

# AI Appendix

## POL837 Issues in Comparative Politics - Research Paper

Daniel Sánchez

GitHub Copilot Welcome, @dsanchezp18, I'm your Copilot and I'm here to help you get things done faster.

I'm powered by AI, so surprises and mistakes are possible. Make sure to verify any generated code or suggestions, and share feedback so that we can learn and improve. Check out the Copilot documentation to learn more.

dsanchezp18: apply the labels to a column which was read from a sav file with haven in R

GitHub Copilot: To apply labels to a column that was read from a .sav file with the **haven** package in R, you can use the **labelled** function from the **haven** package. Here's an example:

```
# Load the haven package
library(haven)

# Read the .sav file
data <- read_sav("your_file.sav")

# Apply labels to a column
data$your_column <- labelled(data$your_column, labels = c("Label1" = 1, "Label2" = 2))

# Print the data
print(data)
```

In this code, replace "your\_file.sav" with the path to your .sav file, and replace **your\_column** with the name of the column you want to label. The **labels = c("Label1" = 1, "Label2" = 2)** part is where you specify the labels you want to apply: "Label1" will be applied to all 1s in the column, and "Label2" will be applied to all 2s. Adjust this to match your specific labels and values.

dsanchezp18: is there an automatic way to do that

GitHub Copilot: Yes, there is an automatic way to apply labels to a column read from a .sav file using the **haven** package in R. When you read a .sav file using **read\_sav**, it automatically converts labelled variables to a special class called **haven\_labelled**. You can convert these to factors using the **haven::as\_factor** function, which will use the labels from the .sav file.

Here's an example:

```
# Load the haven package
library(haven)

# Read the .sav file
data <- read_sav("your_file.sav")

# Convert labelled variables to factors
data <- purrr::map_df(data, haven::as_factor)

# Print the data
print(data)
```

In this code, replace "your\_file.sav" with the path to your .sav file. The **purrr::map\_df(data, haven::as\_factor)** line converts all the labelled variables in the data frame to factors using their labels.

dsanchezp18: without factors

GitHub Copilot: If you want to keep the data as numeric but still have the labels attached for reference, you can use the **labelled** class from the **haven** package. When you read a .sav file with **read\_sav**, it automatically preserves the value labels from SPSS. You can access these labels using the **labels** function from **haven**.

Here's an example:

```
# Load the haven package
library(haven)

# Read the .sav file
data <- read_sav("your_file.sav")

# Access labels of a column
column_labels <- labels(data$your_column)

# Print the labels
print(column_labels)
```

In this code, replace "your\_file.sav" with the path to your .sav file, and replace your\_column with the name of the column whose labels you want to access. The `labels(data$your_column)` line gets the labels of the specified column.

Please note that this will not “apply” the labels in the sense of replacing the numeric values with their corresponding labels. It will simply allow you to access the labels for reference while keeping the data numeric.

dsanchezp18: filter for only missing values in R

GitHub Copilot: To filter for only missing values in a specific column in R, you can use the `is.na()` function inside a `filter()` function from the `dplyr` package. Here’s how you can modify your existing code to filter for only missing values in the `paroq` column:

```
library(dplyr)

ecu_ab_raw %>%
  mutate(parish = zap_missing(paroq)) %>%
  filter(is.na(paroq)) %>%
  glimpse()
```

In this code, `filter(is.na(paroq))` filters the data frame to include only rows where `paroq` is missing.

dsanchezp18: another way with dplyr

GitHub Copilot: If you want to filter for rows where the `paroq` column has missing values using `dplyr`, you can use the `filter()` function with `is.na()`. Here’s how you can modify your existing code:

```
library(dplyr)

ecu_ab_raw %>%
  mutate(parish = zap_missing(paroq)) %>%
  select(paroq) %>%
  filter(is.na(paroq)) %>%
  glimpse()
```

In this code, `filter(is.na(paroq))` filters the data frame to include only rows where `paroq` is missing.

dsanchezp18: download a list of files

GitHub Copilot: To download a list of files in R, you can use the `download.file()` function inside a `lapply()` function. Here’s how you can modify your existing code:

```
# List of URLs
urls <- c("url1", "url2", "url3")

# Corresponding list of paths where the files should be saved
paths <- c("path1", "path2", "path3")

# Download the files
lapply(seq_along(urls), function(i) {
  download.file(urls[i], paths[i], mode = "wb")
})
```

In this code, replace "url1", "url2", and "url3" with your actual URLs, and replace "path1", "path2", and "path3" with the paths where you want to save the files. The `lapply(seq_along(urls), function(i) {...})` part applies the `download.file()` function to each pair of URL and path.

dsanchezp18: download.file along a vector of files

GitHub Copilot: If you have vectors of URLs and corresponding file paths, you can use the `mapply()` function to download each file. Here's how you can modify your existing code:

```
# Vectors of URLs and paths
max_temperature_urls <- c("url1", "url2", "url3")
max_temp_paths <- c("path1", "path2", "path3")

# Download the files
mapply(download.file, max_temperature_urls, max_temp_paths, MoreArgs = list(mode = "wb"))
```

In this code, replace "url1", "url2", and "url3" with your actual URLs, and replace "path1", "path2", and "path3" with the paths where you want to save the files. The `mapply(download.file, max_temperature_urls, max_temp_paths, MoreArgs = list(mode = "wb"))` part applies the `download.file()` function to each pair of URL and path.

dsanchezp18: how do I execute r files in bashj

GitHub Copilot: To execute R scripts from a bash shell, you can use the `Rscript` command followed by the path to your R script. Here's an example:

```
Rscript /path/to/your/script.R
```

In your case, if you want to execute the `canton_weather_data_prepare.R` script, you would use:

```
Rscript canton_weather_data_prepare.R
```

This assumes that the script is in your current working directory. If it's in a different directory, you would need to specify the full path to the script.

dsanchezp18: compute the average between two columns in R

GitHub Copilot: To compute the average between two columns in R, you can use the `rowMeans()` function. Here's an example:

```
# Assuming df is your data frame and col1 and col2 are the columns
df$average <- rowMeans(df[,c('col1', 'col2')], na.rm = TRUE)
```

In this code, replace `df` with your actual data frame, and replace `'col1'` and `'col2'` with the names of the columns you want to average. The `rowMeans(df[,c('col1', 'col2')], na.rm = TRUE)` part computes the average of `col1` and `col2` for each row, ignoring NA values. The result is stored in a new column called `average`.

dsanchezp18: using mutate

GitHub Copilot: To compute the average between two columns in R using `mutate()` from the `dplyr` package, you can do the following:

```
library(dplyr)

df <- df %>%
  mutate(average = (col1 + col2) / 2)
```

In this code, replace `df` with your actual data frame, and replace `col1` and `col2` with the names of the columns you want to average. The `(col1 + col2) / 2` part computes the average of `col1` and `col2` for each row. The result is stored in a new column called `average`.

dsanchezp18: use rowmeans too

GitHub Copilot: To compute the average between two columns in R using `mutate()` and `rowMeans()` from the `dplyr` package, you can do the following:

```
library(dplyr)

df <- df %>%
  mutate(average = rowMeans(select(., col1, col2), na.rm = TRUE))
```

In this code, replace `df` with your actual data frame, and replace `col1` and `col2` with the names of the columns you want to average. The `rowMeans(select(., col1, col2), na.rm = TRUE)` part computes the average of `col1` and `col2` for each row, ignoring NA values. The result is stored in a new column called `average`.

dsanchezp18: calculate the number of day in the year with lubridate

GitHub Copilot: To calculate the day of the year in R, you can use the `yday()` function from the `lubridate` package. Here's how you can modify your existing code:

```
library(lubridate)

ecuador_avg_temp_daily <-
  temperature_df %>%
  group_by(day_of_year = yday(date)) %>%
  summarise(avg_temp = mean(avg_temp, na.rm = T))
```

In this code, `yday(date)` calculates the day of the year for each date. The `group_by(day_of_year = yday(date))` part groups the data by day of the year. The result is a data frame where each row corresponds to a day of the year, and `avg_temp` is the average temperature for that day.

dsanchezp18: show only the top portion of the bar in ggplot

GitHub Copilot: To show only the top portion of the bar in a bar plot using `ggplot2`, you can adjust the y-axis limits using the `coord_cartesian()` function. Here's an example:

```
library(ggplot2)

# Assuming df is your data frame, x_var is your x variable, and y_var is your y variable
ggplot(df, aes(x = x_var, y = y_var)) +
  geom_bar(stat = "identity") +
  coord_cartesian(ylim = c(50, 100)) # Adjust these values to show the desired portion of the bar
```

In this code, replace `df` with your actual data frame, and replace `x_var` and `y_var` with your actual variables. The `coord_cartesian(ylim = c(50, 100))` part adjusts the y-axis limits to show only the portion of the bar from 50 to 100. Adjust these values to show the desired portion of the bar.

dsanchezp18: how to show the name of a month from a date

GitHub Copilot: To extract the name of the month from a date in R, you can use the `month()` function from the `lubridate` package with the `label` argument set to `TRUE`. Here's an example:

```
library(lubridate)

# Assuming df is your data frame and date_var is your date variable
df$month_name <- month(df$date_var, label = TRUE)
```

In this code, replace `df` with your actual data frame, and replace `date_var` with your actual date variable. The `month(df$date_var, label = TRUE)` part extracts the name of the month from `date_var`. The result is stored in a new column called `month_name`.

dsanchezp18: count missing values in R

GitHub Copilot: To count the number of missing values in a data frame in R, you can use the `is.na()` function along with the `sum()` function. Here's an example:

```
# Assuming df is your data frame
missing_values <- sum(is.na(df))

# Print the number of missing values
print(missing_values)
```

In this code, replace `df` with your actual data frame. The `sum(is.na(df))` part counts the number of missing values in the data frame. The result is stored in the `missing_values` variable.

dsanchezp18: use a dplyr chain

GitHub Copilot: To count the number of missing values in a data frame in R using a `dplyr` chain, you can use the `summarise_all()` function along with the `is.na()` and `sum()` functions. Here's an example:

```
library(dplyr)

# Assuming df is your data frame
df %>%
  summarise_all(function(x) sum(is.na(x))) %>%
  sum()
```

In this code, replace `df` with your actual data frame. The `summarise_all(function(x) sum(is.na(x)))` part counts the number of missing values in each column of the data frame. The `sum()` function then adds up these counts to get the total number of missing values in the data frame.

dsanchezp18: how to apply value labels in stata

GitHub Copilot: To apply value labels in Stata, you can use the `label define` and `label values` commands. Here's an example:

```
* Define the labels
label define mylabel 1 "Label 1" 2 "Label 2" 3 "Label 3"

* Apply the labels to a variable
label values myvar mylabel
```

In this code, replace `mylabel` with the name you want to give to your set of labels, replace `1 "Label 1" 2 "Label 2" 3 "Label 3"` with your actual values and labels, and replace `myvar` with the name of the variable you want to apply the labels to. The `label define mylabel 1 "Label 1" 2 "Label 2" 3 "Label 3"` part defines a set of labels, and the `label values myvar mylabel` part applies these labels to `myvar`.

dsanchezp18: how to use `scale_color_DISCRETE`

GitHub Copilot: The `scale_color_discrete()` function in `ggplot2` is used to change the default discrete color scale. It can be used to change the color palette, the labels, and the legend name.

