15 Ways to Visualize Regression Results! (#5 will surprise you!)

A Tutorial in R

Tim Fraser, Northeastern University

April 15, 2021

1. Introduction

Visualization is central to communicating the results of social science studies, but deciding *how* to best visualize the results of a statistical model can be challenging. Below, I introduce 8 ways to visualize your regression results. These visualizations all use statistical simulation, a technique developed by Gary King and colleagues in 2000. This technique is available in SPSS, Stata, and R, and while functions may differ slightly, all visuals made below in R could also be generated using other programs. Let's briefly dig into what that is.

1.1 Statistical Simulation

Statistical simulation injects random variation into the beta coefficients of a statistical model, producing 1000 slightly different sets of beta coefficients. By supplying a set of hypothetical data, usually the traits of an average observation in your dataset, we can then vary just one of those traits, feed them into our 1000 models with 1000 slightly different beta coefficients, and get 1000 slightly different projected outcomes. This approximates estimation uncertainty - the uncertainty inherent in your model. We can even average those outcomes over random error again, so that we account for fundamental uncertainty. The 1000 projected outcomes we produce are called *expected values,* and they represent 1000 very precise estimates of the range of outcomes you would get given the conditions we supplied to the model. While a statistical model outputs just a beta coefficient and a standard error, we have 1000 expected values to deal with! This creates lots of opportunities for interesting and useful visualizations that will capture your reader's eye and allow you to precisely estimate quantities of interest about your data.

1.2 Our packages

To make these simulations, we are going to need the *Zelig* package, and to arrange and visualize them, we need the *tidyverse* package. Finally, we will use the *dataverse* package to access our data.

```
library(Zelig) # for modeling and convenient visualization
library(tidyverse) # for data manipulation
library(dataverse) # for accessing data from the Harvard Dataverse
```

1.2 Our data

So, first, what kind of data are we going to visualize? We're going to borrow from a recent project of mine with colleagues on emissions. This dataset records the kilowatts of renewable energy (solar, wind, geothermal, tidal, microhydro, or biomass) adopted per 1000 residents each municipality in Japan, including towns, cities, and urban

wards, for the years 2005 to 2017. We can download it directly from the Harvard Dataverse using the following code:

```
# First, let's specify we want to download from this server.
Sys.setenv("DATAVERSE_SERVER" = "dataverse.harvard.edu")

# Second, import dataset of emissions from Harvard Dataverse
dataset <- get_dataframe_by_name(
  filename  = "dataset.tab",
  dataset   = "10.7910/DVN/E8PEW7")
```

1.3 Format Data

Finally, we're going to format that data, zooming into just cities and wards in the Tokyo area. We will create our outcome of interest, the ***rate*** of carbon emissions per 1,000 residents. We'll also create a few extra covariates (many are saved in this dataset), including population density.

```
# Next, we're going to overwrite that dataset,
# keeping just a subset of cities located in Tokyo.
dat <- dataset %>%
  # Grab just the cities and wards in Tokyo
  filter(pref == "Tokyo") %>%
  # Calculate rate of consumer emissions per 1000 residents
  mutate(consumer_emissions = emissions_consumer_subtotal / pop * 1000) %>%
   # Calculate population density
  mutate(pop_density = pop / area_inhabitable) %>%
  ungroup() %>%
  # Ensure year and city code are treated as factors for fixed effects
  mutate(year = factor(year),
         muni_code = factor(muni_code)) %>%
  # Finally, let's keep just a few of the many variables:
  select(
    muni, # name of municipality
    muni_code, # each city's five-digit unique identifier code
    year, # year
    consumer_emissions, # Consumer emissions per 1000 residents
    fin_str_index, # government capacity
    # and a bunch of covariates that might also shape emissions, including:
    pop_density, # population density
    income_per_capita, # income per capita
    pop_age_65_plus, # % Age 65 or older
    pop_college, # % graduated college
    value_agr_mill, # agricultural output in millions of yen per capita
    value_manuf_mill, # manufacturing output in millions of yen per capita
    value_commerce_mill # commerical output in millions of yen per capita
  )
```

1.4 A basic model

Finally, we are going to make a basic model. Since emissions is a right-skewed rate, we will use a gamma model (the virtual equivalent of an OLS model with a log-transformed outcome variable).

```
m1 <- dat %>%
  zelig(formula = consumer_emissions ~ fin_str_index +
          pop_density + income_per_capita +
          pop_age_65_plus + pop_college +
          value_agr_mill + value_manuf_mill + value_commerce_mill + year,
        model = "gamma", cite = FALSE)
```

```
## Warning: `tbl_df()` was deprecated in dplyr 1.0.0.
## Please use `tibble::as_tibble()` instead.
```

```
## Warning: `group_by_()` was deprecated in dplyr 0.7.0.
## Please use `group_by()` instead.
## See vignette('programming') for more help
```

```
# Sometimes, beta coefficients can be tricky, especially when they are log-odds; simulat
ing is clearer.
```

2. Visualizations

Next, we are going to create several visualizations using this model! These will generally highlight the powerful, negative effect on consumer emissions per 1000 residents made by strong governance capacity, represented by the financial strength index of a city.

Now, let's set a default theme for the rest of our visuals.

```
mytheme <- theme_set(
  theme_classic(base_size = 14) +
  theme(panel.border = element_rect(fill = NA, color = "black"),
        plot.caption = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5))
)
```

A financial strength index reflects the financial strength of a local government. It is calculated by dividing basic financial revenue by basic financial demand for a city government. A financial strength index over means 1 shows that financial resources of the city are much greater than the taxes allocated ordinarily.

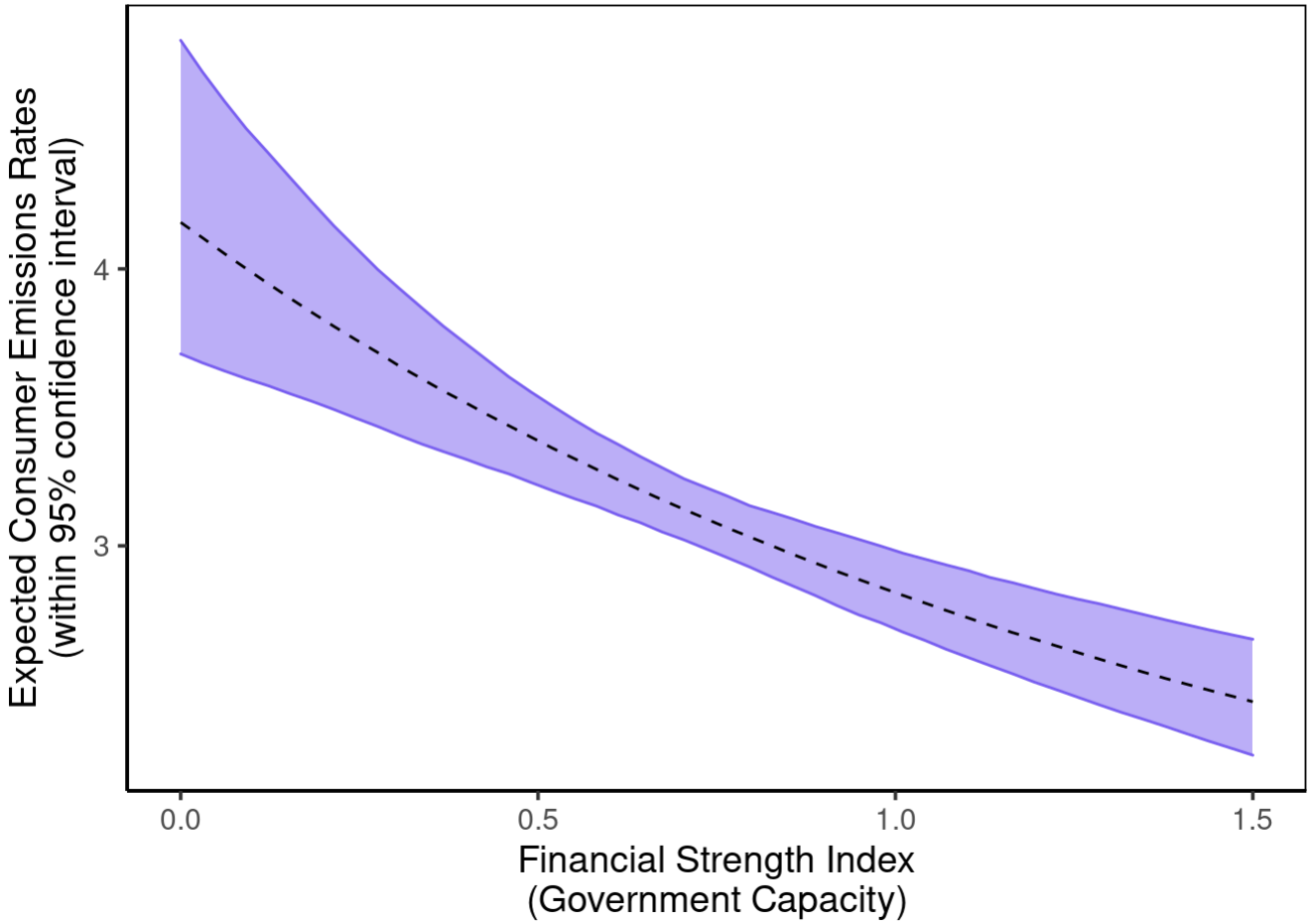
Let's estimate the expected emissions for cities with low, high, and higher government capacity, as measured by financial strength indices between 0 and 1.5.

```
mysim <- m1 %>%
  setx(fin_str_index = seq(from = 0, to = 1.5, length.out = 50)) %>%
  sim() %>%
  zelig_qi_to_df() %>%
  select(ev = expected_value, fin_str_index)
```

2.1 Ribbon Plot of Expected Values

The most common move is to plot a ribbon of expected values, ranging between the 2.5th and 97.5th percentiles, with the median indicated by a line (in this case, a dashed line).

