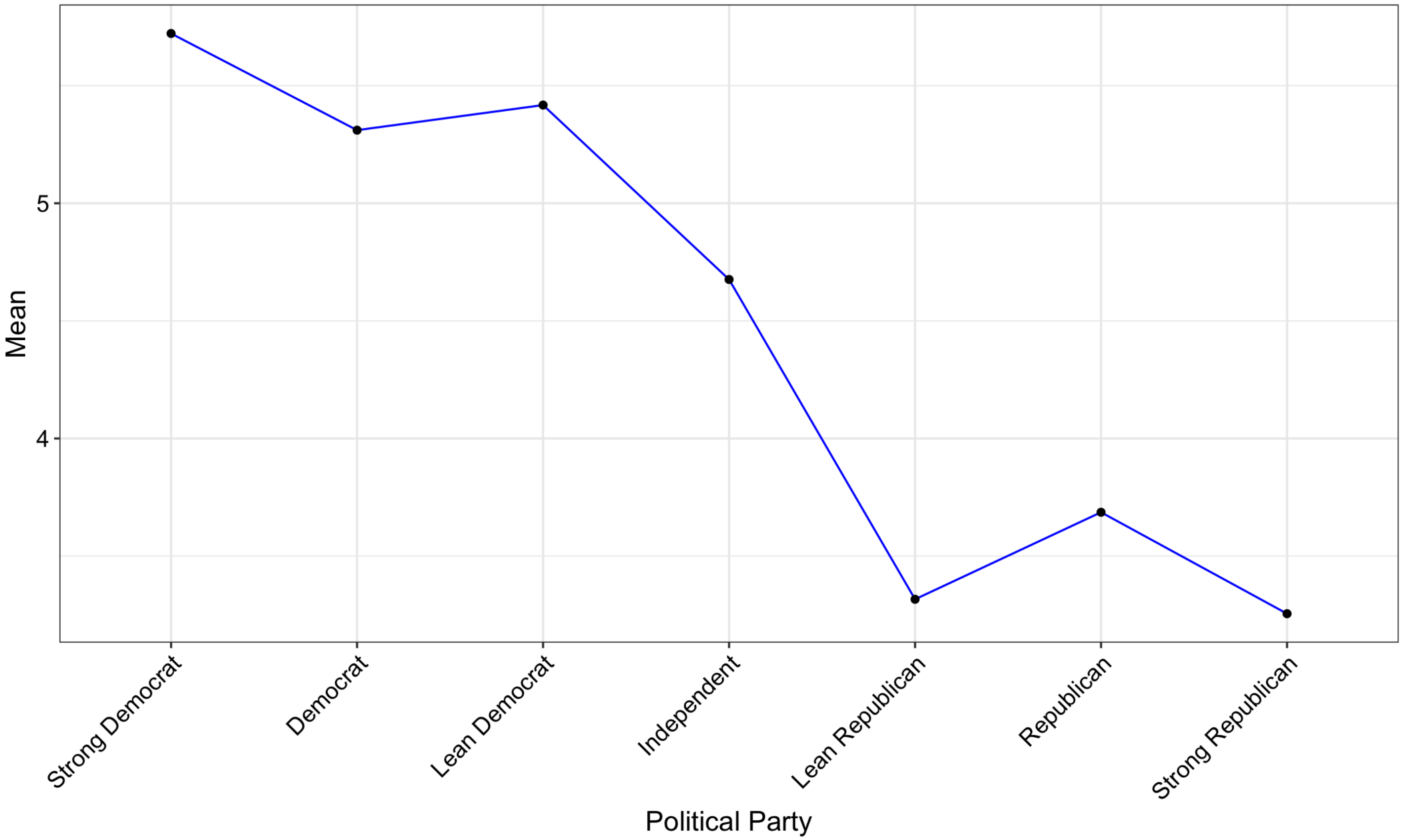
```r
Spending_Line <- Spending_Line %>%
  group_by(PARTY7) %>%
  summarise(
    mean = mean(self_spending, na.rm = TRUE),
    .groups = 'keep'
  ) %>%
  filter(!is.na(PARTY7))
```