In your code, `scale_color_discrete(name = "Wave")` is used to change the legend title to “Wave”.

If you want to change the color palette, you can use the `values` argument. For example, if you want to use the colors red and blue, you can do:

```
scale_color_discrete(name = "Wave", values = c("red", "blue"))
```

If you want to change the labels, you can use the `labels` argument. For example, if you want to change the labels to “Wave 1” and “Wave 2”, you can do:

```
scale_color_discrete(name = "Wave", labels = c("Wave 1", "Wave 2"))
```

Note that the order of the colors and labels should match the order of the levels of the factor variable.

dsanchezp18: apply another kind of palette to the data

GitHub Copilot: To apply a different color palette to your data, you can use the `scale_color_brewer()` function from the `ggplot2` package. This function provides color scales from ColorBrewer. Here's how you can modify your existing code:



```
# Plotting survey responses through time
df %>%
  group_by(interview_date, wave) %>%
  summarise(count_of_responses = n()) %>%
  ggplot(aes(x = interview_date, y = count_of_responses, color = wave)) +
  geom_col() +
  scale_x_date(date_labels = "%b %Y", date_breaks = "1 year") +
  scale_color_brewer(name = "Wave", palette = "Set1")
```

In this code, `scale_color_brewer(name = "Wave", palette = "Set1")` applies the “Set1” palette from ColorBrewer to your data. You can replace “Set1” with the name of any other ColorBrewer palette.

dsanchezp18: facetting in ggplot2

GitHub Copilot: Facetting in ggplot2 is a way to create multiple small plots that share the same x and y axes. It's a powerful tool for creating complex graphics, and it's especially useful for comparing different subsets of your data.

In your code, `facet_wrap(~wave)` is used to create a separate plot for each level of the `wave` variable. Each plot shares the same x and y axes, which makes it easy to compare the plots.

Here's a brief explanation of how `facet_wrap()` works:

- `facet_wrap(~wave)`: This creates a separate plot for each level of the `wave` variable. The plots are arranged in a grid that fills up row-wise.

You can also control the layout of the grid by specifying the `nrow` or `ncol` argument. For example, `facet_wrap(~wave, nrow = 2)` would arrange the plots in a grid with 2 rows.

If you want to facet by two variables, you can use `facet_grid()`. For example, `facet_grid(rows = vars(wave), cols = vars(another_variable))` would create a grid of plots with different levels of `wave` along the rows and different levels of `another_variable` along the columns.

dsanchezp18: currently my graph does not show the values of the variable in x, how to show.

GitHub Copilot: To show the values of the variable on the x-axis, you can use the `scale_x_date()` function from the `ggplot2` package. This function provides a variety of options for formatting the date labels. Here's how you can modify your existing code:

```
df %>%
  group_by(interview_date, wave) %>%
  summarise(count_of_responses = n()) %>%
  ggplot(aes(x = interview_date, y = count_of_responses)) +
  geom_col() +
```

```
facet_wrap(~wave, nrow = 2) +
scale_color_discrete(palette = "Set4") +
scale_x_date(date_labels = "%b %Y", date_breaks = "1 month") # Add this line
```

In this code, `scale_x_date(date_labels = "%b %Y", date_breaks = "1 month")` formats the date labels on the x-axis. The `date_labels = "%b %Y"` part specifies that the date labels should be displayed as the abbreviated month name followed by the four-digit year (e.g., “Jan 2020”), and the `date_breaks = "1 month"` part specifies that there should be a tick mark on the x-axis for every month. You can adjust these options to suit your needs.

dsanchezp18: free y

GitHub Copilot: To allow each facet to have its own y-axis scale in ggplot2, you can use the `scales` argument in the `facet_wrap()` function and set it to “free\_y”. Here’s how you can modify your existing code:

```
df %>%
  group_by(interview_date, wave) %>%
  summarise(count_of_responses = n()) %>%
  ggplot(aes(x = interview_date, y = count_of_responses)) +
  geom_col() +
  facet_wrap(~wave, nrow = 2, scales = "free_y") + # Add scales = "free_y"
  scale_x_date(date_labels = "%b %Y", date_breaks = "6 month") +
  scale_color_discrete(palette = "Set4")
```

In this code, `facet_wrap(~wave, nrow = 2, scales = "free_y")` creates a separate plot for each level of the `wave` variable, with each plot having its own y-axis scale.

dsanchezp18: show the scale of dates on both the first and second row of the facets

GitHub Copilot: To show the x-axis on every facet, you can use the `theme()` function with the `strip.placement` argument set to “outside”. Here’s how you can modify your existing code:

```
df %>%
  group_by(interview_date, wave) %>%
  summarise(count_of_responses = n()) %>%
  ggplot(aes(x = interview_date, y = count_of_responses)) +
  geom_col() +
  facet_wrap(~wave, nrow = 2, scales = "free_y") +
  scale_x_date(date_labels = "%b%Y", date_breaks = "6 month") +
  scale_color_discrete(palette = "Set4") +
  theme(strip.placement = "outside") # Add this line
```

In this code, `theme(strip.placement = "outside")` moves the facet labels (i.e., the `wave` values) outside of the plot area, which makes the x-axis appear on every facet. Note that this will only work if you have the latest version of `ggplot2` installed. If you have an older version, you may need to update it.

dsanchezp18: customize the rectangle of the factor title

GitHub Copilot: To customize the rectangle of the facet title in `ggplot2`, you can use the `strip.background` argument in the `theme()` function. Here's how you can modify your existing code:

```
interview_dates_barchart <- df %>%
  filter(!is.na(interview_date), !(interview_date %>% between(as.Date("2008-07-10"), as.Date("2017-01-02"))))
  group_by(interview_date, wave) %>%
  summarise(count_of_responses = n()) %>%
  ggplot(aes(x = interview_date, y = count_of_responses)) +
  geom_col() +
  geom_vline(xintercept = as.Date("2008-01-01"), linetype = "dashed", color = "blue") +
  geom_vline(xintercept = as.Date("2009-01-01"), linetype = "dashed", color = "blue") +
  geom_vline(xintercept = as.Date("2012-01-01"), linetype = "dashed", color = "blue") +
  geom_vline(xintercept = as.Date("2017-01-02"), linetype = "dashed", color = "blue", linewidth = 1) +
  facet_wrap(~wave, nrow = 2, scales = "free") +
  scale_x_date(date_labels = "%Y-%m-%d", date_breaks = "1 week") +
  scale_color_discrete(palette = "Set4") +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1, size = 6),
    text = element_text(size = 10, family = 'serif'),
    plot.background = element_rect(fill = "white"),
    panel.border = element_rect(colour = "black", fill = NA, linewidth = 1, linetype = "solid"),
    plot.caption = element_text(hjust = 0),
    panel.grid.major = element_line(linetype = "dashed"),
    panel.grid.minor = element_line(linetype = "dashed"),
    strip.background = element_rect(fill = "lightblue", colour = "black", size = 1) # Add the background color of the facet title rectangle
  )

interview_dates_barchart
```

In this code, `strip.background = element_rect(fill = "lightblue", colour = "black", size = 1)` changes the background color of the facet title rectangle to light blue, the border color to black, and the border size to 1. You can adjust these options to suit your needs.

dsanchezp18: start my date axis in january

GitHub Copilot: To start your date axis in January, you can use the `expand` argument in the `scale_x_date()` function. This argument allows you to specify the limits of the x-axis. Here's how you can modify your existing code:

```
temperature_df %>%
  mutate(month_year = floor_date(date, "month")) %>%
  group_by(month_year) %>%
  summarise(min_temp = mean(min_temperature, na.rm = T),
            max_temp = mean(max_temperature, na.rm = T)) %>%
  pivot_longer(cols = c(min_temp, max_temp),
               names_to = "temperature_type",
               values_to = "value") %>%
  ggplot(aes(x = month_year, y = value, color = temperature_type)) +
  geom_line() +
  scale_x_date(date_labels = "%b %Y", date_breaks = "3 months",
               expand = c(0, 0), limits = as.Date(c("first January of your data", "last date
  labs(x = "Date",
        y = "Temperature (C)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

In this code, `expand = c(0, 0)`, `limits = as.Date(c("first January of your data", "last date of your data"))` sets the limits of the x-axis to start from the first January of your data and end at the last date of your data. Replace "first January of your data" and "last date of your data" with the actual dates you want to use.

dsanchezpl8: does not work

GitHub Copilot: The issue seems to be with the `limits` argument in the `scale_x_date()` function. The `limits` argument expects a vector of two dates, but in your code, it's receiving only one date because the `c()` function is not used correctly.

Here's how you can fix it:

```
temperature_df %>%
  mutate(month_year = floor_date(date, "month")) %>%
  group_by(month_year) %>%
  summarise(min_temp = mean(min_temperature, na.rm = T),
            max_temp = mean(max_temperature, na.rm = T)) %>%
  pivot_longer(cols = c(min_temp, max_temp),
               names_to = "temperature_type",
               values_to = "value") %>%
  ggplot(aes(x = month_year, y = value, color = temperature_type)) +
```

```

geom_line() +
labs(x = "Date",
     y = "Temperature (C)") +
scale_x_date(date_labels = "%b %Y",
             date_breaks = "6 months",
             expand = c(0,0),
             limits = c(as.Date("2001-01-01"), as.Date("2024-07-01"))) + # Modify this line
theme_minimal() +
theme(axis.text.x = element_text(angle = 45, hjust = 1))

```

In this code, `limits = c(as.Date("2001-01-01"), as.Date("2024-07-01"))` sets the limits of the x-axis to start from January 2001 and end at July 2024. The `as.Date()` function is used to convert the date strings to Date objects.

dsanchezp18: make axis ticks every 2 units in the y axis

GitHub Copilot: To make axis ticks every 2 units on the y-axis, you can modify the `breaks` argument in the `scale_y_continuous()` function. Here's how you can modify your existing code:

```

ggplot(aes(x = month_year, y = value, color = temperature_type)) +
  geom_line() +
  labs(x = "",
       y = "Temperature (\u00B0C)",
       color = "Temperature") +
  scale_x_date(date_labels = "%b %Y",
              date_breaks = "6 months",
              expand = c(0,0),
              limits = c(as.Date("2008-01-01"), as.Date("2024-07-01"))) +
  scale_y_continuous(labels = number_format(accuracy = 2),
                    breaks = seq(min(value), max(value), by = 2)) + # Modify this line
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, color = "black"),
        text = element_text(family = 'serif', color = "black"),
        plot.background = element_rect(fill = "white"),
        panel.border = element_rect(colour = "black", fill = NA, linewidth = 1, linetype = "dashed"),
        plot.caption = element_text(hjust = 0),
        panel.grid.major = element_line(linetype = "dashed", linewidth = 0.3),
        panel.grid.minor = element_line(linetype = "dashed", linewidth = 0.3),
        strip.background = element_rect(fill = "grey80", colour = "black", linewidth = 1),
        legend.position = c(0.9, 0.05)
)

```

In this code, `breaks = seq(min(value), max(value), by = 2)` sets the y-axis to have a tick mark every 2 units. The `min(value)` and `max(value)` parts ensure that the y-axis covers the entire range of the `value` variable. Replace `value` with the actual variable you're using for the y-axis if it's different.

dsanchezp18: round to one decimal

GitHub Copilot: To round the y-axis labels to one decimal place, you can modify the `labels` argument in the `scale_y_continuous()` function. Here's how you can modify your existing code:

```
ecuador_monthly_mean_temps_fig <-  
  ecuador_monthly_mean_temps %>%  
  ggplot(aes(x = month_year, y = value, color = temperature_type)) +  
  geom_line() +  
  labs(x = "",  
       y = "Temperature (\u00B0C)",  
       color = "Temperature") +  
  scale_x_date(date_labels = "%b %Y",  
              date_breaks = "6 months",  
              expand = c(0,0),  
              limits = c(as.Date("2008-01-01"), as.Date("2024-07-01"))) +  
  scale_y_continuous(breaks = seq(min(ecuador_monthly_mean_temps$value), max(ecuador_monthly.  
                                labels = function(x) format(x, nsmall = 1)) + # Modify this line  
  theme_minimal() +  
  theme(axis.text.x = element_text(angle = 45, hjust = 1, color = "black"),  
        text = element_text(family = 'serif', color = "black"),  
        plot.background = element_rect(fill = "white"),  
        panel.border = element_rect(colour = "black", fill = NA, linewidth = 1, linetype = "dashed"),  
        plot.caption = element_text(hjust = 0),  
        panel.grid.major = element_line(linetype = "dashed", linewidth = 0.3),  
        panel.grid.minor = element_line(linetype = "dashed", linewidth = 0.3),  
        strip.background = element_rect(fill = "grey80", colour = "black", linewidth = 1),  
        legend.position = c(0.9, 0.05)  
  )
```

In this code, `'labels = function(x) format`

dsanchezp18: use scales::

GitHub Copilot: To round the y-axis labels to one decimal place using the `scales` package, you can use the `scales::label_number()` function in the `labels` argument of the `scale_y_continuous()` function. Here's how you can modify your existing code:

```

ecuador_monthly_mean_temps_fig <-
  ecuador_monthly_mean_temps %>%
  ggplot(aes(x = month_year, y = value, color = temperature_type)) +
  geom_line() +
  labs(x = "",
       y = "Temperature (\u00B0C)",
       color = "Temperature") +
  scale_x_date(date_labels = "%b %Y",
               date_breaks = "6 months",
               expand = c(0,0),
               limits = c(as.Date("2008-01-01"), as.Date("2024-07-01"))) +
  scale_y_continuous(breaks = seq(min(ecuador_monthly_mean_temps$value), max(ecuador_monthly_mean_temps$value), by = 1),
                     labels = scales::label_number(accuracy = 1)) + # Modify this line
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, color = "black"),
        text = element_text(family = 'serif', color = "black"),
        plot.background = element_rect(fill = "white"),
        panel.border = element_rect(colour = "black", fill = NA, linewidth = 1, linetype = "solid"),
        plot.caption = element_text(hjust = 0),
        panel.grid.major = element_line(linetype = "dashed", linewidth = 0.3),
        panel.grid.minor = element_line(linetype = "dashed", linewidth = 0.3),
        strip.background = element_rect(fill = "grey80", colour = "black", linewidth = 1),
        legend.position = c(0.9, 0.05))
)

```

In this code, `labels = scales::label_number(accuracy = 1)` uses the `scales::label_number()` function to format the y-axis labels with one decimal place.

dsanchezp18: doesnt work

GitHub Copilot: I apologize for the confusion. The `scales::label_number()` function should be used with the `scales` package explicitly called. Here's the corrected code:

```

ecuador_monthly_mean_temps_fig <-
  ecuador_monthly_mean_temps %>%
  ggplot(aes(x = month_year, y = value, color = temperature_type)) +
  geom_line() +
  labs(x = "",
       y = "Temperature (\u00B0C)",
       color = "Temperature") +
  scale_x_date(date_labels = "%b %Y",
               date_breaks = "6 months",
               expand = c(0,0),

```

```

        limits = c(as.Date("2008-01-01"), as.Date("2024-07-01"))) +
scale_y_continuous(breaks = seq(min(ecuador_monthly_mean_temps$value), max(ecuador_monthly_mean_temps$value),
                             labels = scales::label_number(accuracy = 1)) + # Modify this line
theme_minimal() +
theme(axis.text.x = element_text(angle = 45, hjust = 1, color = "black"),
      text = element_text(family = 'serif', color = "black"),
      plot.background = element_rect(fill = "white"),
      panel.border = element_rect(colour = "black", fill = NA, linewidth = 1, linetype = "solid"),
      plot.caption = element_text(hjust = 0),
      panel.grid.major = element_line(linetype = "dashed", linewidth = 0.3),
      panel.grid.minor = element_line(linetype = "dashed", linewidth = 0.3),
      strip.background = element_rect(fill = "grey80", colour = "black", linewidth = 1),
      legend.position = c(0.9, 0.05)
)

ecuador_monthly_mean_temps_fig

```

In this code, `labels = scales::label_number(accuracy = 1)` uses the `scales::label_number()` function to format the y-axis labels with one decimal place.

dsanchezp18: save the current dataset in memory in stata

GitHub Copilot: To save the current dataset in memory in Stata, you can use the **save** command followed by the name of the file you want to save it as. Here's how you can do it:

```
save mydata.dta, replace
```

In this code, `mydata.dta` is the name of the file you want to save the dataset as. The **replace** option is used to overwrite the existing file if it already exists. If you don't want to overwrite the existing file, you can omit the **replace** option.

dsanchezp18: no, save it in memory to load another variable and reload again

GitHub Copilot: In Stata, you can use the **preserve** command to save the current dataset in memory. Then, you can load or modify other datasets. When you're done, you can use the **restore** command to reload the preserved dataset. Here's how you can do it:

```

preserve

* Load or modify other datasets here

restore

```



In this code, **preserve** saves the current dataset in memory, and **restore** reloads the preserved dataset. You can put any commands to load or modify other datasets between these two commands.

dsanchezp18: translate judicial

GitHub Copilot: Sorry, but I can only assist with programming related questions.

dsanchezp18: do an feglm with logistic

GitHub Copilot: The **feglm** function from the **fixest** package in R is used to estimate generalized linear models with multiple fixed-effects. The **family** argument specifies the error distribution to use. For a logistic regression, you can use **family = binomial(link = "logit")**.

Here's how you can modify your existing code:

```
simple_model1 <-  
  feglm(approves_president ~ min_temperature,  
        data = full_df,  
        family = binomial(link = "logit")) # Use logistic regression
```

In this code, **approves\_president ~ min\_temperature** is the formula for the model, **full\_df** is the data frame containing the data, and **family = binomial(link = "logit")** specifies that a logistic regression should be used.

dsanchezp18: change the name of an object in R

GitHub Copilot: To change the name of an object in R, you can simply assign the object to a new name and then remove the old object. Here's how you can do it:

```
new_name <- old_name  
rm(old_name)
```

In this code, **new\_name** is the new name you want to give to the object, and **old\_name** is the current name of the object. The **rm()** function is used to remove the old object.

For example, if you want to change the name of **simple\_model1** to **baseline\_model**, you can do:

```
baseline_model <- simple_model1  
rm(simple_model1)
```

dsanchezp18: can feglm work with a factor dependent variable

GitHub Copilot: Yes, the **feglm** function from the **fixest** package in R can work with a factor dependent variable. When the dependent variable is a factor, it is typically used with

`family = binomial(link = "logit")` to perform logistic regression, which is suitable for binary outcomes.

In your code, `approves_president` is presumably a binary factor variable, and you're correctly specifying `family = binomial(link = "logit")` to perform logistic regression.

```
simple_model1 <-  
  feglm(approves_president ~ min_temperature | canton + as_factor(interview_date),  
        data = full_df,  
        family = binomial(link = "logit"))
```

This code models the log odds of `approves_president` as a linear function of `min_temperature`, with fixed effects for `canton` and `interview_date`.

dsanchezp18: how to get the fixed effects from my reg

GitHub Copilot: In the `fixest` package, you can use the `fixef()` function to extract the fixed effects from a fitted model. Here's how you can do it:

```
# Extract fixed effects  
fixed_effects_simple_model1 <- fixef(simple_model1)  
fixed_effects_simple_model2 <- fixef(simple_model2)  
  
# Print fixed effects  
print(fixed_effects_simple_model1)  
print(fixed_effects_simple_model2)
```

In this code, `fixef(simple_model1)` and `fixef(simple_model2)` extract the fixed effects from `simple_model1` and `simple_model2`, respectively. The `print()` function is then used to print these fixed effects.

dsanchezp18: Error in `feglm(update(controls_formula, approves__president ~ min_temperature + : Evaluation of the right-hand-side of the formula raises an error: In unclass(e1) + unclass(e2): non-numeric argument to binary operator`

GitHub Copilot: The error message suggests that there's a non-numeric argument being used in a binary operation, which is likely due to the `update()` function in your formula. The `update()` function is used to modify the terms of a formula, but it seems like it's causing an issue here.

If `controls_formula` is a formula that includes other control variables, you can try specifying the formula directly in the `feglm()` function:

```
model1_controls <-
  feglm(approves_president ~ min_temperature + control_var1 + control_var2,
        data = df,
        family = binomial(link = "logit"),
        cluster = ~ canton_dpa)
```

In this code, `control_var1` and `control_var2` should be replaced with the actual control variables in your `controls_formula`. If there are more control variables, you can add them to the formula with `+`.

If `controls_formula` is a character string that represents the control variables, you can convert it to a formula with `as.formula()`:

```
model1_controls <-
  feglm(as.formula(paste("approves_president ~ min_temperature +", controls_formula)),
        data = df,
        family = binomial(link = "logit"),
        cluster = ~ canton_dpa)
```

In this code, `paste("approves_president ~ min_temperature +", controls_formula)` creates a character string that represents the full formula, and `as.formula()` converts this character string to a formula.

dsanchezp18: what is glue

GitHub Copilot: The `glue` package in R is used for string interpolation. The `glue()` function evaluates the expressions within curly braces `{}` in a string and replaces them with their results. This is useful for dynamically constructing strings.

Here's a simple example:

```
name <- "John"
age <- 30

glue::glue("My name is {name} and I am {age} years old.")
```

This will output: My name is John and I am 30 years old.

In your code, however, `glue` doesn't seem to be used. If you want to concatenate strings, you can use the `paste()` function, which is what you're currently doing. If you want to use `glue` instead, you can do something like this:

```

model1_controls <-
  feglm(glue::glue("approves_president ~ min_temperature {controls_formula}") %>% as.formula(),
        data = df,
        family = binomial(link = "logit"),
        cluster = ~ canton_dpa)

```

In this code, `glue::glue("approves_president ~ min_temperature {controls_formula}")` constructs a string that includes the value of `controls_formula`. The `%>% as.formula` part then converts this string to a formula.

dsanchezp18: is glue a package

GitHub Copilot: Yes, `glue` is a package in R. It is designed to make it easy to interpolate (“glue”) your data into strings. It works similarly to `paste()`, but with a more intuitive syntax and better handling of inputs.

Here’s a simple example of how to use `glue`:

```

library(glue)

name <- "Alice"
age <- 25

# Use glue to interpolate variables into a string
message <- glue("My name is {name} and I am {age} years old.")
print(message)

```

In this example, the `{name}` and `{age}` inside the string are replaced with the values of the `name` and `age` variables. The output would be: `My name is Alice and I am 25 years old.`

You can install the `glue` package in R with the command `install.packages("glue")`.

dsanchezp18: interaction terms in R

GitHub Copilot: In R, interaction terms in a model formula can be specified using the `*` or `:` operators.

- The `*` operator creates main effects and interaction terms. For example, `y ~ x*z` is equivalent to `y ~ x + z + x:z`, which includes the main effects of `x` and `z` and the interaction term `x:z`.
- The `:` operator creates only the interaction term and does not include the main effects. For example, `y ~ x:z` includes only the interaction term `x:z`.

In your code, you're using the `:` operator to create interaction terms between `region` and other variables:

```
model1_hetero <-  
  feglm(paste("approves_president ~ region: + max_temperature + precipitation + region * (min_temperature + max_temperature + precipitation)",  
    data = df,  
    family = binomial(link = "logit"),  
    cluster = ~ canton_dpa)
```

However, there seems to be a mistake in the formula. The `region:` part is not complete because it doesn't specify what `region` should interact with. If you want to include interaction terms between `region` and `min_temperature`, `max_temperature`, and `precipitation`, you can do:

```
model1_hetero <-  
  feglm(paste("approves_president ~ region * (min_temperature + max_temperature + precipitation)",  
    data = df,  
    family = binomial(link = "logit"),  
    cluster = ~ canton_dpa)
```

In this code, `region * (min_temperature + max_temperature + precipitation)` creates interaction terms between `region` and `min_temperature`, `max_temperature`, and `precipitation`.

dsanchezp18: how to do a title page in Quarto

GitHub Copilot: In Quarto, the title page is automatically created from the metadata specified in the YAML header of your document. The title, subtitle, and author fields you've already specified will be used to create the title page.

```
---  
title: "POL837 Research Paper"  
subtitle: "POL837 Research Paper"  
author: "Daniel Sánchez Pazmiño"  
format: pdf  
---
```

If you want to add more information to the title page, you can use the `abstract` and `date` fields:

```

---
title: "POL837 Research Paper"
subtitle: "POL837 Research Paper"
author: "Daniel Sánchez Pazmiño"
date: "2024-03-19"
abstract: "This is the abstract of the paper."
format: pdf
---

```

In this code, `date: "2024-03-19"` adds the current date to the title page, and `abstract: "This is the abstract of the paper."` adds an abstract. The `2024-03-19` part is an inline R code chunk that gets the current date.

Please note that the appearance of the title page can depend on the output format and the template used.

dsanchezpz18: add page numbers to a quarto pdf number

GitHub Copilot: To add page numbers to a Quarto PDF document, you can use the `header-includes` field in the YAML header to include LaTeX commands that set the page style to include page numbers.

Here's how you can modify your existing YAML header:

```

---
title: "My Research Paper Title"
subtitle: "POL837 Research Paper"
date: "2024-03-08"
date-format: "MMMM YYYY"
author: "Daniel Sánchez Pazmiño"
format: pdf
pdf-engine: xelatex
mainfont: Times New Roman
monofont: Times New Roman
mathfont: "Libertinus Math"
include-in-header:
  text: |
    \usepackage{lipsum}
    \newfontfamily\tfont{Times New Roman}
    \addtokomafont{title}{\tfont}
    \newfontfamily\sfont{Times New Roman}
    \addtokomafont{section}{\sfont}
header-includes:
  - \usepackage{fancyhdr}

```

```

- \pagestyle{fancy}
- \fancyhf{}
- \fancyfoot[C]{\thepage}
---
```

In this code, the `header-includes` field includes several LaTeX commands:

- `\usepackage{fancyhdr}` loads the `fancyhdr` package, which allows you to customize the headers and footers.
- `\pagestyle{fancy}` sets the page style to `fancy`, which allows you to customize the headers and footers.
- `\fancyhf{}` clears the headers and footers.
- `\fancyfoot[C]{\thepage}` adds the page number to the center of the footer. The `[C]` specifies that the page number should be centered, and `\thepage` is the LaTeX command for the current page number.

dsanchezpl18: left of the page footer

GitHub Copilot: To place the page number on the left side of the footer, you can change the `\fancyfoot[R]{\thepage}` line to `\fancyfoot[L]{\thepage}` in the `header-includes` field of your YAML header. The `[L]` specifies that the page number should be on the left.

Here's how your YAML header should look:

```

---
title: "My Research Paper Title"
subtitle: "POL837 Research Paper"
date: "2024-03-08"
date-format: "MMMM YYYY"
author: "Daniel Sánchez Pazmiño"
format: pdf
pdf-engine: xelatex
mainfont: Times New Roman
monofont: Times New Roman
mathfont: "Libertinus Math"
include-in-header:
  text: |
    \usepackage{lipsum}
    \newfontfamily\tfont{Times New Roman}
    \addtokomafont{title}{\tfont}
    \newfontfamily\sfont{Times New Roman}
    \addtokomafont{section}{\sfont}
header-includes:
- \usepackage{fancyhdr}
```

```

- \pagestyle{fancy}
- \fancyhf{}
- \fancyfoot[L]{\thepage}
# reference-doc: "apa_format_reference_doc.docx"
# bibliography: "one_pager_bibliography.bib"
# csl: "apa_one_pager.csl"
# suppress-bibliography: true
---
```

In this code, `\fancyfoot[L]{\thepage}` places the page number on the left side of the footer.

dsanchezp18: it wont change

GitHub Copilot: I apologize for the confusion. Quarto uses Pandoc under the hood to convert markdown to PDF, and Pandoc uses LaTeX for PDF generation. However, the `fancyhdr` package is not fully compatible with the KOMA-Script classes that Pandoc uses for PDF generation.

Instead, you can use the `scrlayer-scrpage` package, which is fully compatible with KOMA-Script classes. Here's how you can modify your YAML header to use this package and place the page number on the left side of the footer:

```

---
title: "My Research Paper Title"
subtitle: "POL837 Research Paper"
date: "2024-03-08"
date-format: "MMMM YYYY"
author: "Daniel Sánchez Pazmiño"
format: pdf
pdf-engine: xelatex
mainfont: Times New Roman
monofont: Times New Roman
mathfont: "Libertinus Math"
include-in-header:
  text: |
    \usepackage{lipsum}
    \newfontfamily\tfont{Times New Roman}
    \addtokomafont{title}{\tfont}
    \newfontfamily\sfont{Times New Roman}
    \addtokomafont{section}{\sfont}
header-includes:
- \usepackage{scrlayer-scrpage}
- \pagestyle{scrheadings}

```



```

- \clearpairofpagestyles
- \foot[\pagemark]{\pagemark}
# reference-doc: "apa_format_reference_doc.docx"
# bibliography: "one_pager_bibliography.bib"
# csl: "apa_one_pager.csl"
# suppress-bibliography: true
---
```

In this code:

- `\usepackage{scrlayer-scrpage}` loads the `scrlayer-scrpage` package.
- `\pagestyle{scrheadings}` sets the page style to `scrheadings`, which allows you to customize the headers and footers.
- `\clearpairofpagestyles` clears the headers and footers.
- `\foot[\pagemark]{\pagemark}` adds the page number to the outer side of the footer. In a one-sided document, this will be the right side. In a two-sided document, this will be the right side on right-hand pages and the left side on left-hand pages. If you want the page number on the left side regardless of the page being a right-hand or left-hand page, you can use `\ifoot[\pagemark]{\pagemark}` instead.

dsanchezp18: look for the first folder in my computer which has a certain string in it with r

GitHub Copilot: You can use the `list.files()` function in R with the `recursive = TRUE` argument to search for folders in your computer. Here's a function that searches for the first folder that contains a certain string:

```

find_folder <- function(search_string, path = ".") {
  # List all folders recursively
  folders <- list.files(path, recursive = TRUE, full.names = TRUE, include.dirs = TRUE)

  # Find the first folder that contains the search string
  folder <- folders[grepl(search_string, folders)][1]

  return(folder)
}

# Use the function
find_folder("your_search_string", "C:/")
```

In this code, replace `"your_search_string"` with the string you're searching for and `"C:/"` with the path where you want to start the search. The function will return the path of the first folder that contains the search string.

Please note that this function can take a long time to run if you're searching a large directory tree. Also, the function uses case-sensitive matching, so make sure to use the correct case for your search string.

dsanchezp18: add more to the header of the quarto document

GitHub Copilot: To add more to the header of your Quarto document, you can add more fields to the YAML header. For example, you can add an **abstract** field to include an abstract, or a **keywords** field to include keywords.

Here's how you can modify your existing YAML header to include an abstract and keywords:

```
---
title: "My Research Paper Title"
subtitle: "POL837 Research Paper"
date: "2024-03-08"
date-format: "MMMM YYYY"
author: "Daniel Sánchez Pazmiño"
abstract: "This is the abstract of my research paper."
keywords: "keyword1, keyword2, keyword3"
format: pdf
pdf-engine: xelatex
mainfont: Times New Roman
monofont: Times New Roman
mathfont: "Libertinus Math"
include-in-header:
  text: |
    \usepackage{lipsum}
    \newfontfamily\tfont{Times New Roman}
    \addtokomafont{title}{\tfont}
    \newfontfamily\sfont{Times New Roman}
    \addtokomafont{section}{\sfont}
    \usepackage{scrlayer-scrpage}
    \lofoot{\thepage}
# reference-doc: "apa_format_reference_doc.docx"
# bibliography: "one_pager_bibliography.bib"
# csl: "apa_one_pager.csl"
# suppress-bibliography: true
knitr:
  opts_chunk:
    echo : false
    message: false
---
```

In this code, `abstract: "This is the abstract of my research paper."` adds an abstract to the document, and `keywords: "keyword1, keyword2, keyword3"` adds keywords. You can replace `"This is the abstract of my research paper."` and `"keyword1, keyword2, keyword3"` with your actual abstract and keywords.

Please note that the appearance of the abstract and keywords can depend on the output format and the template used.

dsanchezp18: how to add an instructor name

GitHub Copilot: You can add an instructor's name to the YAML header of your Quarto document by creating a new field, such as `instructor`. Here's how you can modify your existing YAML header to include an instructor's name:

```
---
title: "My Research Paper Title"
subtitle: "POL837 Research Paper"
date: "2024-03-08"
date-format: "MMMM YYYY"
author: "Daniel Sánchez Pazmiño (301560593)"
instructor: "Professor Eline de Roij"
keywords: "Professor Eline de Roij"
format: pdf
pdf-engine: xelatex
mainfont: Times New Roman
monofont: Times New Roman
mathfont: "Libertinus Math"
include-in-header:
  text: |
    \usepackage{lipsum}
    \newfontfamily\tfont{Times New Roman}
    \addtokomafont{title}{\tfont}
    \newfontfamily\sfont{Times New Roman}
    \addtokomafont{section}{\sfont}
    \usepackage{scrlayer-scrpage}
    \lofoot{\thepage}
# reference-doc: "apa_format_reference_doc.docx"
# bibliography: "one_pager_bibliography.bib"
# csl: "apa_one_pager.csl"
# suppress-bibliography: true
knitr:
  opts_chunk:
    echo : false
    message: false
```

```
warning: false
abstract: "This is the abstract of the paper. It should be a brief
---
```

In this code, `instructor: "Professor Eline de Roij"` adds the instructor's name to the document. You can replace `"Professor Eline de Roij"` with your actual instructor's name.

Please note that the appearance of the instructor's name can depend on the output format and the template used. The `instructor` field is not a standard field in Quarto or Pandoc, so it may not appear in the output document unless your template is designed to include it.

`dsanchezp18`: do not number sections in Quarto

GitHub Copilot: To prevent section numbering in a Quarto document, you can add `number_sections: false` to the YAML header of your document. Here's how you can modify your existing YAML header to include this:

```
---
title: "My Research Paper Title"
subtitle: "POL837 Research Paper"
date: "2024-03-08"
date-format: "MMMM YYYY"
author: "Daniel Sánchez Pazmiño"
instructor: "Professor

E

line de Roij"
format: pdf
pdf-engine: xelatex
mainfont: Times New Roman
monofont: Times New Roman
mathfont: "Libertinus Math"
number_sections: false
include-in-header:
  text: |
    \usepackage{lipsum}
    \newfontfamily\tfont{Times New Roman}
    \addtokomafont{title}{\tfont}
    \newfontfamily\sfont{Times New Roman}
    \addtokomafont{section}{\sfont}
    \usepackage{scrlayer-scrpage}
    \lofoot{\thepage}
```

```

# reference-doc: "apa_format_reference_doc.docx"
# bibliography: "one_pager_bibliography.bib"
# csl: "apa_one_pager.csl"
# suppress-bibliography: true
knitr:
  opts_chunk:
    echo : false
    message: false
abstract: "This is the abstract of the paper. It should be a brief
---

```

In this code, `number_sections: false` prevents sections from being numbered.

dsanchezp18: change the type of numbering to roman numbers

GitHub Copilot: To change the section numbering to Roman numerals in your Quarto document, you can add LaTeX commands to the `include-in-header` field in the YAML header. Here's how you can modify your existing YAML header to include this:

```

---
title: "My Research Paper Title"
subtitle: "POL837 Research Paper"
date: "2024-03-08"
date-format: "MMMM YYYY"
author: "Daniel Sánchez Pazmiño (301560593)"
format: pdf
pdf-engine: xelatex
number-sections: true
mainfont: Times New Roman
monofont: Times New Roman
mathfont: "Libertinus Math"
include-in-header:
  text: |
    \usepackage{lipsum}
    \newfontfamily\tfont{Times New Roman}
    \addtokomafont{title}{\tfont}
    \newfontfamily\sfont{Times New Roman}
    \addtokomafont{section}{\sfont}
    \addtokomafont{subsection}{\sfont}
    \usepackage{scrlayer-scrpage}
    \lofoot{\thepage}
    \renewcommand{\thesection}{\Roman{section}}
# reference-doc: "apa_format_reference_doc.docx"

```

```
# bibliography: "one_pager_bibliography.bib"
# csl: "apa_one_pager.csl"
# suppress-bibliography: true
knitr:
  opts_chunk:
    echo : false
    message: false
---
```

In this code, `\renewcommand{\thesection}{\Roman{section}}` changes the section numbering to Roman numerals. Please note that this will affect all sections in your document. If you want to change the numbering for subsections or subsubsections, you can use `\thesubsection` or `\thesubsubsection` instead of `\thesection`.

dsanchezpl8: eliminate the page numbering from the first page

GitHub Copilot: To eliminate the page numbering from the first page in your Quarto document, you can add the `\thispagestyle{empty}` LaTeX command to the `include-before` field in the YAML header. This command will remove the page number from the first page of your document. Here's how you can modify your existing YAML header to include this:

```
---
title: "My Research Paper Title"
subtitle: "POL837 Research Paper"
date: "2024-03-08"
date-format: "MMMM YYYY"
author: "Daniel Sánchez Pazmiño (301560593)"
format: pdf
pdf-engine: xelatex
number-sections: true
mainfont: Times New Roman
monofont: Times New Roman
mathfont: "Libertinus Math"
include-in-header:
  text: |
    \usepackage{lipsum}
    \newfontfamily\tfont{Times New Roman}
    \addtokomafont{title}{\tfont}
    \newfontfamily\sfont{Times New Roman}
    \addtokomafont{section}{\sfont}
    \addtokomafont{subsection}{\sfont}
    \renewcommand{\thesection}{\Roman{section}}
    \renewcommand{\thesubsection}{\Alph{subsection}}
```

```

include-before:
  text: |
    \thispagestyle{empty}
# reference-doc: "apa_format_reference_doc.docx"
# bibliography: "one_pager_bibliography.bib"
# csl: "apa_one_pager.csl"
# suppress-bibliography: true
knitr:
  opts_chunk:
    echo : false
    message: false
    warning: false
---
```

In this code, `include-before: \thispagestyle{empty}` removes the page number from the first page. Please note that this will only affect the first page of your document. If you want to remove the page number from other pages, you can use the `\pagestyle{empty}` command instead.

dsanchezp18: this is a mistake

GitHub Copilot: I'm sorry for any confusion. Could you please clarify what the mistake is or what you need assistance with? I'm here to help with software development topics, particularly related to your Quarto document or any other programming issues you might have.

dsanchezp18: footnotes in the first page of the document, but not mapped to any particular word

GitHub Copilot: To add footnotes on the first page of your Quarto document that are not mapped to any particular word, you can use the `\footnotetext{}` LaTeX command. This command allows you to add a footnote without a marker. Here's how you can modify your existing LaTeX code to include this:

```

```\tex
\thispagestyle{empty}

\begin{center}

POL837 Issues in Comparative Politics

Spring 2024

\textbf{Student ID}: 301560593
```

```

Professor Eline de Rooij, Ph.D.

\textbf{Word count}: XXXX

\vspace{0.5cm}

\end{center}

\begin{abstract}

\lipsum[1]

\end{abstract}

\footnotetext{This is the first footnote.}
\footnotetext{This is the second footnote.}

\clearpage

```

In this code, `\footnotetext{This is the first footnote.}` and `\footnotetext{This is the second footnote.}` add two footnotes to the first page. You can replace "This is the first footnote." and "This is the second footnote." with your actual footnotes.

Please note that these footnotes will appear at the bottom of the first page, and they will not have a marker linking them to any particular word in the text.

dsanchezp18: eliminate the 0 footnote that appears.

GitHub Copilot: The footnote number is automatically generated by LaTeX and it starts from 1 by default. If a footnote with number 0 is appearing, it might be due to some custom settings or commands in your document.

However, if you want to remove the footnote number completely, you can use the `\footnotetext*{}` command instead of `\footnotetext{}`. The asterisk (\*) after `\footnotetext` tells LaTeX to not print a footnote marker.

Here's how you can modify your existing LaTeX code to include this:

```

\footnotetext*{I thank Alonso Quijano-Ruiz, M.A. for providing valuable guidance on the empiri

```

In this code, `\footnotetext*{I thank Alonso Quijano-Ruiz, M.A. for providing valuable guidance on the empirical approach and the algorithm to obtain daily weather data from the U.S. government National Oceanic and Atmospheric Administration (NOAA).}` adds a footnote without a marker.



dsanchezp18: insert a horizontal line

GitHub Copilot: To insert a horizontal line in your Quarto document, you can use the `\hrulefill` LaTeX command. This command creates a horizontal line that fills the available horizontal space. Here's how you can modify your existing LaTeX code to include this:

```
\textbf{Word count}: XXXX

\vspace{0.5cm}

\end{center}

\begin{abstract}

\lipsum[1]

\end{abstract}

\rule

\footnotesize{I thank Alonso Quijano-Ruiz, M.A. for providing valuable guidance on the empir

\hrulefill

# Introduction

# Background

# Empirical Approach

## Data
```

In this code, `\hrulefill` inserts a horizontal line after the footnote.

dsanchezp18: writ a `sprintf` statement to format with two decimals and a thousands separator

GitHub Copilot: In R, you can use the `sprintf` function to format numbers. However, `sprintf` does not directly support thousands separators. You can use the `formatC` function instead, which supports both decimal places and thousands separators. Here's an example:

```
# Define a number
num <- 123456.789
```

```
# Format the number with two decimal places and a thousands separator
formatted_num <- formatC(num, format = "f", digits = 2, big.mark = ",")

# Print the formatted number
print(formatted_num)
```

In this code, `formatC(num, format = "f", digits = 2, big.mark = ",")` formats the number `num` with two decimal places and a thousands separator. The `format = "f"` argument specifies fixed-point formatting (i.e., not scientific notation), the `digits = 2` argument specifies two decimal places, and the `big.mark = ","` argument specifies a comma as the thousands separator.

dsanchezp18: use fmt functions to format with thousands

GitHub Copilot: In R, you can use the `scales` package to format numbers with thousands separators. The `comma` function from this package formats numbers by adding a comma as the thousands separator. Here's how you can modify your existing R code to include this:

```
# Load the scales library
library(scales)

# Define a number
num <- 123456.789

# Format the number with a thousands separator
formatted_num <- comma(num)

# Print the formatted number
print(formatted_num)
```

In this code, `comma(num)` formats the number `num` with a thousands separator. The `comma` function automatically rounds to the nearest whole number. If you want to keep decimal places, you can specify the number of decimal places with the `accuracy` argument, like this: `comma(num, accuracy = 0.01)`.