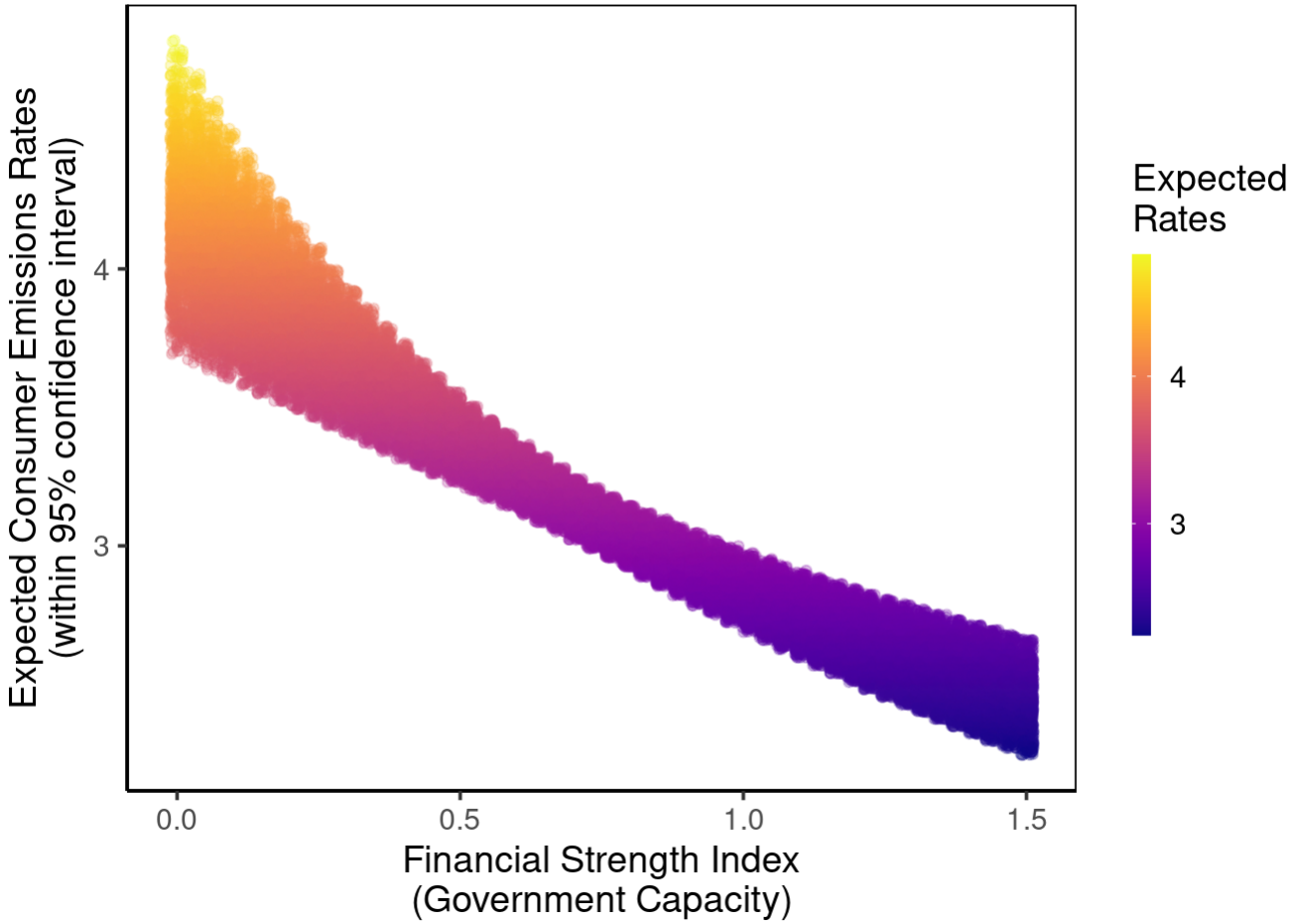
```
mysim %>%
  group_by(fin_str_index) %>%
  summarize(median = median(ev),
            lower = quantile(ev, 0.025),
            upper = quantile(ev, 0.975)) %>%
  ggplot(mapping = aes(x = fin_str_index, y = median,
                       ymin = lower, ymax = upper)) +
  geom_ribbon(alpha = 0.5, color = "#785EF0", fill = "#785EF0") +
  geom_line(linetype = "dashed") +
  labs(x = "Financial Strength Index\n(Government Capacity)",
       y = "Expected Consumer Emissions Rates\n(within 95% confidence interval)")
```



2.2 Scatterplot of Expected Values

Alternatively, we could plot every expected value within the 95% confidence interval (meaning, those greater than 2.5% and less than 97.5% of all values.) It's a little bumpy, but pretty.

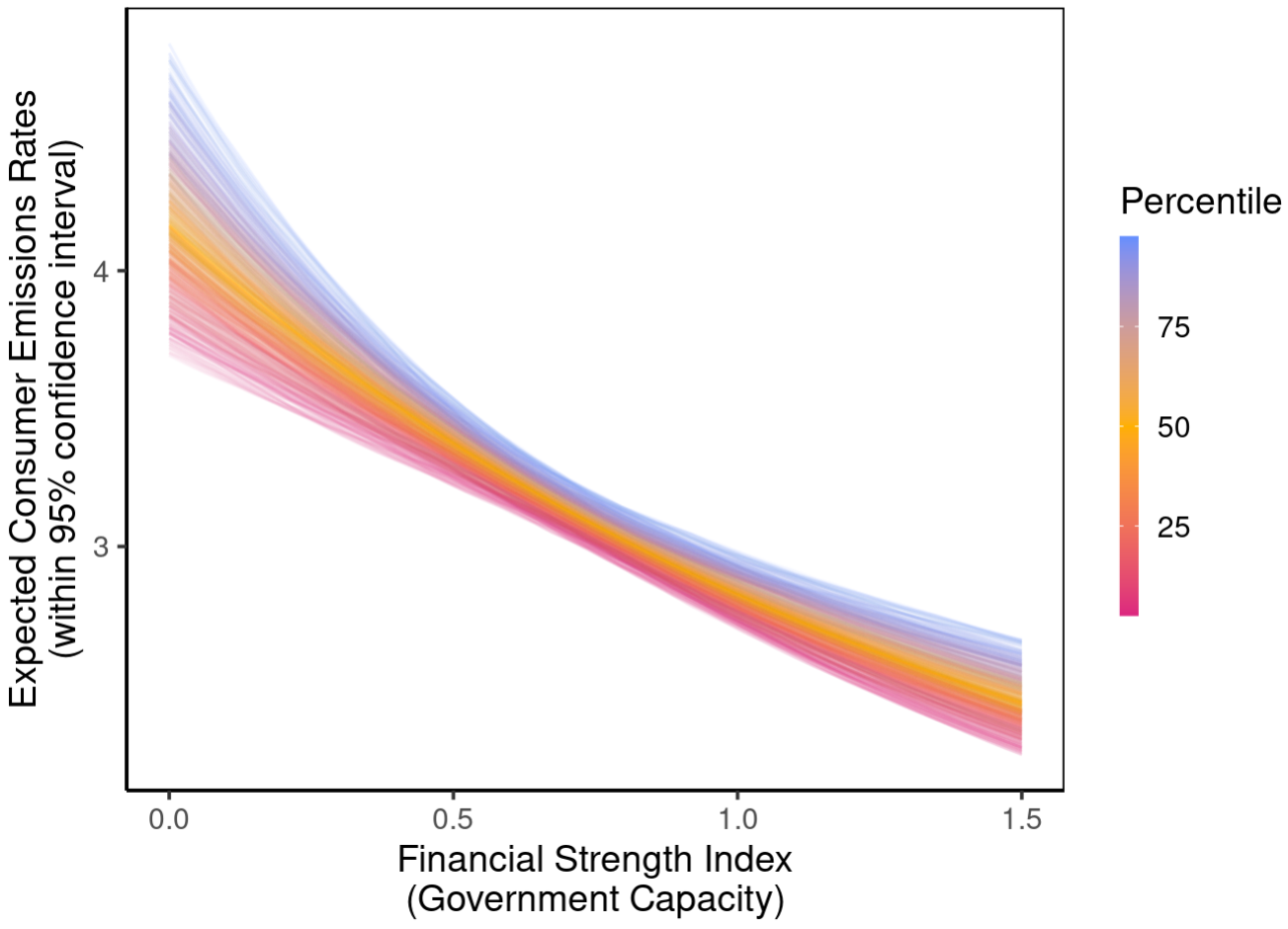
```
mysim %>%
  group_by(fin_str_index) %>%
  filter(ev > quantile(ev, 0.025),
         ev < quantile(ev, 0.975)) %>%
  ungroup() %>%
  ggplot(mapping = aes(x = fin_str_index, y = ev, color = ev)) +
  geom_jitter(alpha = 0.25) +
  viridis::scale_color_viridis(option = "plasma") +
  guides(color = guide_colorbar(barheight = 10, barwidth = 0.5)) +
  labs(x = "Financial Strength Index\n(Government Capacity)",
       y = "Expected Consumer Emissions Rates\n(within 95% confidence interval)",
       color = "Expected\nRates")
```



