figure-code.R

mollyofferwestort

2025-10-24

```r
# Load necessary libraries
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyr)
library(ggplot2)
library(distributional)
library(ggdist)
library(estimatr)
library(ggthemes)

data <- read.csv("../data/data_with_additional_vars.csv")

# Colorblind-friendly palette
cbPalette <-c("#999999","#E69F00", "#56B4E9","#009E73",
              "#F0E442", "#0072B2", "#D55E00","#CC79A7")

# Colorblind-friendly palette
cbPalette_c <- c( '#125A56', '#00767B', '#238F9D', '#42A7C6', '#60BCE9', '#9DCCEF', '#C6DBED', '#DEE6E7


ssi3_theme <- theme(
  legend.text = element_text(size = 12),
  panel.grid.major = element_blank(),
  panel.grid.minor = element_blank(),
  plot.margin = margin(t = 20, r = 20, b = 20, l = 20),
  axis.ticks.x = element_line(size = 0.5),
  axis.ticks.y = element_line(size = 0.5),
  axis.ticks.length = unit(5, "pt"),
  legend.position = 'bottom')
```

```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
## i Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
```

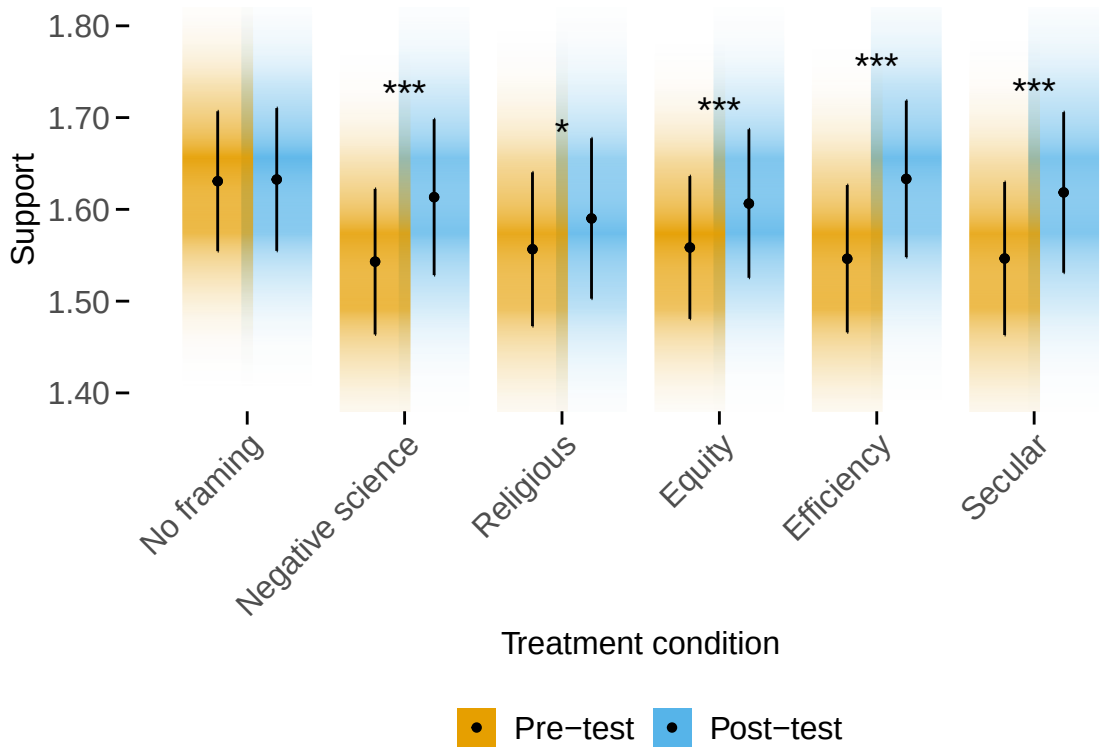
```
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
# Define treatments
treatments <- c("No framing", "Negative science", "Religious", "Equity", "Efficiency", "Secular")
covariates_pre <- c("gastax", "carbtax", "treaty", "regcarb")

# Recode treatment variable
data <- data |>
  mutate(treatment_frame = factor(treatment_value, labels = treatments))

stars_model <- estimatr::lm_robust(I(post_test-pre_test) ~ treatment_frame - 1, data) |>
  estimatr::tidy() |>
  mutate(star = case_when(
    p.value < 0.001 ~ "***",
    p.value < 0.01 ~ "**",
    p.value < 0.05 ~ "*",
    p.value < 0.1 ~ "+",
    TRUE ~ ""
  ),
  term = gsub("treatment_frame", "", term))

# Reshape data to long format for pre-test and post-test responses
data_long <- data |>
  select(treatment_frame, pre_test, post_test) |>
  pivot_longer(cols = c(pre_test, post_test), names_to = "test", values_to = "response") |>
  group_by(treatment_frame, test) |>
  summarise(
    mean = mean(response, na.rm = TRUE),
    sd = sd(response, na.rm = TRUE)/sqrt(n())
  ) |>
  mutate(test = factor(test, levels = c("pre_test", "post_test"),
                       labels = c("Pre-test", "Post-test")))
```

