Week 3.3: Themes, Fonts… Ultimate SPICE UP

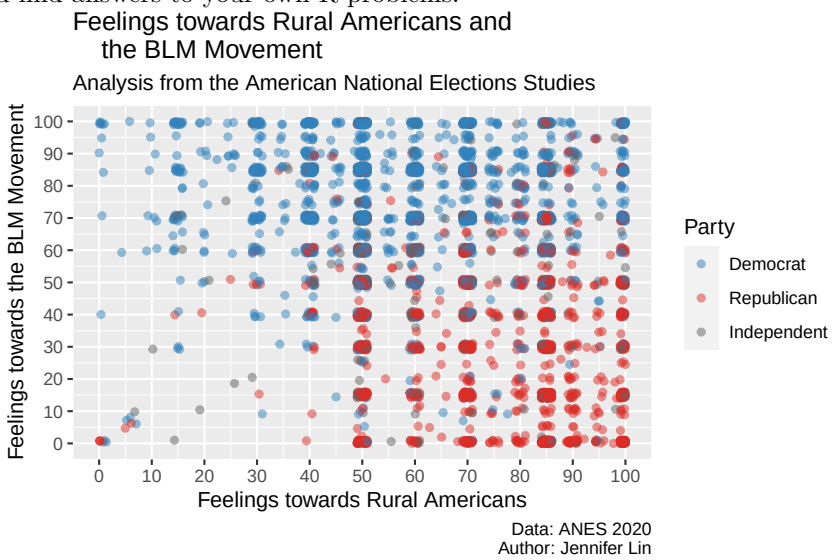
Jennifer Lin

2021-11-10

Your plot from last week now has better labels and colors compared to the week before. But, there is still more to do to make it ready. Some of these steps might include

1. Add a theme to change out of the gray background
2. Adjust the font sizes and layouts of various graph backgrounds
3. Add any additional features to spice up your graph as needed

Time permitting, I will also show you some tips on how to google and find answers to your own R problems.



The Theme Layer

The theme layer can be conceptualized in two broad categories: Global and specific options.

Starting with the global themes, there are many options to change the overall look and feel of your graph. Some of these settings are already loaded in the `ggplot2` package but there are still others that you can get using the `ggthemes` package.

The options built in `ggplot2` include some of the following:

- `theme_bw()`
- `theme_classic()`
- `theme_light()`
- `theme_linedraw()`

Options in `gthemes` expand the ones in `ggplot2` and can include things like:

- `theme_tufte()`
- `theme_gdocs()`
- `theme_calc()`

To use these, simply add them to your `ggplot` code chunks and follow this with a `+` to add other theme options.

Speaking of other theme options, we can use `theme()` to specify things like font colors, sizes, angles and orientations. Options here are based on the *specific* part of the graph that you are looking to customize. A brief synopsis of how to use `theme()` is as follows:

```
theme(
  plot.title      = element_text(hjust  = 0.5),
  plot.subtitle   = element_text(face = "bold")
  axis.title      = element_text(color ="black"),
  axis.title.y    = element_text(size   = 12),
  axis.text.x     = element_text(angle = 45),
  legend.position = 'bottom'
  legend.title    = element_text(family="serif"),
)
```

Where:

- `hjust`: Left (0), Center (0.5), Right (1) justified
- `color`: Color – can use names or HEX codes
- `size`: Font size
- `face`: Takes "plain", "italic", "bold", "bold.italic"
- `font`: Font family – assumes sans serif
- `angle`: Angle of object (0 - 360)

To determine what arguments in `theme()` you need, think about the *specific* graph section you want to change.

These follow the pattern:

`PART OF THE GRAPH + (.) + WHAT ABOUT THIS PART + (.) + ANYTHING ELSE?`

Looking at the `PART OF THE GRAPH`, this aspect generally contains three categories

- `plot.*`: Addresses the entire plot
- `axis.*`: Addresses the axis
- `legend.*`: Addresses the legends

In the `WHAT ABOUT THIS PART`, these components can include things like:

- `title`
- `subtitle`
- `text`
- `caption`
- `background`
- `position`

All that describe the feature that you are seeking to change

Usually, you can append `.x` or `.y` to the argument to specify axis changes. This is usually the only component in the `ANYTHING ELSE?` component.