2.3 Line Plot of Expected Values, by Percentile

Alternatively, we could plot each expected value as a line, colored by percentile! Then, we could show only those greater than 2.5% or less than 97.5% of all values.

```
mysim %>%
  group_by(fin_str_index) %>%
  # Sort and get percentile
  arrange(ev) %>%
  mutate(percentile = 1:n() / 10) %>%
  # Filter within these quantiles
  filter(ev > quantile(ev, 0.025),
         ev < quantile(ev, 0.975)) %>%
  ungroup() %>%
  ggplot(mapping = aes(x = fin_str_index, y = ev, group = percentile, color = percentil
e)) +
  geom_line(alpha = 0.05) +
  scale_color_gradient2(low = "#DC267F", high = "#648FFF", mid = "#FFB000", midpoint = 5
0) +
  #viridis::scale_color_viridis(option = "plasma") +
  guides(color = guide_colorbar(barheight = 10, barwidth = 0.5)) +
  labs(x = "Financial Strength Index\n(Government Capacity)",
       y = "Expected Consumer Emissions Rates\n(within 95% confidence interval)",
       color = "Percentile")
```



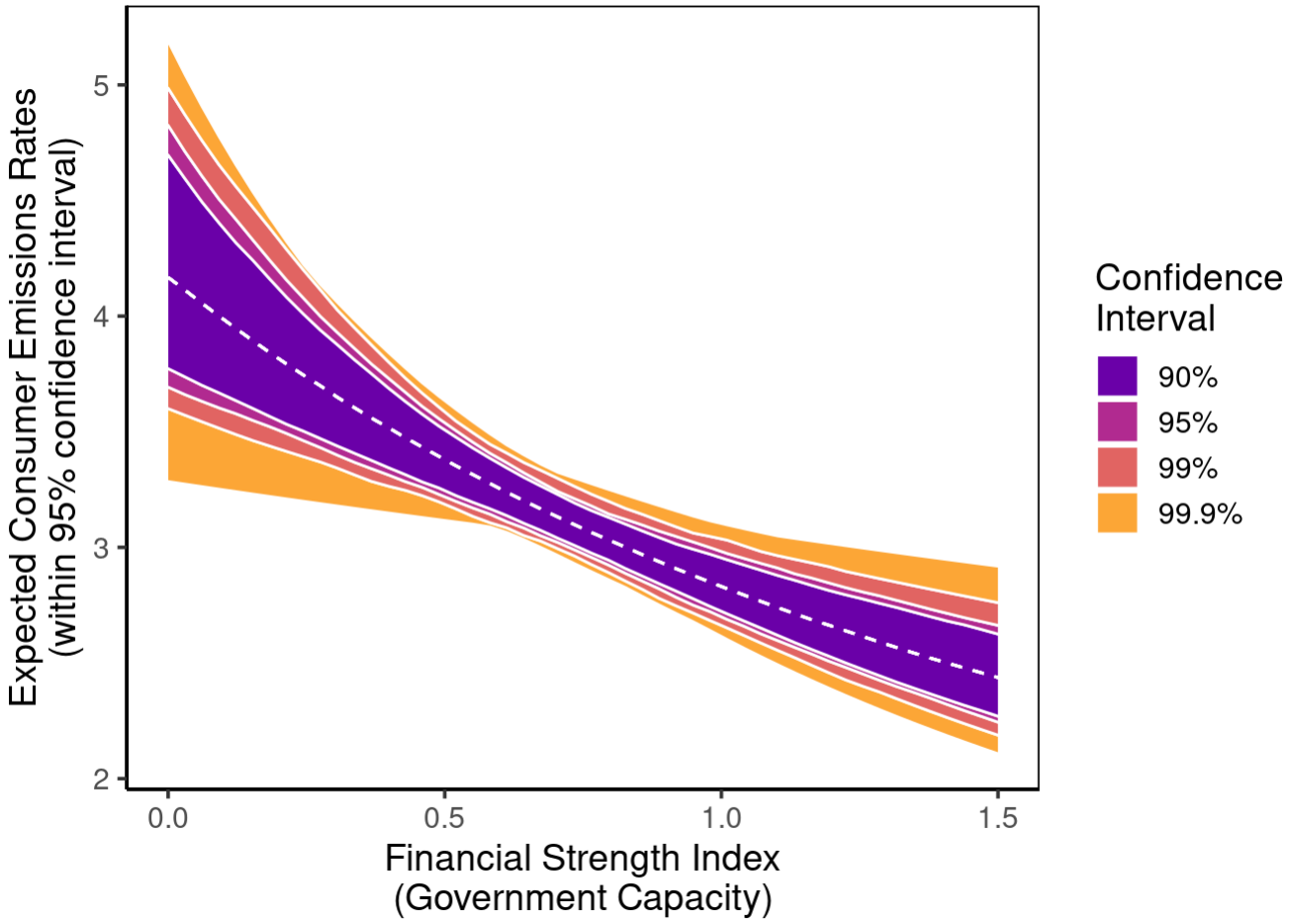
2.4 Ribbon Plot by Level of Confidence

But maybe we like ribbons, but want to show different levels of confidence very clearly. Let’s make a series of bands, one for each interval.

```
out <- bind_rows(
  # 99.9 interval
  mysim %>%
    group_by(fin_str_index) %>%
    summarize(median = median(ev),
              lower = quantile(ev, 0.0005),
              upper = quantile(ev, 0.9995),
              ci = "99_9"),
  # 99 interval
  mysim %>%
    group_by(fin_str_index) %>%
    summarize(median = median(ev),
              lower = quantile(ev, 0.005),
              upper = quantile(ev, 0.995),
              ci = "99"),
  # 95th percentile
  mysim %>%
    group_by(fin_str_index) %>%
    summarize(median = median(ev),
              lower = quantile(ev, 0.025),
              upper = quantile(ev, 0.975),
              ci = "95"),
  # 90th percentile
  mysim %>%
    group_by(fin_str_index) %>%
    summarize(median = median(ev),
              lower = quantile(ev, 0.05),
              upper = quantile(ev, 0.95),
              ci = "90")) %>%
  # Pivot longer into a tidy format
  pivot_longer(cols = c(lower, upper), names_to = "type", values_to = "ev") %>%
  # Arrange in order
  arrange(desc(fin_str_index),ev) %>%
  # Now for each round of simulations,
  group_by(fin_str_index) %>%
  # Grab the median, and make upper and lower bands
  summarize(
    median = median[1],
    lower = c(ev[1], ev[2], ev[3], ev[4], ev[5], ev[6], ev[7]),
    upper = c(ev[2], ev[3], ev[4], ev[5], ev[6], ev[7], ev[8]),
    group = 1:7) %>%
  # And label them
  mutate(ci = group %>% recode_factor(
    "1" = "99.9%",
    "2" = "99%",
    "3" = "95%",
    "4" = "90%",
    "5" = "95%",
    "6" = "99%",
    "7" = "99.9%") %>%
      # And order the intervals
      factor(levels = c("90%", "95%", "99%", "99.9%")))
```