```
## `summarise()` has grouped output by 'treatment_frame'. You can override using
## the `.groups` argument.
```

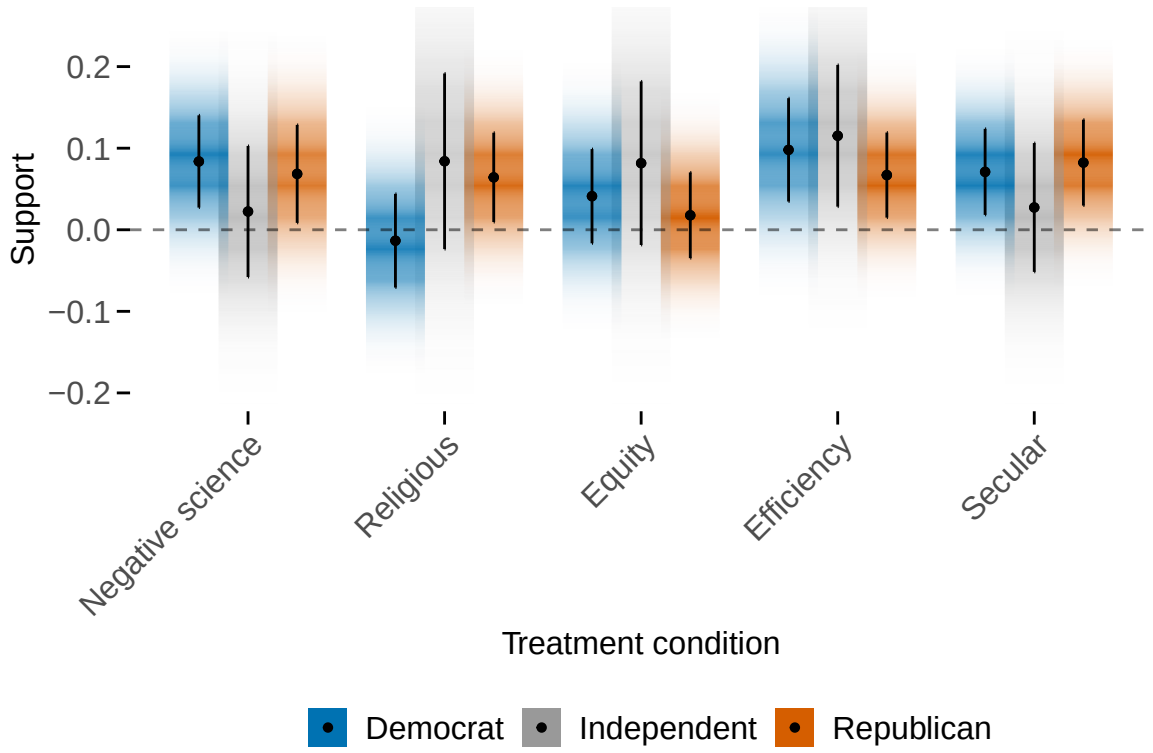
```
# Calculate the midpoint between pre-test and post-test for each treatment
data_mid <- data_long |>
  group_by(treatment_frame) |>
  summarise(midpoint = mean(mean),
            test = 'Pre-test')

# Merge significance stars into the data_mid
data_mid <- data_mid |>
  left_join(stars_model, by = c("treatment_frame" = "term"))

# Plot pre-test and post-test response means and standard errors, by treatment condition with stars in
g <- ggplot(data_long, aes(x = treatment_frame, y = mean, fill = test, color = test)) +

  # Gradient Interval
  stat_gradientinterval(
    aes(x = treatment_frame, ydist = distributional::dist_normal(mean, 1.75*sd),
        color = test, fill = test),
    width = 1,
```

```
    position = position_dodge(0.75),
    linewidth = 0,
    point_size = 1.5,
    point_alpha = 1,
    point_color = "black",
    shape = 20,
    interval_alpha = 0,
    show.legend = c(size = FALSE, slab_alpha = FALSE),
    fill_type = 'segments'
  ) +

  # Data points and error bars
  geom_errorbar(
    aes(ymin = mean - 1.96*sd, ymax = mean + 1.96*sd),
    width = 0, position = position_dodge(0.75), linetype = "solid", color = "black"
  ) +

  # Add significance stars at the midpoint between pre-test and post-test
  geom_text(
    data = data_mid,
    aes(label = star, x = treatment_frame, y = midpoint + conf.high + 0.05),  # Position stars slightly
    size = 5,
    color = "black"
  ) +

  # Labels and Titles
  labs(
    y = "Support",  # Updated y-axis label
    x = "Treatment condition"
  ) +

  # Customize Axes
  scale_x_discrete(expand = c(0.15, 0)) +
  scale_y_continuous(breaks = seq(1.4, 1.8, by = 0.1), labels = scales::label_number()) +  # Ensure y-a

  # Theme Settings
  theme_minimal() +
  scale_color_manual(name=element_blank(), values = c(cbPalette[2], cbPalette[3])) +
  scale_fill_manual(name=element_blank(), values = c(cbPalette[2], cbPalette[3])) +
  ssi3_theme +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1, size = 12),
    axis.text.y = element_text(size = 12),
    axis.title = element_text(size = 12),
    strip.text = element_text(size = 20, face = 'bold')
  ) +
coord_cartesian(ylim=c(1.4,1.8)) #limits of plot

# Print the plot
print(g)
```




```
ggsave("../figures/pre_post_plot.png", plot = g,