Putting the theme layer together, it can look something like this

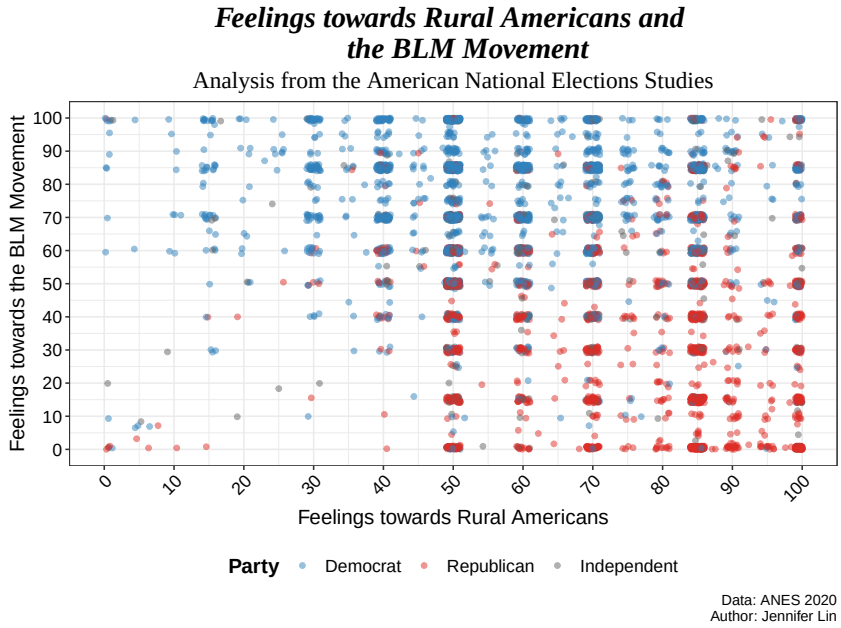
```
theme_bw()+
theme(
  plot.title         = element_text(
    hjust = 0.5, size = 20, colour="black",
    face = "bold.italic", family="serif"),
  plot.subtitle      = element_text(
    hjust = 0.5, size = 16,
    colour="black", family="serif"),
  legend.title       = element_text(
    hjust = 0.5, size = 14,
    colour="black", face = "bold"),
  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",
    angle = 45, hjust = 1),
  axis.text.y        = element_text(size = 12, colour="black"),
  legend.position    = 'bottom',
  legend.direction   = "horizontal",
  legend.text        = element_text(size = 12, colour="black")
)
```

Now, our full graph looks something like so[1]:

[1] This is *not* a publication quality graph as fonts should be in the same font family if submitting to a journal

```
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
```

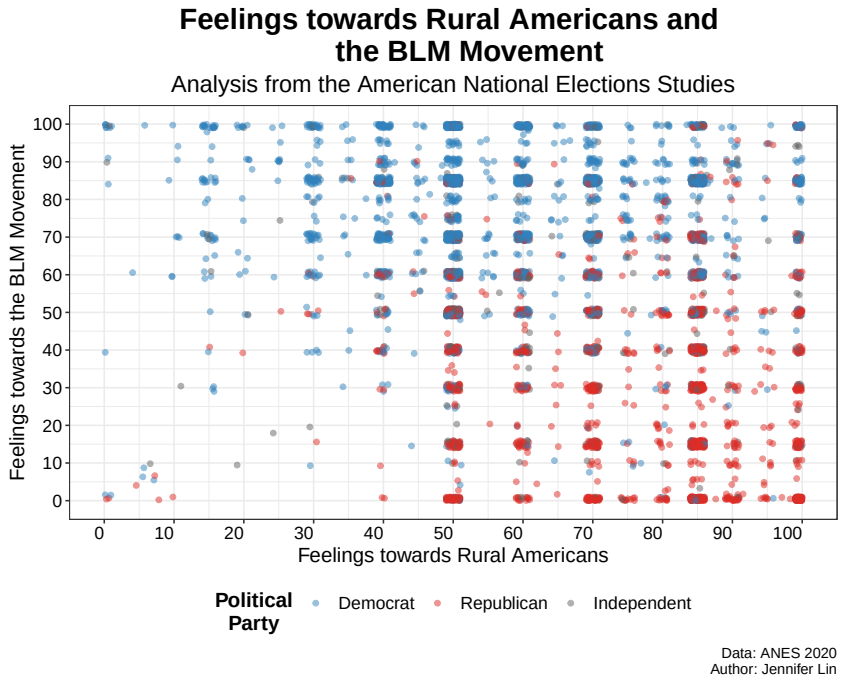
```
  the BLM Movement",
  subtitle = "Analysis from the American National Elections Studies",
  caption = "Data: ANES 2020
  Author: Jennifer Lin"
)+
scale_color_manual(
  name = "Party",
  breaks = c("Democrat",  "Republican", "Independent"),
  values = c("Democrat" = "#3182bd", "Republican" = "#de2d26", "Independent" = "#636363")
)+
scale_x_continuous(
  breaks = seq(0, 100, 10),
  limits = c(0, 100)
)+
scale_y_continuous(
  breaks = seq(0, 100, 10),
  limits = c(0, 100)
)+
theme_bw()+
  theme(
  plot.title         = element_text(hjust = 0.5, size = 20, colour="black", face = "bold.italic", fami
  plot.subtitle      = element_text(hjust = 0.5, size = 16, colour="black", family="serif"),
  legend.title       = element_text(hjust = 0.5, size = 14, colour="black", face = "bold"),
  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", angle = 45, hjust = 1),
  axis.text.y        = element_text(size = 12, colour="black"),
  legend.position    = 'bottom',
  legend.direction   = "horizontal",
  legend.text        = element_text(size = 12, colour="black")
)
```