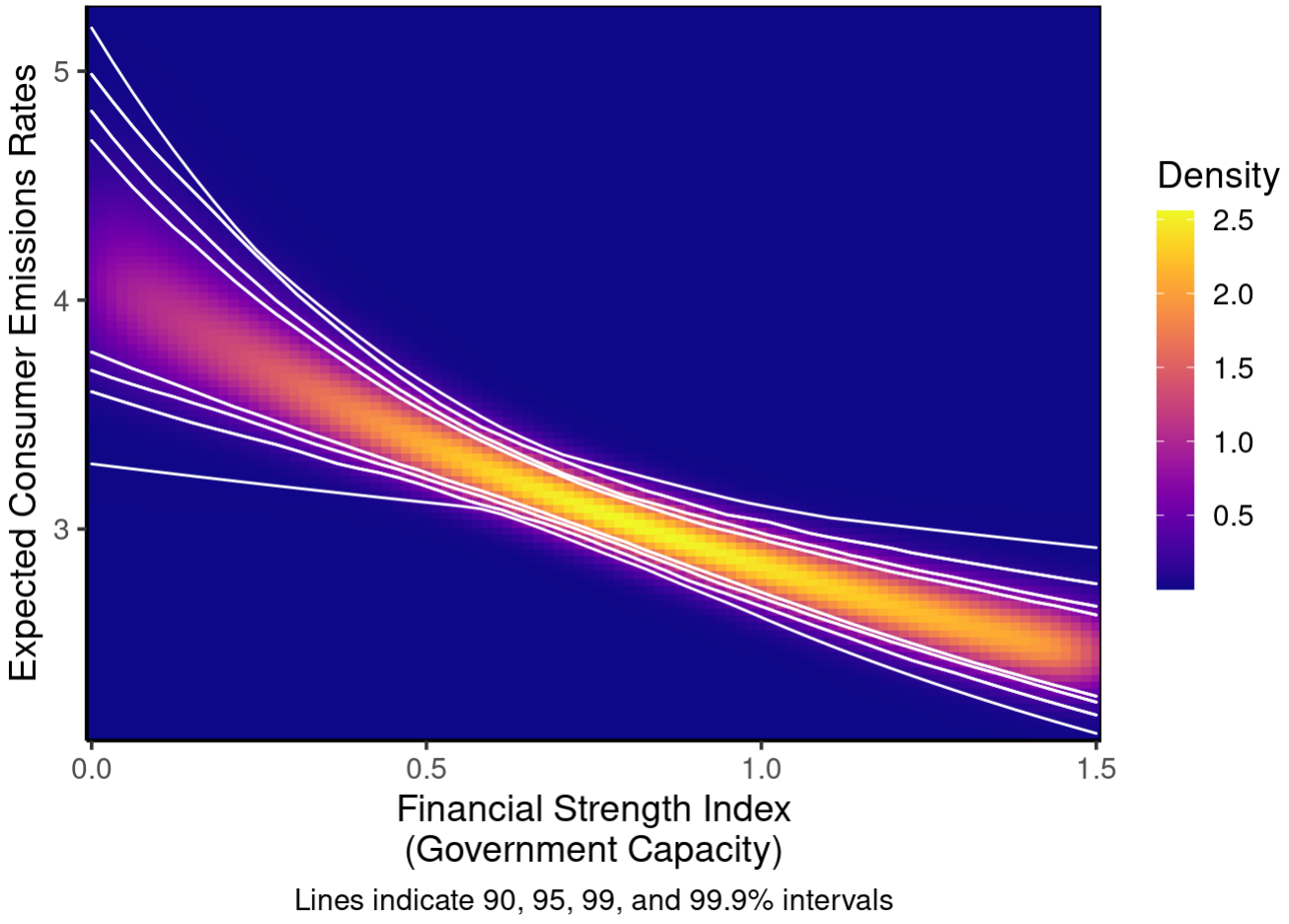
```
## `summarise()` has grouped output by 'fin_str_index'. You can override using the `.gro
ups` argument.
```

```
# Finally, visualize
out %>%
  ggplot(mapping = aes(x = fin_str_index, y = median,
                       ymin = lower, ymax = upper,
                       group = group, fill = ci)) +
  geom_ribbon(color = "white") +
  geom_line(color = "white", linetype = "dashed") +
  viridis::scale_fill_viridis(discrete = TRUE, option = "plasma",
                              direction = 1, begin = 0.2, end = 0.8) +
  labs(x = "Financial Strength Index\n(Government Capacity)",
       y = "Expected Consumer Emissions Rates\n(within 95% confidence interval)",
       fill = "Confidence\nInterval")
```



2.5 2-D Density Plane of Expected Values

```
# Using raster
mysim %>%
  ggplot(mapping = aes(x = fin_str_index, y = ev)) +
  # Map the density plane here
  stat_density_2d(aes(fill = ..density..), geom = "raster", contour = FALSE) +
  # And for kicks, overlay the confidence intervals we generated before.
  geom_ribbon(data = out,
              mapping = aes(x = fin_str_index, y = median,
                            ymin = lower, ymax = upper, group = group),
              color = "white", fill = NA) +
  viridis::scale_fill_viridis(option = "plasma") +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0)) +
  guides(fill = guide_colorbar(barwidth = 1, barheight = 10)) +
    labs(x = "Financial Strength Index\n(Government Capacity)",
       y = "Expected Consumer Emissions Rates",
       fill = "Density", caption = "Lines indicate 90, 95, 99, and 99.9% intervals")
```



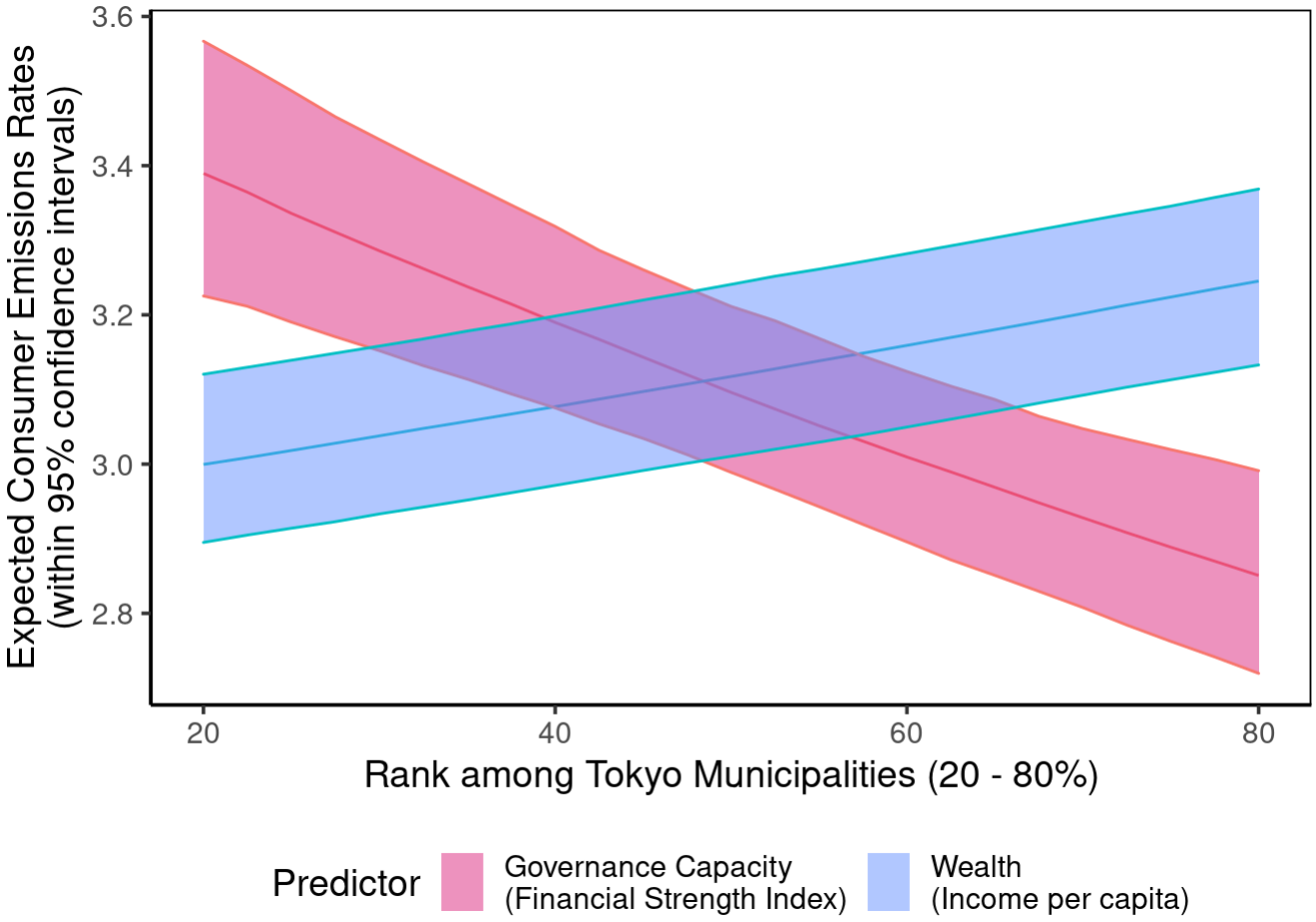
2.6 Two Ribbons or More!