PARTY7	mean
Strong Democrat	5.721714
Democrat	5.310705
Lean Democrat	5.417421
Independent	4.675900
Lean Republican	3.316340
Republican	3.686303

```
ggplot(Spending_Line,
       aes(x = PARTY7, y = mean, group = 1))+
  geom_line(color = "blue")+
  geom_point(color = "black")+
  labs(
    title = "Distribution of Spending and Services Attitudes",
    subtitle = "By Political Party",
    caption = "Data: ANES 2020
    Author: Jennifer Lin"
  )+
  xlab("Political Party")+
  ylab("Mean")+
  theme_rworkshop()+
  theme(
    axis.text.x     = element_text(angle = 45, hjust = 1)
  )
```

Distribution of Spending and Services Attitudes

By Political Party



Data: ANES 2020
Author: Jennifer Lin

Creating Line Plots with Facets

```
Personal_Placement <- ANES %>%
  mutate(
    PARTY7          = case_when(
      pid7 == 1 ~ "Strong Democrat",
      pid7 == 2 ~ "Democrat",
      pid7 == 3 ~ "Lean Democrat",
      pid7 == 4 ~ "Independent",
      pid7 == 5 ~ "Lean Republican",
      pid7 == 6 ~ "Republican",
      pid7 == 7 ~ "Strong Republican"
    ),
    PARTY7          = factor(
      PARTY7,
      levels = c(
        "Strong Democrat",
        "Democrat","Lean Democrat",
        "Independent","Lean Republican",
        "Republican","Strong Republican"
      ),
      ordered = TRUE
    )
  )
```

```r
Personal_Placement <- Personal_Placement %>%
    group_by(PARTY7) %>%
  summarise(
    spending   = mean(self_spending, na.rm = TRUE),
    defense    = mean(self_defense, na.rm = TRUE),
    medical    = mean(self_medical, na.rm = TRUE),
    job        = mean(self_job, na.rm = TRUE),
    blacks     = mean(self_blacks, na.rm = TRUE),
    business   = mean(self_business, na.rm = TRUE),
    .groups    = 'keep'
  ) %>%
  filter(!is.na(PARTY7))
```

PARTY7	spending	defense	medical	job	blacks	business
Strong Democrat	5.721714	3.643570	2.512263	2.889393	2.391257	1.808149
Democrat	5.310705	3.618858	2.767767	3.374680	2.981061	2.240320
Lean Democrat	5.417421	3.287383	2.478652	3.160183	2.648678	1.906321
Independent	4.675900	4.060140	3.491391	4.006658	3.885375	2.964817
Lean Republican	3.316340	4.734177	4.898515	5.349815	4.897114	4.230769
Republican	3.686303	4.757282	4.660787	5.028649	4.743555	3.786020

```
Personal_Placement <- Personal_Placement %>%
  reshape2::melt() %>%
  mutate(
    question = case_when(
      variable == "spending" ~ "Spending & Services",
      variable == "defense" ~ "Defense Spending",
      variable == "medical" ~ "Government/Private Medical Insuranc
      variable == "job" ~ "Guaranteed job/income",
      variable == "blacks" ~ "Government Assistance to Blacks",
      variable == "business" ~ "Environment-Business Tradeoff"
    )
  )
```

PARTY7	variable	value	question
Strong Democrat	spending	5.721714	Spending & Services
Democrat	spending	5.310705	Spending & Services
Lean Democrat	spending	5.417421	Spending & Services
Independent	spending	4.675900	Spending & Services
Lean Republican	spending	3.316340	Spending & Services
Republican	spending	3.686303	Spending & Services