       width = 10, height = 5, units = "in", dpi = 300)


# Reshape data to long format for pre-test and post-test responses
formula <- as.formula("post_test ~ treatment_frame")
model1_rep <- lm_lin(formula,
                     covariates =
                       formula(paste0("~ ",
                                      paste(covariates_pre, collapse = " + "))),
                     data = data[which(data$party == 1),], se = "HC3")
model1_dem <- lm_lin(formula,
                     covariates =
                       formula(paste0("~ ",
                                      paste(covariates_pre, collapse = " + "))),
                     data = data[which(data$party == -1),], se = "HC3")
model1_ind <- lm_lin(formula,
                     covariates =
                       formula(paste0("~ ",
                                      paste(covariates_pre, collapse = " + "))),
                     data = data[which(data$party == 0),], se = "HC3")


data_long2 <- bind_rows(
  "Republican" = tidy(model1_rep),
  "Democrat" =  tidy(model1_dem),
  "Independent" = tidy(model1_ind),
  .id = "party"
) |>
  filter(!grepl("_c|Intercept", term)) |>
```

```r
  mutate(treatment_frame = factor(gsub("treatment_frame", "", term), levels = treatments[-1]))

# Plot pre-test and post-test response means and standard errors, by treatment condition with stars in
g <- ggplot(data_long2, aes(x = treatment_frame, y = estimate, fill = party, color = party)) +

  # Gradient Interval
  stat_gradientinterval(
    aes(x = treatment_frame, ydist = distributional::dist_normal(estimate, 1.75*std.error),
        color = party, fill = party),
    width = 1,
    position = position_dodge(0.75),
    linewidth = 0,
    point_size = 1.5,
    point_alpha = 1,
    point_color = "black",
    shape = 20,
    interval_alpha = 0,
    show.legend = c(size = FALSE, slab_alpha = FALSE),
    fill_type = 'segments'
  ) +

  # Data points and error bars
  geom_errorbar(
    aes(ymin = estimate - 1.96*std.error, ymax = estimate + 1.96*std.error),
    width = 0, position = position_dodge(0.75), linetype = "solid", color = "black"
  ) +

  # Labels and Titles
  labs(
    y = "Support",
    x = "Treatment condition"
  ) +

  # Customize Axes
  scale_x_discrete(expand = c(0.15, 0)) +
  scale_y_continuous(breaks = seq(-0.5, 1, by = 0.1), labels = scales::label_number()) +  # Ensure y-ax
  geom_hline(yintercept=0, linetype="dashed", alpha=0.5) +