Still, maybe we want to compare *two* variables. We could check, for instance, the expected values as governance capacity increases from its 20th percentile to its 80th percentile, compared against expected value as income per capita increases from its 20th percentile to its 80th percentile.

```
# Get quantile ranges below
myfin <- quantile(dat$fin_str_index, c(0.2, 0.8))
myinc <- quantile(dat$income_per_capita, c(0.2, 0.8))

mysim <- bind_rows(
  m1 %>%
    setx(fin_str_index = seq(from = myfin[1], to = myfin[2], length.out = 25)) %>%
    sim() %>%
    zelig_qi_to_df() %>%
    qi_slimmer(qi_type = "ev", ci = 0.95) %>%
  select(x = fin_str_index, median= qi_ci_median,
           lower = qi_ci_min, upper = qi_ci_max) %>%
    mutate(type = "Governance Capacity\n(Financial Strength Index)"),
  m1 %>%
    setx(income_per_capita = seq(from = myinc[1], to = myinc[2], length.out = 25)) %>%
    sim() %>%
    zelig_qi_to_df() %>%
    qi_slimmer(qi_type = "ev", ci = 0.95) %>%
    select(x = income_per_capita, median= qi_ci_median,
           lower = qi_ci_min, upper = qi_ci_max) %>%
    mutate(type = "Wealth\n(Income per capita)")
) %>%
  group_by(type) %>%
  arrange(x) %>%
  mutate(value = seq(from = 20, to = 80, length.out = 25))
```

```
## Slimming Expected Values . . .
## Slimming Expected Values . . .
```

```
mysim %>%
  ggplot(mapping = aes(x = value, y = median, ymin = lower, ymax = upper,
                       group = type, fill = type, color = type)) +
  geom_line() +
  geom_ribbon(alpha = 0.5) +
  scale_fill_manual(values = c("#DC267F", high = "#648FFF"))  +
  theme(legend.position = "bottom") +
  guides(color = FALSE) +
  labs(x = "Rank among Tokyo Municipalities (20 - 80%)",
       y = "Expected Consumer Emissions Rates\n(within 95% confidence intervals)",
       fill = "Predictor")
```

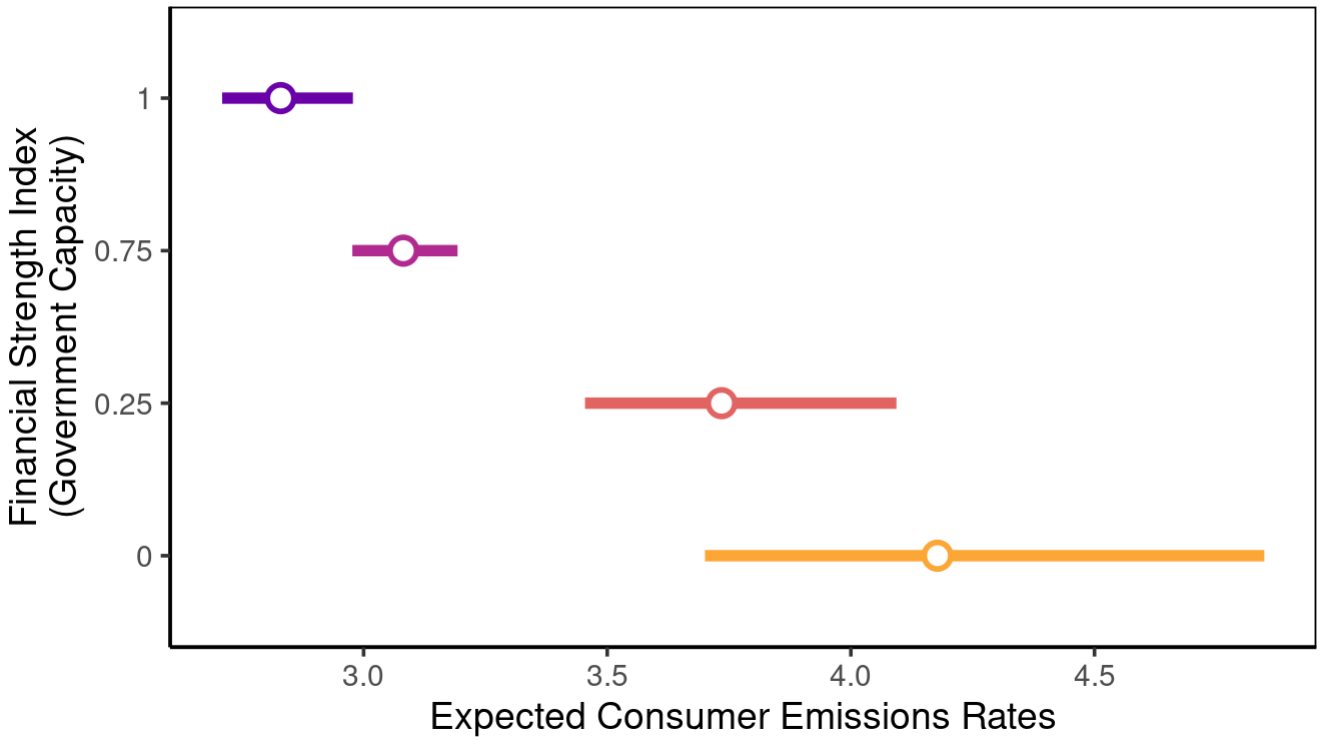

Expected Consumer Emissions Rates
(within 95% confidence intervals)
3.6
3.4
3.2
3.0
2.8
20
40
60
80
Rank among Tokyo Municipalities (20 - 80%)
Predictor
Governance Capacity
(Financial Strength Index)
Wealth
(Income per capita)

2.7 Dot and Whisker Plots

Alternatively, we might want to show expected emissions given very specific quantities, not the whole range. In that case, we can use a dot and whisker plot. This is commonly used to visualize beta coefficients, but using simulation, we can estimate the expected emissions given specific levels of financial strength

```
mysim <- m1 %>%
  setx(fin_str_index = c(0, 0.25, 0.75, 1)) %>%
  sim() %>%
  zelig_qi_to_df() %>%
  select(ev = expected_value, fin_str_index)
```

```
mysim %>%
  # For each level of financial strength specified, get the median & 95% CI
  group_by(fin_str_index) %>%
  summarize(median = median(ev),
            lower = quantile(ev, 0.025),
            upper = quantile(ev, 0.975)) %>%
  ggplot(mapping = aes(x = factor(fin_str_index), y = median,
                       ymin = lower, ymax = upper, color = factor(fin_str_index))) +
  geom_linerange(size = 2) +
  geom_point(size = 5) +
  geom_point(size = 3, color = "white") +
  coord_flip() +
  viridis::scale_color_viridis(discrete = TRUE, direction = -1,
                               begin = 0.2, end = 0.8, option = "plasma") +
  labs(x = "Financial Strength Index\n(Government Capacity)",
       y = "Expected Consumer Emissions Rates") +
  guides(color = FALSE)
```