However, in your provided code, if you want to apply this formatting to the output of your models, you would need to create a custom function and apply it to the `fmt` argument in the `modelsummary` function. Here's an example:

```
# Define a custom formatting function
comma_format <- function(x) scales::comma(x)

# Apply the custom formatting function to the modelsummary function
```

```

modelsummary(simple_models,
  stars = stars,
  output = "latex",
  booktabs = TRUE,
  title = "Baseline Models",
  add_rows = rows_baseline_models,
  vcov = ~ canton_dpa,
  escape = T,
  gof_map = gf_map,
  fmt = comma_format)

```

In this code, `comma_format <- function(x) scales::comma(x)` defines a custom formatting function that formats numbers with a thousands separator, and `fmt = comma_format` applies this custom formatting function to the output of the `modelsummary` function.

dsanchezp18: do a clustering with modelsummary

GitHub Copilot: To perform clustering with `modelsummary`, you can use the `vcov` argument to specify a function that computes the variance-covariance matrix of the model's coefficients. If you want to cluster standard errors by a specific variable, you can use the `vcov::vcovCL` function from the `sandwich` package.

Here's how you can modify your existing R code to include this:

```

# Load the sandwich library
library(sandwich)

# Define a list of models
simple_models <- list(
  "Model 1" = model1,
  "Model 2" = model2,
  "Model 3" = model3,
  "Model 4" = model4,
  "Model 5" = model5
)

# Define the stars argument
stars <- c("*" = .1, "***" = .05, "****" = .01)

# Define the add_rows argument
rows_baseline_models <- tribble(
  ~v1, ~v2, ~v3, ~v4, ~v5, ~v6,
  "Std. errors", "By canton", "By canton", "By canton", "By canton", "By canton"
)

```

```

)

# Define the gof_map argument
gof_map <- list(
  "N" = "Observations",
  "R2" = "R-squared",
  "R2 Adj." = "Adjusted R-squared",
  "AIC" = "AIC",
  "BIC" = "BIC"
)

# Apply the modelsummary function with clustered standard errors
modelsummary(simple_models,
  stars = stars,
  output = "latex",
  booktabs = TRUE,
  fmt = fmt_decimal(3),
  title = "Baseline Models",
  add_rows = rows_baseline_models,
  vcov = function(x) vcovCL(x, cluster = data$cluster_var),
  escape = F,
  gof_map = gof_map)

```

In this code, `vcov = function(x) vcovCL(x, cluster = data$cluster_var)` specifies that the standard errors should be clustered by the variable `cluster_var` in the data frame `data`. You should replace `data$cluster_var` with your actual clustering variable.

dsanchezp18: create a function to display with thousands separator and no decimals with `formatC`

GitHub Copilot: In R, you can create a custom function that uses the `formatC` function to format numbers with a thousands separator and no decimal places. Here's how you can do this:

```

# Define a custom formatting function
format_thousands <- function(x) {
  formatC(x, format = "f", digits = 0, big.mark = ",")
}

# Test the function
num <- 123456.789
formatted_num <- format_thousands(num)
print(formatted_num)

```

In this code, `format_thousands <- function(x) { formatC(x, format = "f", digits = 0, big.mark = ",") }` defines a custom formatting function that formats numbers with a thousands separator and no decimal places. The `format = "f"` argument specifies fixed-point formatting (i.e., not scientific notation), the `digits = 0` argument specifies no decimal places, and the `big.mark = ","` argument specifies a comma as the thousands separator.

dsanchezp18: degree symbol in R

GitHub Copilot: In R, you can include the degree symbol in your strings using the unicode character `\u00B0`. Here's how you can modify your existing R code to include this:

```
# Define the coefficient names
coefficients_baseline_models <- c(
  "min_temperature" = "Min. temperature (\u00B0C)",
  "max_temperature" = "Max. temperature (\u00B0C)",
  "avg_temperature" = "Avg. temperature (\u00B0C)",
  "precipitation" = "Precipitation (mm)"
)
```

In this code, `\u00B0` is the unicode character for the degree symbol. It is included in the strings `"Min. temperature (\u00B0C)"`, `"Max. temperature (\u00B0C)"`, and `"Avg. temperature (\u00B0C)"` to represent degrees Celsius.

dsanchezp18: change the spacing of the document

GitHub Copilot: To change the spacing of your Quarto document, you can use the `geometry` and `setspace` LaTeX packages. The `geometry` package allows you to change the margins of your document, and the `setspace` package allows you to change the line spacing of your document.

Here's how you can modify your existing Quarto code to include this:

```
```\{=tex\}
\usepackage[margin=1in]{geometry}
\usepackage{setspace}
\onehalfspacing
```

In this code, `\usepackage[margin=1in]{geometry}` loads the `geometry` package with a margin of 1 inch, `\usepackage{setspace}` loads the `setspace` package, and `\onehalfspacing` sets the line spacing to one and a half.

You can replace `1in` with your desired margin size, and you can replace `\onehalfspacing` with `\doublespacing` for double line spacing or `\singlespacing` for single line spacing.

Please note that these commands should be placed at the beginning of your document, before the `\begin{document}` command. If you're using a Quarto document, you can place these commands in a code chunk at the beginning of your document, like this:

```
```\tex
\usepackage[margin=1in]{geometry}
\usepackage{setspace}
\onehalfspacing
```

dsanchezp18: can this be done in the yaml without latex

GitHub Copilot: Yes, you can set the document's spacing and margins directly in the YAML header.

```
```\quarto
---
title: "Your Document Title"
output:
  html_document:
    css: styles.css
    includes:
      in_header: header.tex
geometry: margin=1in
linestretch: 1.5
---
```

In this YAML header, `geometry: margin=1in` sets the margin to 1 inch, and `linestretch: 1.5` sets the line spacing to one and a half.

Please note that these settings might not work for all output formats. If you're creating a PDF document, you might need to use the `geometry` and `setspace` LaTeX packages as I described in my previous message. If you're creating an HTML document, you might need to use CSS to set the margins and line spacing.

dsanchezp18: use lapply to apply the avg\_slopes function to a list called simple\_models

GitHub Copilot: In R, you can use the `lapply` function to apply a function to each element of a list. Here's how you can modify your existing R code to apply the `avg_slopes` function to the `simple_models` list:

```
# Apply the avg_slopes function to the simple_models list
apes_baseline <- lapply(simple_models, avg_slopes)
```

In this code, `lapply(simple_models, avg_slopes)` applies the `avg_slopes` function to each element of the `simple_models` list. The result is a list of the same length as `simple_models`, where each element is the result of applying `avg_slopes` to the corresponding element of `simple_models`. The result is assigned to the `apes_baseline` variable.

dsanchezp18: use one argument, `type = "response"` for the `avg_slopes`

GitHub Copilot: In R, you can use the `lapply` function to apply a function with additional arguments to each element of a list. Here's how you can modify your existing R code to apply the `avg_slopes` function with the `type = "response"` argument to the `simple_models` list:

```
# Apply the avg_slopes function to the simple_models list
apes_baseline <- lapply(simple_models, avg_slopes, type = "response")
```

In this code, `lapply(simple_models, avg_slopes, type = "response")` applies the `avg_slopes` function with the `type = "response"` argument to each element of the `simple_models` list. The result is a list of the same length as `simple_models`, where each element is the result of applying `avg_slopes` with the `type = "response"` argument to the corresponding element of `simple_models`. The result is assigned to the `apes_baseline` variable.

dsanchezp18: use `scales` to format with decimals, 2

GitHub Copilot: In R, you can use the `scales` package to format numbers with a specified number of decimal places. The `scales::number_format()` function allows you to specify the number of decimal places with the `accuracy` argument. Here's how you can modify your existing R code to include this:

```
# Load the scales library
library(scales)

# Plot the slopes
plot_slopes(simple_model5, variables = "max_temperature", condition = "max_temperature") +
  theme_minimal() +
  labs(x = "Max. Temperature (\u00B0C)",
       y = "Marginal Probability of Presidential Approval") +
  scale_x_continuous(breaks = seq(0, 40, 5), labels = scales::number_format(accuracy = 0.01)) +
  scale_y_continuous(labels = scales::percent_format()) +
  theme(legend.position = "none")
```

In this code, `scales::number_format(accuracy = 0.01)` formats numbers with two decimal places. It is used as the `labels` argument to the `scale_x_continuous` function, so the x-axis labels will be formatted with two decimal places.

dsanchezp18: change the confidence region color in ggplot

GitHub Copilot: In the `plot_slopes` function from the `jtools` package, the confidence region is controlled by the `ribbon` aesthetic. You can change the color of the confidence region by setting the `fill` aesthetic in the `geom_ribbon` function.

However, `plot_slopes` does not provide a direct way to modify the `geom_ribbon` aesthetics. You will need to modify the `plot_slopes` function to add this functionality.

Here's how you can modify your existing R code to change the color of the confidence region:

```
# Load the necessary libraries
library(jtools)
library(ggplot2)

# Define a custom plot_slopes function
plot_slopes_custom <- function(model, ...) {
  # Call the original plot_slopes function
  p <- plot_slopes(model, ...)