This is not a publication ready graph mainly because of the font differences between the title and the rest of the graph. For teaching purposes, I used different sizes and styles to show you what happens. However, if I were submitting it as part of a paper, here is what it would look like:

```
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 Studies",
    caption = "Data: ANES 2020
    Author: Jennifer Lin"
  )+
  scale_color_manual(
    name = "Political\nParty",
    breaks = c("Democrat",  "Republican", "Independent"),
    values = c("Democrat" = "#3182bd", "Republican" = "#de2d26", "Independent" = "#636363")
  )+
  scale_x_continuous(
    breaks = seq(0, 100, 10),
```

```
  limits = c(0, 100)
)+
scale_y_continuous(
  breaks = seq(0, 100, 10),
  limits = c(0, 100)
)+
theme_bw()+
  theme(
  plot.title          = element_text(hjust = 0.5, size = 20, colour="black", face = "bold"),
  plot.subtitle       = element_text(hjust = 0.5, size = 16, colour="black"),
  legend.title        = element_text(hjust = 0.5, size = 14, colour="black", face = "bold"),
  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", hjust = 1),
  axis.text.y         = element_text(size = 12, colour="black"),
  legend.position     = 'bottom',
  legend.direction    = "horizontal",
  legend.text         = element_text(size = 12, colour="black")
)
```



Exercise: Adding Themes to your Plots

1. Take your code from last week and add it into this week's script
2. Add a global theme
3. Add specific theme options to adjust your titles, axis labels and

legend

Other Components

There are some other components to `ggplot2` and these include:

- `facet_grid()` and `facet_wrap()` can group your data by a designated grouping variable
- `coord_flip()` changes what is on your x-axis to y-axis and vice versa

But the possibilities of other components in `ggplot2` are not limited to these components, or the things you learned in the past 3 weeks. There is a lot that a module, or a course even, cannot cover for each of your specific research project's graphics needs.

Addressing your own R Problems

Thinking back to Week 1, in the first exercise, when you were deciding what graph to make, I asked you to first think about it *without code*. This was deliberate because often we want to do things in R but do not have all the skills for it or do not know what code to even use. But not having the skills should not stop you from wanting to make something in R.

In this course, I did not cover everything in `ggplot2`. There is just so much to do with any of the tidyverse packages that a short course does not cover all that you will need.

We can use Google to help fill the gaps! Google is often your best friend when it comes to solving R problems, and especially Stack Overflow. However, the challenge is knowing what to google. (Finding words can be hard.)

Here are some tips for how to google your R problems and getting the most success[2]:

1. Draw (on paper) your desired end result and find words around that
2. Take advantage of related searches
3. Use the "Ask Question" feature and have Stack Overflow AI help you
4. Post your own Question on Stack Overflow.

[2] This will often take several tries and requires patience!

Exercise: Stack Overflow

Here are some common R challenges that I have used stack overflow for. Pick one, do a stack overflow search and apply it to your graph

1. Make the legend 2 column
2. Make the legend run vertical/horizontal
3. Add space between the axis text and the plot
4. Or ANYTHING that you think can help make your graph look nicer

Your Submission to the Lab Assignment for this week

1. A finalized, publication ready PDF version of your graph and the corresponding script file
2. Your stack overflow search and the answer that you used – you can write this as part of your script file in the form of a comment. State your question and insert the URL for the thread.
3. In a few words, discuss how you added your stack overflow findings to your plot.