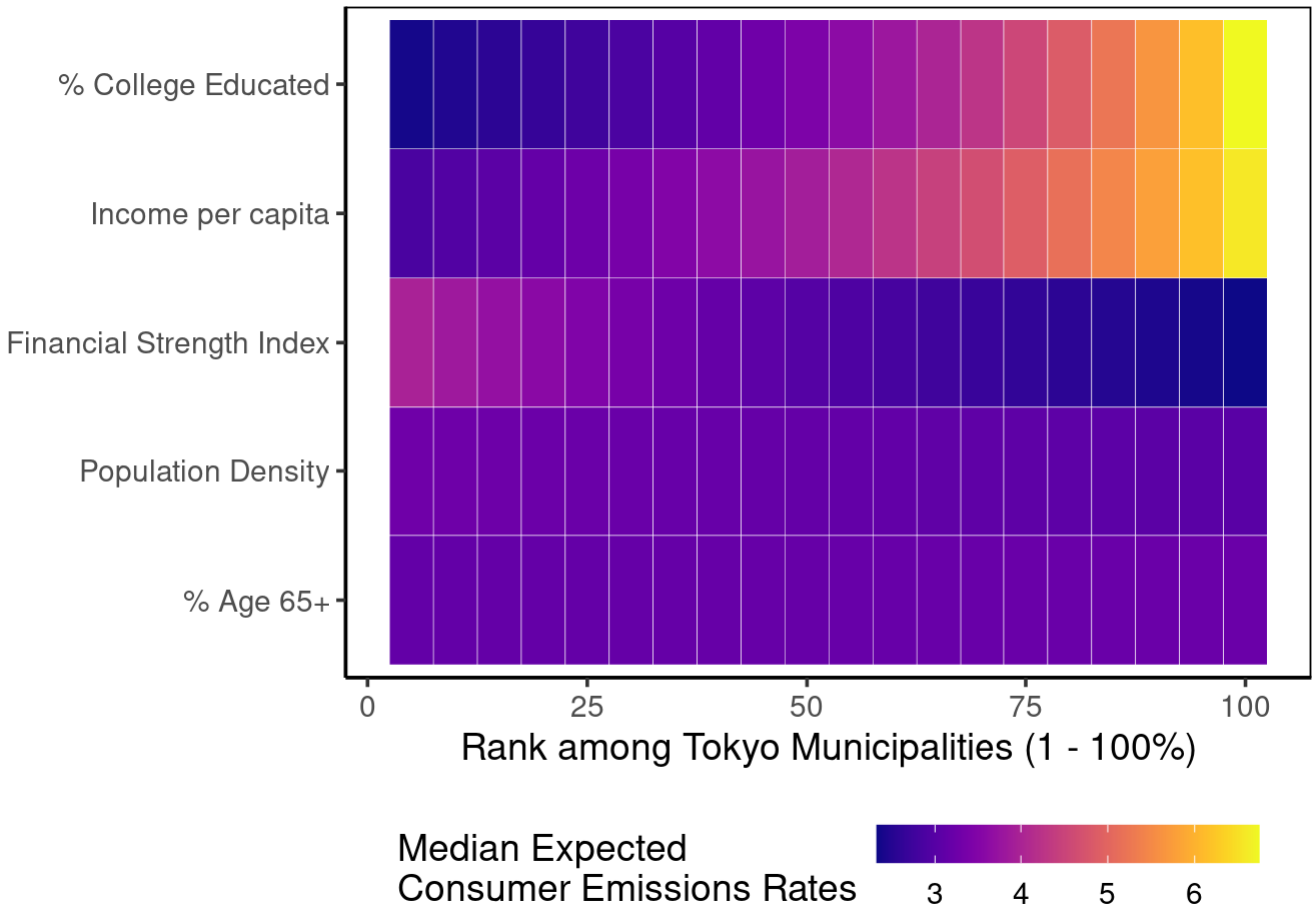
2.8 Shaded Bar Plots of Expected Values

Alternatively, we may want to show how expected values vary across the extent of several variables at once. The following plot uses the median expected value to show change over time. One downside is that it does not show confidence intervals, but if you select only variables with statistically significant effects, that might offer a way to justify that the trends you are showing are valid.

```
mysim <- list(
  m1 %>%
    setx(fin_str_index = seq(from = min(dat$fin_str_index),
                             to = max(dat$fin_str_index), length.out = 20)) %>%
    sim(),
  m1 %>%
    setx(income_per_capita = seq(from = min(dat$income_per_capita),
                                 to = max(dat$income_per_capita), length.out = 20)) %>%
    sim(),
  m1 %>%
    setx(pop_density = seq(from = min(dat$pop_density),
                           to = max(dat$pop_density), length.out = 20)) %>%
    sim(),
  m1 %>%
    setx(pop_age_65_plus = seq(from = min(dat$pop_age_65_plus),
                               to = max(dat$pop_age_65_plus), length.out = 20)) %>%
    sim(),
  m1 %>%
    setx(pop_college = seq(from = min(dat$pop_college),
                           to = max(dat$pop_college), length.out = 20)) %>%
    sim()) %>%
  map_dfr(~zelig_qi_to_df(.) %>%
            qi_slimmer(qi_type = "ev", ci = 0.95), .id = "id") %>%
  mutate(x = case_when(
    id == 1 ~ fin_str_index,
    id == 2 ~ income_per_capita,
    id == 3 ~ pop_density,
    id == 4 ~ pop_age_65_plus,
    id == 5 ~ pop_college)) %>%
  mutate(id = id %>% recode_factor(
    "1" = "Financial Strength Index",
    "2" = "Income per capita",
    "3" = "Population Density",
    "4" = "% Age 65+",
    "5" = "% College Educated")) %>%
  select(id, x, median = qi_ci_median, lower = qi_ci_min, upper = qi_ci_max) %>%
  arrange(id, x) %>%
  group_by(id) %>%
  mutate(group = 1:n() / 20 * 100) %>%
  mutate(rank = max(median)) %>%
  ungroup()
```

```
## Slimming Expected Values . . .
## Slimming Expected Values . . .
## Slimming Expected Values . . .
## Slimming Expected Values . . .
## Slimming Expected Values . . .
```

```
mysim %>%
  ggplot(mapping = aes(x = reorder(id, rank), y = group, fill = median)) +
  geom_tile(color = "white") +
  viridis::scale_fill_viridis(option = "plasma")  +
  theme(legend.position = "bottom") +
  guides(fill = guide_colorbar(barwidth = 10, barheight = 1)) +
  coord_flip() +
  labs(x = NULL,
       y = "Rank among Tokyo Municipalities (1 - 100%)",
       fill = "Median Expected\nConsumer Emissions Rates")
```



2.9 Density Plots

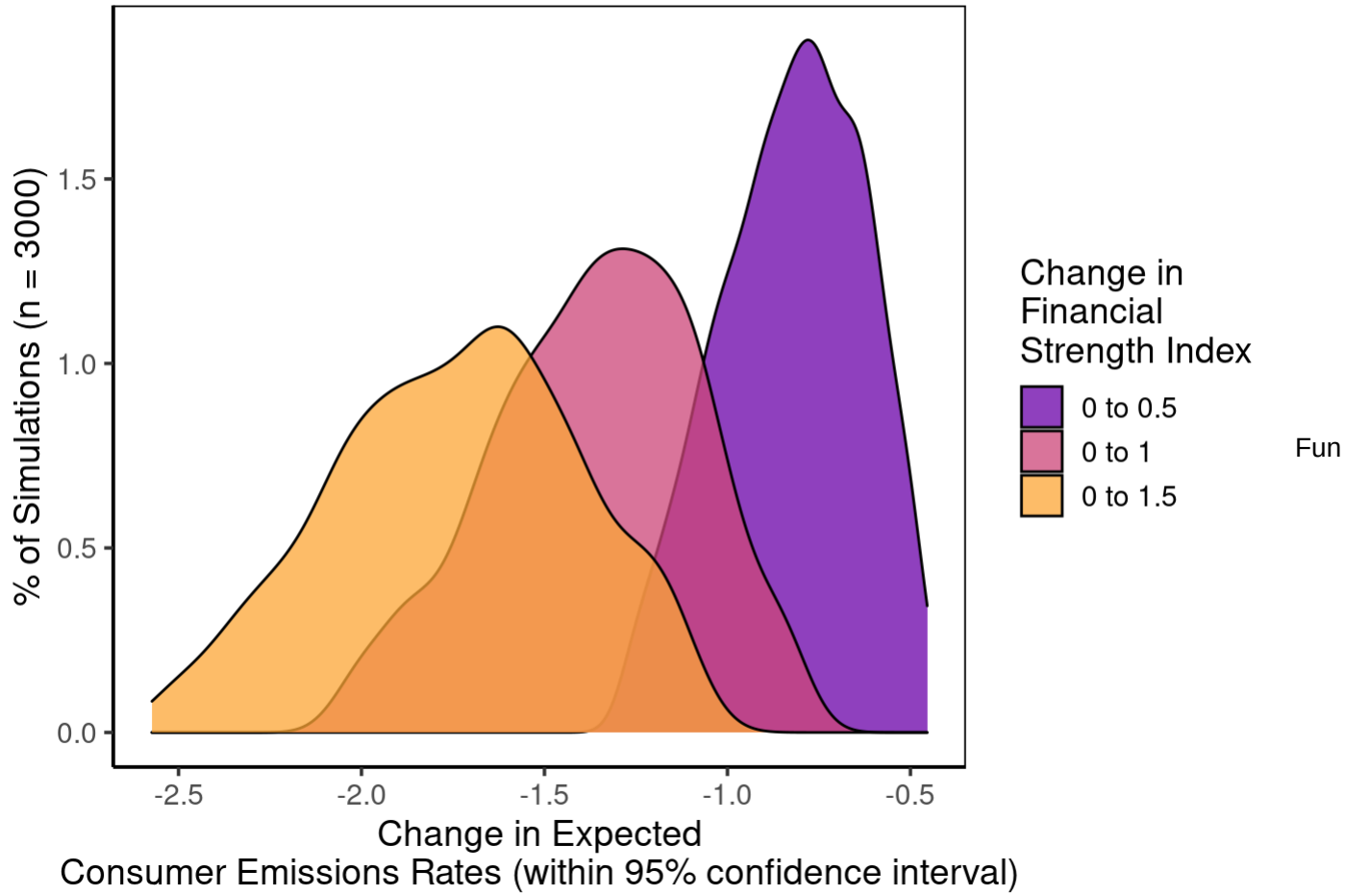
However, it might be more useful for know exactly how many more consumer emissions per thousand residents a city faces if they were to increase their government capacity. This requires a related technique called first differences, where you generate 1000 expected values based on one condition, and then subtract each from 1000 more expected values based on another condition. The result is a first difference, more easily read as the expected change in consumer emissions rates.

For example, let's estimate the change in consumer emissions given three different degrees of change in government capacity. These include a change in their financial strength index from 0 to 0.5, from 0 to 1, and from 0 to 1.5.

```
mysim <- bind_rows(
  m1 %>%
    sim(., setx(., fin_str_index = 0),
        setx1(., fin_str_index = 0.5)) %>%
    with(sim.out$x1$fd %>% unlist()) %>%
    data.frame(fd = .) %>%
    mutate(type = "0 to 0.5"),

  m1 %>%
    sim(., setx(., fin_str_index = 0),
        setx1(., fin_str_index = 1)) %>%
    with(sim.out$x1$fd %>% unlist()) %>%
    data.frame(fd = .) %>%
    mutate(type = "0 to 1"),


  m1 %>%
    sim(., setx(., fin_str_index = 0),
        setx1(., fin_str_index = 1.5)) %>%
    with(sim.out$x1$fd %>% unlist()) %>%
    data.frame(fd = .) %>%
    mutate(type = "0 to 1.5")
)
```