  # Modify the plot
  p <- p +
    geom_ribbon(aes(ymin = lower, ymax = upper), fill = "your_color", alpha = 0.2)

  # Return the modified plot
  return(p)
}

# Use the custom plot_slopes function
plot_slopes_custom(simple_model5,
                   variables = "max_temperature",
                   condition = "max_temperature",
                   type = ~ canton_dpa) +
  theme_minimal() +
  labs(x = "Max. Temperature (\u00B0C)",
       y = "Marginal Probability of Presidential Approval") +
  scale_y_continuous(labels = scales::percent_format()) +
  theme(legend.position = "none")
```

In this code, `geom_ribbon(aes(ymin = lower, ymax = upper), fill = "your_color", alpha = 0.2)` changes the color of the confidence region to `your_color`. You should replace `your_color` with your desired color. The `alpha = 0.2` argument makes the confidence region semi-transparent. You can adjust this value to your liking.



dsanchezp18: do directly from ggplot

GitHub Copilot: To directly modify the confidence region color in ggplot, you can use the `geom_ribbon` function. However, the `plot_slopes` function from the `jtools` package does not return the data frame used for plotting, which is needed for `geom_ribbon`.

You would need to manually calculate the confidence intervals and plot them using `ggplot` and `geom_ribbon`. Here's a simplified example of how you might do this:

```
# Load the necessary libraries
library(ggplot2)
library(broom)

# Get the model's predictions and confidence intervals
preds <- augment(simple_model5, newdata = data.frame(max_temperature = seq(min(data$max_temper

# Plot the predictions and confidence intervals
ggplot(preds, aes(x = max_temperature, y = .fitted)) +
  geom_line() +
  geom_ribbon(aes(ymin = .lower, ymax = .upper), fill = "your_color", alpha = 0.2) +
  theme_minimal() +
  labs(x = "Max. Temperature (\u00B0C)",
       y = "Marginal Probability of Presidential Approval") +
  scale_y_continuous(labels = scales::percent_format()) +
  theme(legend.position = "none")
```

In this code, `augment(simple_model5, newdata = data.frame(max_temperature = seq(min(data$max_temperature), max(data$max_temperature), length.out = 100)))` calculates the model's predictions and confidence intervals for a sequence of `max_temperature` values. `geom_ribbon(aes(ymin = .lower, ymax = .upper), fill = "your_color", alpha = 0.2)` plots the confidence intervals with your desired color. You should replace `your_color` with your desired color.

Please note that this is a simplified example. You might need to adjust this code to match your specific data and model.

dsanchezp18: examine the underlying code of an object

GitHub Copilot: In R, you can use the `getAnywhere` function to examine the underlying code of a function. However, for objects like plots, data frames, or models, the underlying code is not directly accessible because these objects are the result of executing code, not code themselves.

If you want to examine the structure of an object, you can use the `str` function. Here's how you can modify your existing R code to include this:

```
# Examine the structure of the plot object
str(a)
```

In this code, `str(a)` prints the structure of the `a` object. If `a` is a plot object, this will include information about the plot's layers, scales, and other components.

If you want to examine the code that created an object, you will need to look at the code that was executed to create the object. In your case, the `a` object was created by the `plot_slopes` function call in your code.

dsanchezp18: change the font size in the quarto document

GitHub Copilot: In Quarto, you can set the font size of your document directly in the YAML header. Here's how you can modify your existing YAML header to include this:

```
---
title: "My Research Paper Title"
subtitle: "POL837 Research Paper"
author: "Daniel Sánchez Pazmiño"
format: pdf
pdf-engine: xelatex
fontsize: 12pt
number-sections: true
linestretch: 2
mainfont: Times New Roman
monofont: Times New Roman
mathfont: "Libertinus Math"
include-in-header:
  text: |
    \usepackage{lipsum}
    \newfontfamily\tfont{Times New Roman}
    \addtokomafont{title}{\tfont}
    \newfontfamily\sfont{Times New Roman}
    \addtokomafont{section}{\sfont}
    \addtokomafont{subsection}{\sfont}
    \renewcommand{\thesection}{\Roman{section}}
    \renewcommand{\thesubsection}{\Alph{subsection}}
    \usepackage{booktabs}
    \usepackage{siunitx}
    \newcolumntype{d}{S[input-symbols = ()]}
    \usepackage{amsmath, amsfonts}
    \usepackage{float}
knitr:
---
```

In this YAML header, `fontsize: 12pt` sets the font size to 12 points. You can replace `12pt` with your desired font size. Please note that this setting might not work for all output formats. If you're creating a PDF document, you might need to use the `geometry` LaTeX package as I described in my previous message. If you're creating an HTML document, you might need to use CSS to set the font size.

`dsanchezpl8`: redefine table numbers within a section to be A.1, A.2

GitHub Copilot: In LaTeX, which is used by Quarto for PDF output, you can redefine the table numbering within a section to use the section number followed by the table number. Here's how you can modify your existing Quarto document to include this:

```
---
title: "My Research Paper Title"
subtitle: "POL837 Research Paper"
author: "Daniel Sánchez Pazmiño"
format: pdf
pdf-engine: xelatex
fontsize: 12pt
number-sections: true
linestretch: 2
mainfont: Times New Roman
monofont: Times New Roman
mathfont: "Libertinus Math"
include-in-header:
  text: |
    \usepackage{chngcntr}
    \counterwithin{table}{section}
    \renewcommand{\thetable}{\Alph{section}.\arabic{table}}
    \usepackage{lipsum}
    \newfontfamily\tfont{Times New Roman}
    \addtokomafont{title}{\tfont}
    \newfontfamily\sfont{Times New Roman}
    \addtokomafont{section}{\sfont}
    \addtokomafont{subsection}{\sfont}
    \renewcommand{\thesection}{\Roman{section}}
    \renewcommand{\thesubsection}{\Alph{subsection}}
    \usepackage{booktabs}
    \usepackage{siunitx}
    \newcolumntype{d}{S[input-symbols = ()]}
    \usepackage{amsmath, amsfonts}
    \usepackage{float}
knitr:
---
```

In this YAML header, `\counterwithin{table}{section}` makes the table counter reset with each new section, and `\renewcommand{\thetable}{\Alph{section}.\arabic{table}}` changes the table number format to use the section number followed by the table number. The `\Alph{section}` command formats the section number as a capital letter, and the `\arabic{table}` command formats the table number as an Arabic numeral.

dsanchezp18: reduce line spacing kable

GitHub Copilot: In `kable`, you can reduce line spacing by using the `linesep` argument in the `kable_styling` function. This argument allows you to specify the space between lines in the table. Here's how you can modify your existing R code to include this:

```
modelsummary(models_controls,
  gof_map = gf_map,
  coef_map = coefficients_controls,
  stars = stars,
  estimate = "{estimate}{stars}",
  booktabs = TRUE,
  align = "lrrrr",
  threeparttable = TRUE,
  title = "Logit coefficients for specifications with controls",
  vcov = ~ canton_dpa,
  notes = notes_control_models,
  output = "latex",
  threeparttable = TRUE,
  longtable = T) %>%
kable_styling(latex_options = "hold_position", linesep = "")
```

In this code, `linesep = ""` removes the space between lines in the table. You can replace `""` with your desired space. For example, `linesep = "\\smallskipamount"` would set the space to a small skip amount. Please note that this setting might not work for all output formats. If you're creating a PDF document, you might need to use the `arraystretch` LaTeX command to adjust the line spacing. If you're creating an HTML document, you might need to use CSS to set the line spacing.

dsanchezp18: what is the default arraystretch in latex

GitHub Copilot: The default value of `\arraystretch` in LaTeX is 1. This means that the height of each row in the table is 1 times the height of the text in the row. If you increase `\arraystretch`, you will increase the spacing between rows in the table. Conversely, if you decrease `\arraystretch`, you will decrease the spacing between rows.

For example, `\renewcommand{\arraystretch}{1.5}` would increase the row spacing by 50%, and `\renewcommand{\arraystretch}{0.5}` would decrease the row spacing by 50%.

In your document, `\renewcommand{\arraystretch}{1}` is resetting the row spacing to the default value.

dsanchezp18: reset count table

GitHub Copilot: In LaTeX, you can reset the table counter using the `\setcounter` command. Here's how you can modify your existing Quarto document to include this:

```
\setcounter{table}{0}
```

In this code, `\setcounter{table}{0}` resets the table counter to 0. The next table will be Table 1. You can place this command in your Quarto document where you want the table counter to reset. For example, you might place it at the start of a new section.

dsanchezp18: also set the counter for sections to 0

GitHub Copilot: You can reset the section counter in LaTeX using the `\setcounter` command, similar to how you reset the table counter. Here's how you can modify your existing Quarto document to include this:

```
\setcounter{section}{0}
```

In this code, `\setcounter{section}{0}` resets the section counter to 0. The next section will be Section 1. You can place this command in your Quarto document where you want the section counter to reset. For example, you might place it at the start of a new appendix.

dsanchezp18: what are other ways i can call a regressor other than x

GitHub Copilot: In the context of regression models, a regressor (also known as an independent variable, predictor, feature, or explanatory variable) can be referred to in various ways depending on the context or field of study. Here are a few alternatives:

1. Covariate: This term is often used in statistics and econometrics to refer to variables that are possibly predictive of the outcome under study.
2. Factor: In the context of factorial experiments or when dealing with categorical predictors, the term "factor" is often used.
3. Attribute: This term is often used in the field of machine learning and data mining.
4. Input: In the context of machine learning models, predictors are often referred to as inputs.
5. Feature: This is another term commonly used in machine learning and data mining.
6. Dimension: In multivariate analysis, each variable can be considered a dimension in the data space.

Remember, the choice of term often depends on the field of study and the context in which the analysis is being done.

dsanchezp18: in terms of notation, other than x

GitHub Copilot: In terms of mathematical notation for regression models, there are several alternatives to using “x” for the independent variables or regressors. Here are a few examples:

1. You can use other letters from the alphabet, such as “v”, “w”, “z”, etc.
2. You can use Greek letters like “ $\alpha$ ”, “ $\beta$ ”, “ $\gamma$ ”, etc.
3. You can use subscripted letters to denote different variables, like “x1”, “x2”, “x3”, etc.
4. You can use descriptive labels that reflect the nature of the variable, like “income”, “age”, “temperature”, etc.
5. In the context of matrix notation for multiple regression, the matrix of all independent variables is often denoted as “X”, with each column representing a different variable.

Remember, the choice of notation should ideally make the model easier to understand, so it’s often best to choose notation that is descriptive and consistent with the conventions of your field.

dsanchezp18: what other letters in the alphabet

GitHub Copilot: In the context of your regression model equation, you can use any other letter from the alphabet to represent your variables, as long as they are not already being used in your equation. Here are a few examples:

1. “p”, “q”, “r”, “s”, “t”, “u”, “v”, “w”, “z” could be used to represent additional variables or parameters.
2. If you have many variables, you can use subscripted letters to denote different variables, like “p1”, “p2”, “p3”, etc.
3. You can also use capital letters like “P”, “Q”, “R”, “S”, “T”, “U”, “V”, “W”, “Z”.

Remember, the choice of notation should ideally make the model easier to understand, so it’s often best to choose notation that is descriptive and consistent with the conventions of your field.

dsanchezp18:  $\times$  in unicode

GitHub Copilot: The Unicode for the multiplication sign ( $\times$ ) is U+00D7. In HTML, you can use `&#215;` or `&times;` to represent it. In a string in most programming languages, you can use the escape sequence `\u00D7` to represent it.

dsanchezp18: landscape mode package for latex

GitHub Copilot: The `lscape` package is commonly used in LaTeX to change the orientation of a page to landscape. You've already included it in your Quarto document in the `include-in-header` section:

```
\usepackage{lscape}
```

To use it to make a page landscape, you would typically use the `landscape` environment, like this:

```
\begin{landscape}  
% Your content here  
\end{landscape}
```

This will make the content within the `landscape` environment appear in landscape orientation. Note that this will work for PDF output, but not for HTML or other output formats. For those formats, you would need to use a different method to change the page orientation.

dsanchezp18: landscape mode package for xelatex

GitHub Copilot: For XeLaTeX, you can use the `lscape` package to change the orientation of a page to landscape. Here's how you can include it in your Quarto document:

```
\usepackage{lscape}
```

To use it to make a page landscape, you would typically use the `landscape` environment, like this:

```
\begin{landscape}  
% Your content here  
\end{landscape}
```

This will make the content within the `landscape` environment appear in landscape orientation. Note that this will work for PDF output, but not for HTML or other output formats. For those formats, you would need to use a different method to change the page orientation.

However, if you want to rotate the page including headers and footers, you should use the `pdflscape` package instead. The usage is the same as `lscape`, but it only works with pdfLaTeX and not with XeLaTeX or LuaLaTeX.

dsanchezp18: modify facet titles in a ggplot

GitHub Copilot: To modify facet titles in a ggplot, you can use the `labeller` argument in the `facet_wrap` or `facet_grid` function. Here's how you can modify your existing R code to include this:

```
library(ggplot2)
library(jtools)

# Generate the plot
plot <- plot_slopes(model4_hetero, variables = c("max_temperature", "min_temperature") , condition = "ideology")

# Modify the facet titles
plot + facet_wrap(~ideology, labeller = as_labeller(c(`0` = "Conservative", `1` = "Liberal")))
```

In this code, `as_labeller(c(0= "Conservative",1= "Liberal"))` changes the facet title for `ideology = 0` to “Conservative” and the facet title for `ideology = 1` to “Liberal”. You can replace “Conservative” and “Liberal” with your desired facet titles.

Please note that `plot_slopes` function from `jtools` package might not return a ggplot object, so you might need to modify the function or use a different function to generate the plot.

dsanchezp18: `plot_slopes` is from marginal effects

GitHub Copilot: Yes, you’re correct. The `plot_slopes()` function is from the `margins` package in R. This function is used to visualize the marginal effects of variables in a regression model, which are the changes in the predicted outcome for a one-unit change in the predictor variable, holding all other variables constant.

In your code, `plot_slopes(model4_hetero, variables = c("max_temperature", "min_temperature") , condition = "ideology")` is used to plot the marginal effects of `max_temperature` and `min_temperature` on the outcome variable in the `model4_hetero` model, conditioned on the `ideology` variable.

If you want to modify the facet labels in this plot, you can use the `facet_labels` argument in the `plot_slopes()` function. Here’s how you can modify your existing R code to include this:

```
plot_slopes(model4_hetero, variables = c("max_temperature", "min_temperature") , condition = "ideology") +
  theme_minimal() +
  labs(x = "Degrees (\u00B0C)",
       y = "Marginal Probability of Presidential Approval")
```

In this code, `facet_labels = c("Conservative", "Liberal")` changes the facet labels to “Conservative” and “Liberal”. You can replace “Conservative” and “Liberal” with your desired facet labels.

dsanchezp18: Speaking in general of the current administration, how would you rate the job performance of President count the words

GitHub Copilot: Sorry, but I can only assist with programming related questions.



dsanchezp18: how to do a quote in latex

GitHub Copilot: In LaTeX, you can use the `quote` environment to create a block quote. Here's how you can modify your existing Quarto document to include this:

```
\begin{quote}
Speaking in general of the current administration, how would you rate the job performance of
\end{quote}
```

In this code, `\begin{quote}` starts the quote environment, and `\end{quote}` ends it. The text within this environment will be indented on both sides to set it apart from the rest of the text.

dsanchezp18: inline quote

GitHub Copilot: In LaTeX, you can use the `textquote` command from the `csquotes` package to create an inline quote. Here's how you can modify your existing Quarto document to include this:

First, include the `csquotes` package in your preamble:

```
\usepackage{csquotes}
```

Then, use the `textquote` command to create an inline quote:

```
The primary explained variable of interest is presidential job approval, which the AmericasB
```

In this code, `\textquote{}` creates an inline quote. The text within the curly braces will be quoted.

dsanchezp18: include a page number in a quarto citation

GitHub Copilot: In Quarto (which uses Pandoc for rendering), you can include a page number in a citation by using the `@` symbol followed by the citation key and the page number in square brackets. Here's how you can modify your existing Quarto document to include this:

```
\textquote{Speaking in general of the current administration, how would you rate the job per
```

In this code, `[@ABDatasets, p. 123]` cites the `ABDatasets` reference and specifies page 123. You can replace 123 with your desired page number. This will create a citation like “(AB-Datasets, 123)” in the rendered document. The exact format of the citation will depend on the citation style you're using.

dsanchezp18: how to make the latex output of a chunk smaller

GitHub Copilot: To make the output of a chunk smaller in LaTeX, you can wrap the chunk output in a `scriptsize` or `footnotesize` environment. Here's how you can modify your existing Quarto document to include this:

```
\begin{landscape}
\begin{footnotesize}

```{r .cell-code}
# Prepare relevant variables for all years

df_descriptive <-
  full_df %>%
  transmute(`Presidential approval` = approves_president,
            `Daily minimum temperature (C)` = min_temperature,
            `Daily maximum temperature (C)` = max_temperature,
            `Daily average temperature (C)` = avg_temperature,
            `Daily precipitation (mm)` = precipitation,
            `Female` = sex,
            `Age (years)` = age,
            `Rural status` = urban_rural,
            `Education` = education,
            `Labour market status` = labour_market,
            `Worse perception of personal economy` = personal_econ_situation,
            `Worse perception of country economy` = country_econ_situation,
            `Ideology score (0-10)` = ideology,
```

```
# Your code to print the
```

```
dsanchezp18: that is not working
```

GitHub Copilot: I apologize for the confusion. In Quarto, you can adjust the size of the output

However, the exact method to do this depends on the type of output you're generating. If you

Here's an example:

```
```r
```

```
library(knitr)
library(kableExtra)
```

```
# Assume df is your data frame
```

```
kable(df, "latex", booktabs = TRUE) %>%  
  kable_styling(font_size = 10)
```

In this code, `kable(df, "latex", booktabs = TRUE)` generates a LaTeX table from the data frame `df`, and `kable_styling(font_size = 10)` adjusts the font size of the table to 10pt.

If you're generating a different type of output, the method to adjust the font size might be different. If you could provide more information about the type of output you're generating, I could give a more specific answer.

dsanchezp18: degree in latex

GitHub Copilot:

dsanchezp18: degree in latex

GitHub Copilot:

dsanchezp18: how would i calculate the number of words in a quarto doc

GitHub Copilot:

dsanchezp18: how would i calculate the number of words in a quarto doc

GitHub Copilot: To calculate the number of words in a Quarto document, you can use the `texcount` utility, which is a Perl script that counts words in LaTeX documents. Here's how you can do it:

1. First, you need to compile your Quarto document to a LaTeX file. You can do this by running `quarto render your_document.qmd --to latex` in the terminal, replacing `your_document.qmd` with the name of your Quarto document. This will create a LaTeX file with the same name as your Quarto document.
2. Then, you can run `texcount your_document.tex -inc` in the terminal to count the words in the LaTeX document, replacing `your_document.tex` with the name of your LaTeX document. The `-inc` option tells 'texcount

dsanchezp18: greek values

GitHub Copilot:

dsanchezp18: how do i create a pdf file which binds together all the pdfs

GitHub Copilot: To create a single PDF file that binds together all the PDFs, you can use the `pdfunite` command-line tool, which is part of the Poppler utilities for manipulating PDF files. Here's how you can do it:

1. Open the terminal in Visual Studio Code by selecting **View > Terminal** from the menu, or by pressing 'Ctrl+' (backtick).

2. Navigate to the directory containing your PDF files using the `cd` command. For example, if your PDF files are in a directory named `pdfs` on your desktop, you

dsanchezp18: in quarto

GitHub Copilot: To combine multiple PDFs into a single PDF in Quarto, you can simply list all the source files you want to include in the `project.render` section of your `_quarto.yml` file. Quarto will then render these files in the order they are listed and combine them into a single PDF.

Here's an example:

```
project:
  title: "POL837 Research Paper"
  render:
    - paper-text/sections/introduction.qmd
    - paper-text/sections/empirical_strategy.qmd
    - paper-text/sections/section3.qmd
    - paper-text/sections/section4.qmd
    - paper-text/sections/conclusion.qmd

format:
  pdf:
    fontsize: 12pt
    pdf-engine: xelatex

output-file: "sanchez_daniel_POL837_research_paper_first_draft.pdf"

bibliography: paper-text/references.bib
csl: paper-text/apa7_normal.csl
```

In this example, `section3.qmd`, `section4.qmd`, and `conclusion.qmd` are additional Quarto markdown files that will be rendered and included in the final PDF. You should replace these with the actual paths to your source files.

When you run `quarto render` in the terminal, Quarto will render each of these files and combine them into a single PDF named `sanchez_daniel_POL837_research_paper_first_draft.pdf`.

dsanchezp18: only the last is being included

GitHub Copilot: I apologize for the confusion. The `project.render` field in the `_quarto.yml` file is used to specify the order of rendering for multiple documents, but it doesn't combine the output of these documents into a single PDF.

To combine multiple PDFs into a single PDF, you would need to use an external tool. One option is to use the `pdfpages` package in LaTeX. You can create a new LaTeX document that uses the `\includepdf` command to include each of your PDFs. Here's an example:

```
\documentclass{article}
\usepackage{pdfpages}

\begin{document}
\includepdf[pages=--]{file1.pdf}
\includepdf[pages=--]{file2.pdf}
\includepdf[pages=--]{file3.pdf}
\end{document}
```

In this example, replace `file1.pdf`, `file2.pdf`, and `file3.pdf` with the paths to your PDF files. The `pages=--` option includes all pages from each PDF.

You can then compile this LaTeX document to create a single PDF that includes all of your PDFs. Note that this requires having a LaTeX distribution installed on your system.

If you're working in a Windows environment and have access to PowerShell, you can also use the `PDFtk` command-line tool to merge PDFs. Here's an example command:

```
pdftk file1.pdf file2.pdf file3.pdf cat output combined.pdf
```

In this example, replace `file1.pdf`, `file2.pdf`, and `file3.pdf` with the paths to your PDF files. The `cat` option concatenates the PDFs, and `output combined.pdf` specifies the name of the output file. Note that this requires having `PDFtk` installed on your system.