Labeling with Vectors

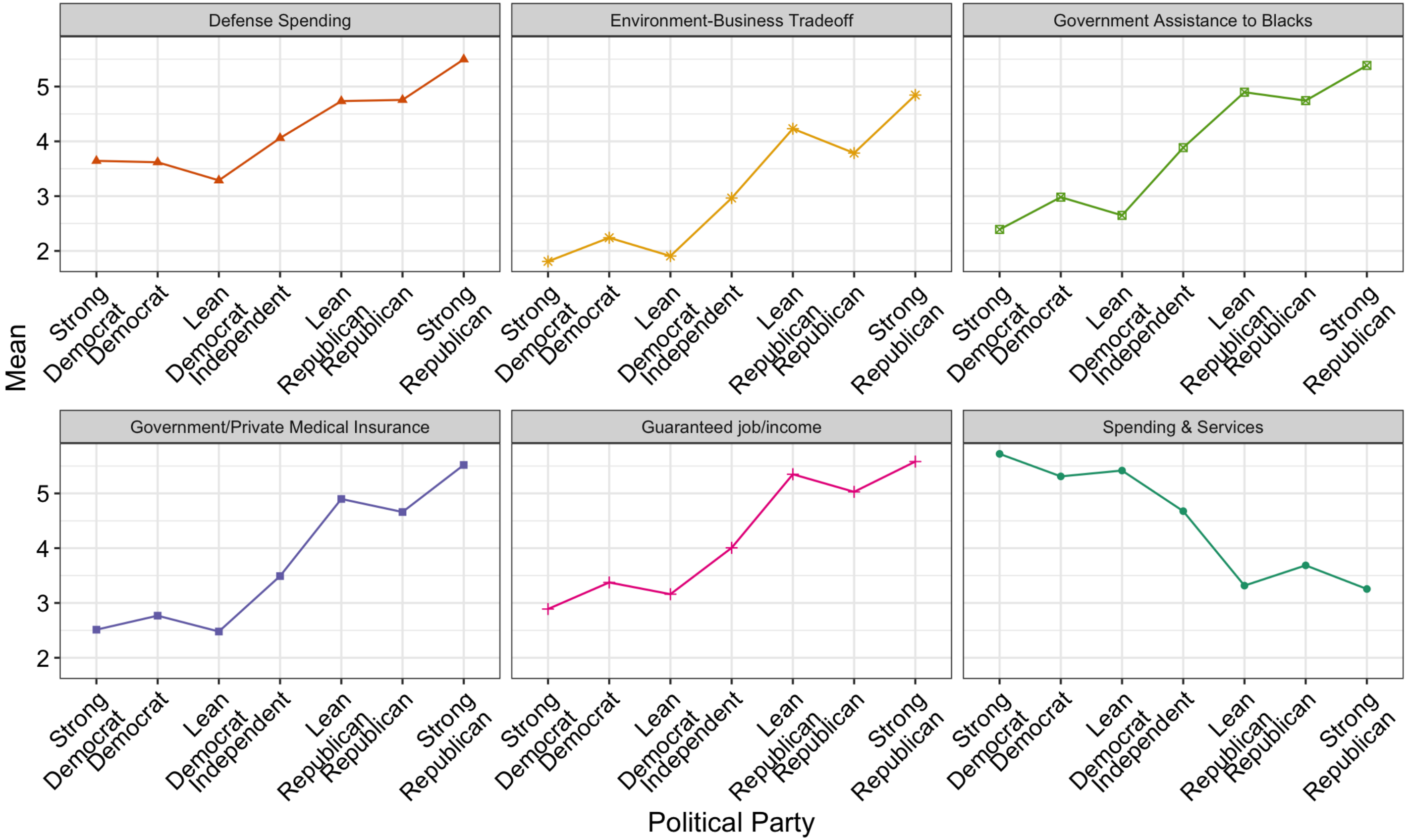
If you are making many plots on a script with similar labels, it might be easier to save the labels as a vector and reference the vector than to type out all of the labels each time.

```r
pid7_labels <- c(
  "Strong\nDemocrat", "Democrat",
  "Lean\nDemocrat", "Independent",
  "Lean\nRepublican", "Republican",
  "Strong\nRepublican")
```

```r
ggplot(Personal_Placement,
       aes(x = PARTY7, y = value,
           group = variable, color = variable, shape = variable))+
  geom_line()+
  geom_point()+
  facet_wrap(~question, scales = "free_x")+
  scale_color_brewer(palette = 'Dark2')+
  scale_x_discrete(labels = pid7_labels)+
  labs(
    title = "Distribution of Issue Attitudes",
    subtitle = "By Political Party",
    caption = "Data: ANES 2020
    Author: Jennifer Lin"
  )+
  xlab("Political Party")+
  ylab("Mean")+
  theme_rworkshop()+
  theme(
    axis.text.x     = element_text(angle = 45, hjust = 1),
    legend.position = "none"
  )
```

Distribution of Issue Attitudes

By Political Party



Data: ANES 2020
Author: Jennifer Lin

Exercise 3

From your variables in Exercise 1, generate a line graph with appropriate facets with any combination of the three variables such that it tells a story about the relationship of the variables. Include proper labels, titles, colors and themes

Your Submission

1. Export your figures from Exercise 1, 2 and 3 as landscape PDF files.
2. Upload each graph and your code to Canvas