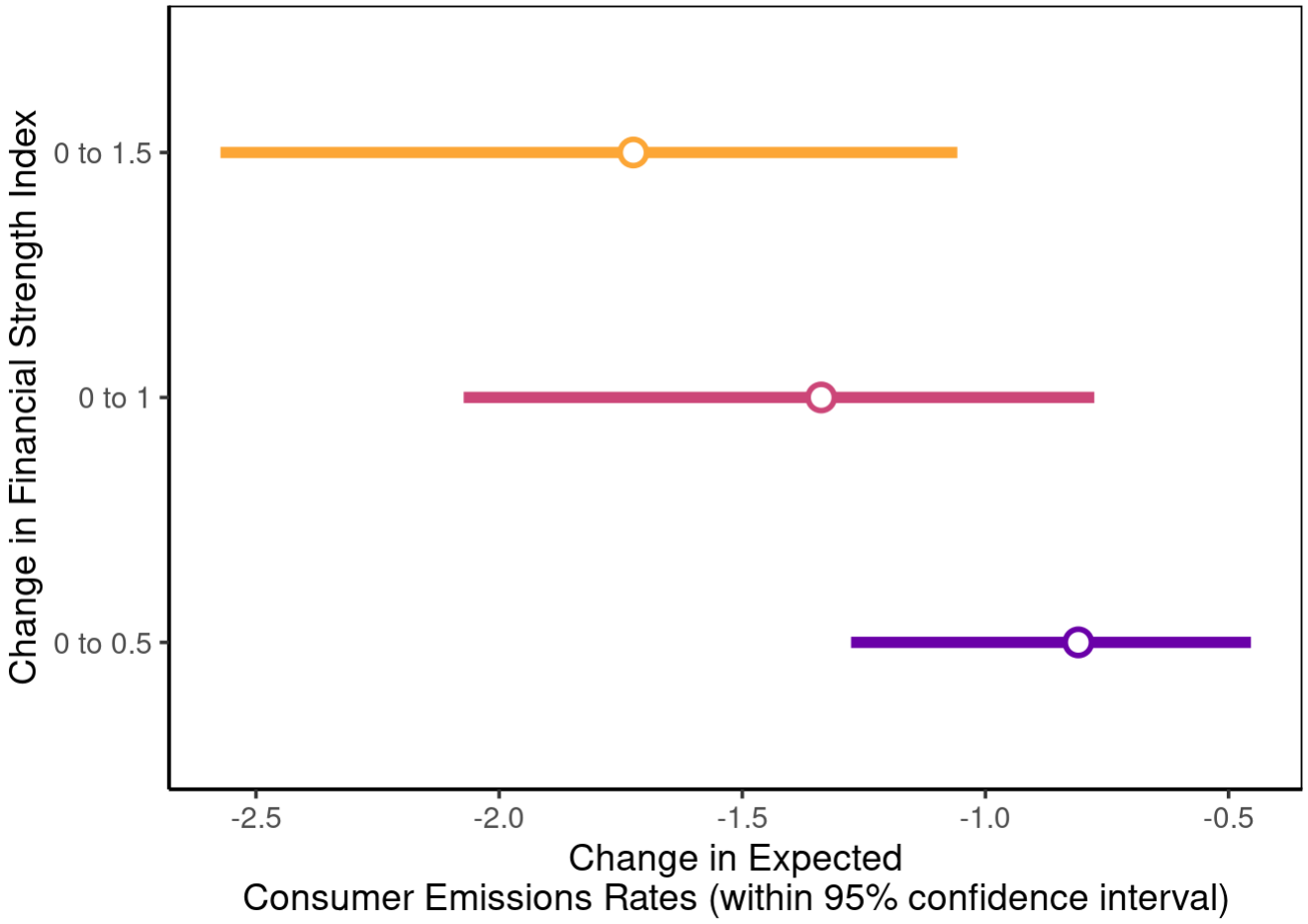
Now, we can visualize the distribution of first differences in several different ways, the first of which is as density plots. These density plots each reflect 1000 simulations, and they show nicely the spread of estimates from highest to lowest possible change in consumer emissions.

```
mysim %>%
  group_by(type) %>%
  # let's filter to just the 95% confidence interval
  filter(fd > quantile(fd, 0.025),
         fd < quantile(fd, 0.975)) %>%
  ggplot(mapping = aes(x = fd,color = type, fill = type)) +
  geom_density(alpha = 0.75, color = "black") +
  guides(color = FALSE) +
  viridis::scale_fill_viridis(discrete = TRUE, begin = 0.2, end = 0.8, option = "plasma"
) +
  labs(y = "% of Simulations (n = 3000)",
       x = "Change in Expected\nConsumer Emissions Rates (within 95% confidence interva
l)",
       fill = "Change in\nFinancial\nStrength Index")
```



fact - density plots are just dot-and-whisker plots with a distribution. So, instead, we could just as easily depict these as dot-and-whisker plots.

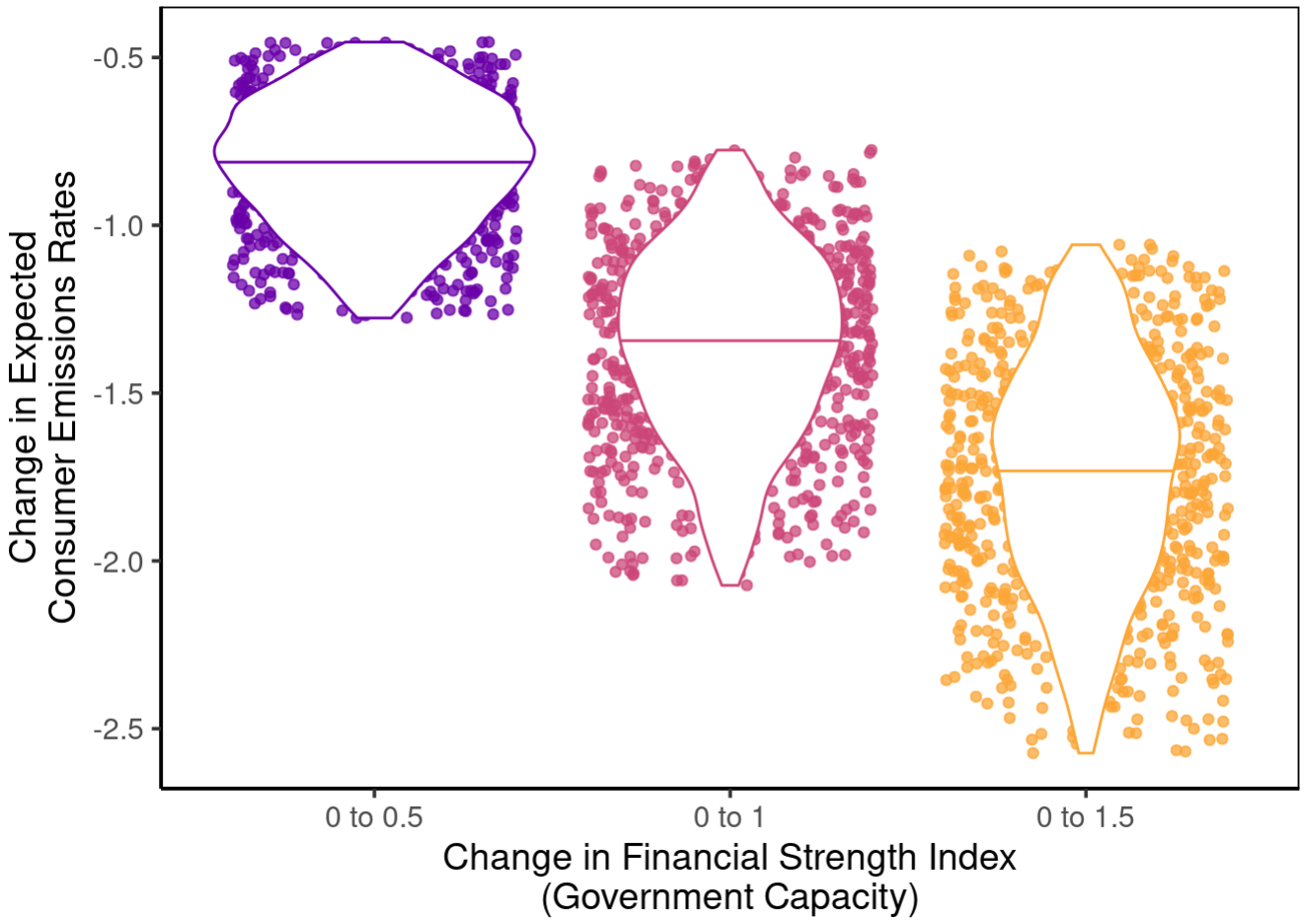
```
mysim %>%
  # let's filter to just the 95% confidence interval
  group_by(type) %>%
  summarize(median = median(fd),
            lower = quantile(fd, 0.025),
            upper = quantile(fd, 0.975)) %>%
  ggplot(mapping = aes(x = type, y = median, ymin = lower, ymax = upper,
                       color = type)) +
  geom_linerange(size = 2) +
  geom_point(size = 5) +
  geom_point(size = 3, color = "white") +
  viridis::scale_color_viridis(discrete = TRUE, begin = 0.2, end = 0.8, option = "plasm
a") +
  coord_flip() +
  guides(color = FALSE) +
  labs(x = "Change in Financial Strength Index",
       y = "Change in Expected\nConsumer Emissions Rates (within 95% confidence interva
l)")
```