  # Theme Settings
  theme_minimal() +
  scale_color_manual(name=element_blank(), values = c(cbPalette[6], cbPalette[1], cbPalette[7])) +
  scale_fill_manual(name=element_blank(), values = c(cbPalette[6], cbPalette[1], cbPalette[7])) +
  ssi3_theme +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1, size = 12),
    axis.text.y = element_text(size = 12),
    axis.title = element_text(size = 12),
    strip.text = element_text(size = 20, face = 'bold')
  ) +
coord_cartesian(ylim=c(-0.2,0.25)) #limits of plot

# Print the plot
print(g)
```

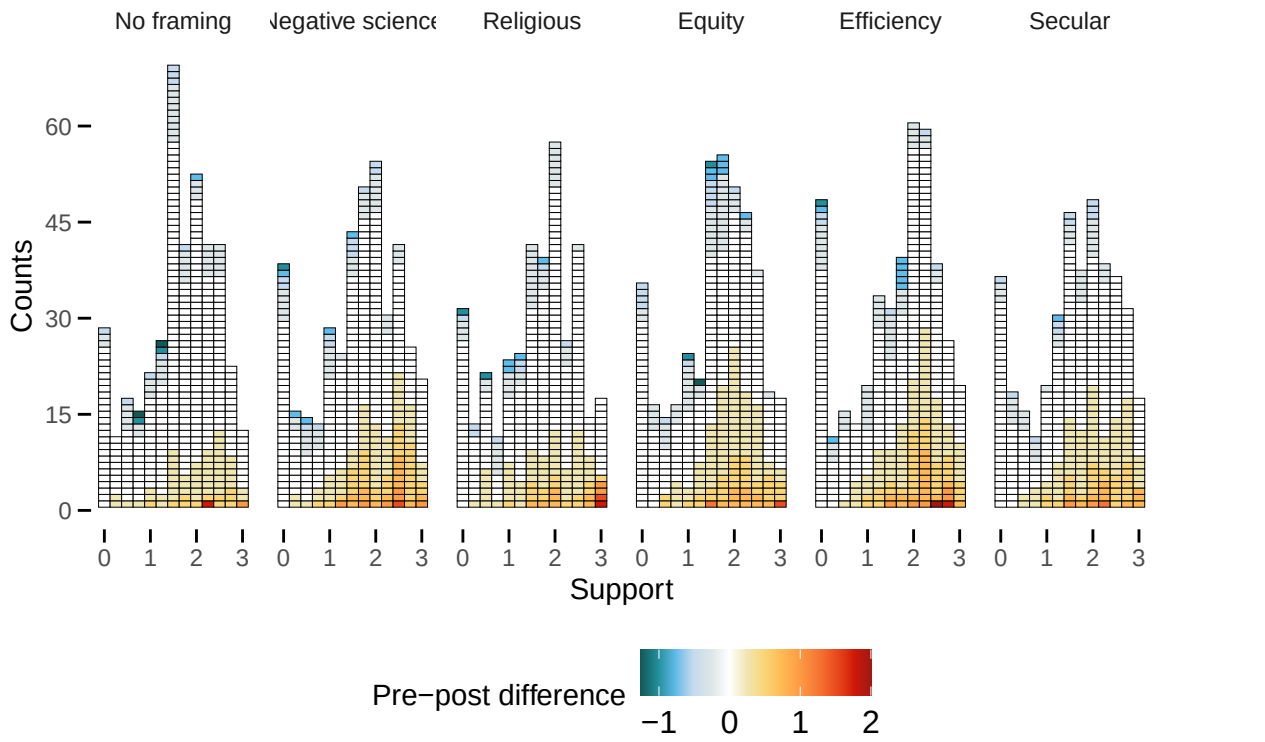



```
ggsave("../figures/party_ate_plot.png", plot = g,
       width = 10, height = 5, units = "in", dpi = 300)


# Recode treatment variable
data <- data |>
  mutate(treatment_frame = factor(treatment_value, labels = treatments),
         diff = post_test - pre_test)

data <- data |>
  group_by(treatment_frame, pre_test) |>
  arrange(pre_test) |>
  mutate(pre_cumulative_count = row_number()) |>
  ungroup() |>
  group_by(treatment_frame, post_test) |>
  arrange(post_test, -diff) |>
  mutate(post_cumulative_count = -1*row_number()) |>
  ungroup()


# Base plot
point_plot <- ggplot(data,
                     aes(x = post_test, y = -post_cumulative_count, fill=diff)) +
  facet_wrap(~ treatment_frame, nrow = 1) +  # Separate plots by treatment
  scale_fill_gradientn(colors = c(cbPalette_c[1], cbPalette_c[3], cbPalette_c[5],
                                  cbPalette_c[7], cbPalette_c[8],
                                  cbPalette_c[9], # middle color
                                  cbPalette_c[10], cbPalette_c[11], cbPalette_c[12],
                                  cbPalette_c[13], cbPalette_c[14], cbPalette_c[15],
                                  cbPalette_c[16], cbPalette_c[17]),
```

```
                        name = "Pre-post difference") +
  geom_tile(aes(width = 0.25, height = 1), color = !is.na(diff)) +
  theme_minimal() +
  ssi3_theme +
  # Labels and Titles
  labs(
    y = "Counts",  # Updated y-axis label
    x = "Support"
  ) +
  # Customize Axes
  scale_y_continuous(breaks = seq(-60, 60, by = 15))
```

```
## Warning in is.na(diff): is.na() applied to non-(list or vector) of type
## 'closure'
```

```
point_plot
```




```
ggsave("../figures/response_distribution_plot.png", plot = point_plot,
       width = 8, height = 6, units = "in", dpi = 300)
```