2.10 Violin Plots

Alternatively, we could take the same results and depict them as violins, which are double-sided density plots that show the distribution, while not overlapping with each other. We could also display the actual expected values, jittered slightly for effect. Like always, it is probably important to first zoom into just a specific confidence interval worth of expected values before plotting distributions; this visual uses a 95% confidence interval. One helpful thing about violin plots is that you can usually draw a line at the median - the 50th percentile - helping crystallize exactly what value it is.

```
mysim %>%
  group_by(type) %>%
  # let's filter to just the 95% confidence interval
  filter(fd > quantile(fd, 0.025),
         fd < quantile(fd, 0.975)) %>%
  ggplot(mapping = aes(x = type, y = fd, color = type)) +
  geom_jitter(alpha = 0.75) +
  geom_violin(draw_quantiles = 0.5)  +
  labs(x = "Change in Financial Strength Index\n(Government Capacity)",
       y = "Change in Expected\nConsumer Emissions Rates") +
  guides(color = FALSE) +
  viridis::scale_color_viridis(discrete = TRUE, begin = 0.2, end = 0.8, option = "plasm
a")
```



2.11 Lollipop Charts

Alternatively, you might want something more recognizable. This is a lollipop chart, which shows how far away from zero the median first difference is, using a line connected to a lollipop, on which is written the median first different. However, we still might want some estimate of the confidence interval, so we show the 95% confidence range by jittering the first differences themselves around the lollipops.

```
avgsim <- mysim %>%
  group_by(type) %>%
  summarize(median = median(fd))


mysim %>%
  group_by(type) %>%
  # let's filter to just the 95% confidence interval
  filter(fd > quantile(fd, 0.025),
         fd < quantile(fd, 0.975)) %>%
  ggplot(mapping = aes(x = type, y = fd, color = type)) +
  geom_jitter(alpha = 0.2)  +
  geom_linerange(data = avgsim,
                 mapping = aes(x = type, y = median, ymax = median, ymin = 0),
                 color = "darkgrey", size = 3) +
  geom_point(data = avgsim,
             mapping = aes(x = type, y = median,color = type),
             size = 35, alpha = 0.75) +
  geom_point(data = avgsim,
             mapping = aes(x = type, y = median,color = type),
             size = 30, color = "white", alpha = 0.75) +
  geom_point(data = avgsim,
             mapping = aes(x = type, y = median,color = type),
             size = 25) +
  geom_text(data = avgsim,
            mapping = aes(x = type, y = median,
                          label = round(median, 2), color = type),
            size = 5, color = "white") +
  guides(color = FALSE) +
  viridis::scale_color_viridis(discrete = TRUE, begin = 0.8, end = 0.2, option = "plasm
a") +
  labs(x = "Change in Financial Strength Index\n(Government Capacity)",
       y = "Change in Expected\nConsumer Emissions Rates")
```

0
-1
-2
Change in Expected
Consumer Emissions Rates
-0.81
-1.34
-1.72
0 to 0.5
0 to 1
0 to 1.5
Change in Financial Strength Index
(Government Capacity)

2.12 Broom / Mustache Plot

There are still several other interesting ways you could present this data. For example, you could take the raw first differences and arrange them in a specific order, to show the full range of pre- and post-treatment values that you would see within a 95% confidence interval.

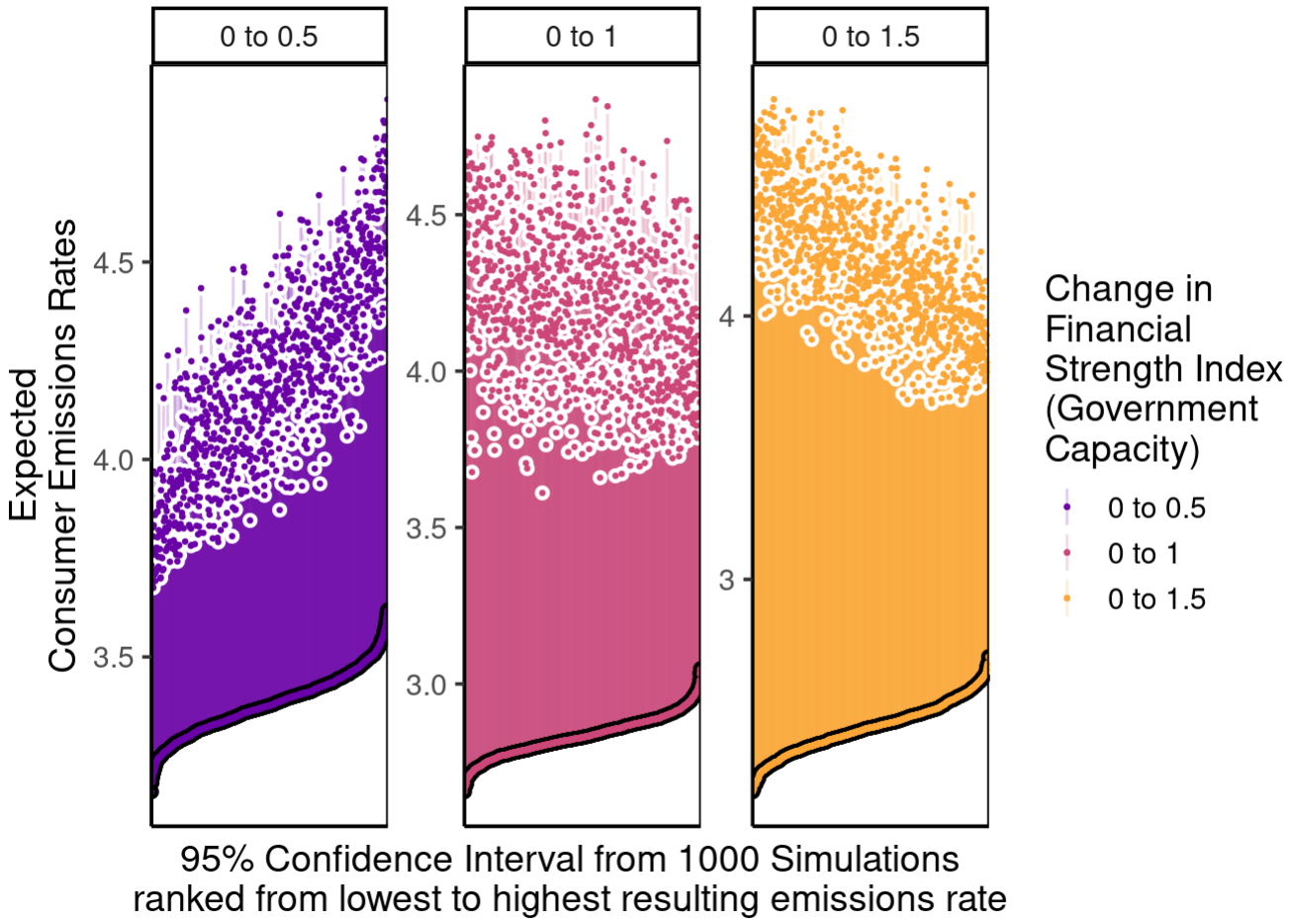
This one is a little tricky, because you need both the first differences and the original two sets of expected values to do it.

```
# Let's write a quick function to extract first differences,
# plus original expected values
get_fd = function(mysimulation){

  data.frame(fd = mysimulation$sim.out$x1$fd %>% unlist(),
             x1 = mysimulation$sim.out$x1$ev %>% unlist(),
             x = mysimulation$sim.out$x$ev %>% unlist()) %>%
    return()
}

# Now let's get those values
mysim <- bind_rows(
  m1 %>%
    sim(., setx(., fin_str_index = 0),
        setx1(., fin_str_index = 0.5)) %>%
    get_fd() %>%
      mutate(type = "0 to 0.5"),

  m1 %>%
    sim(., setx(., fin_str_index = 0),
        setx1(., fin_str_index = 1)) %>%
    get_fd() %>%
    mutate(type = "0 to 1"),


  m1 %>%
    sim(., setx(., fin_str_index = 0),
        setx1(., fin_str_index = 1.5)) %>%
    get_fd() %>%
    mutate(type = "0 to 1.5")
) %>%
    group_by(type) %>%
  filter(fd > quantile(fd, 0.025),
         fd < quantile(fd, 0.975)) %>%
  # And conveniently arrange them
  arrange(type, x1, x) %>%
  group_by(type) %>%
  mutate(order = 1:n()) %>%
  ungroup()
```

As shown below, the resulting plot looks a little like a broom or a mustache. It shows the 95% most common first differences from 1000 simulations in each panel, arranged to show the starting values at the top (with white outlines), linked to their ending values at the bottom (with black outlines). This highlights both the degree and

range of possible emissions rates given a change in governance capacity. It may be more visually interesting than useful, but it certainly is another valid technique.

```
mysim %>%
  ggplot(mapping = aes(x = reorder(order, x1), y =x1, ymin = x, ymax = x1, color = typ
e)) +
  geom_linerange(alpha = 0.2) +
  # Add points for the bottom
  geom_point(size = 2, color = "black") +
  geom_point(size = 0.5) +
  # Add points for the top
  geom_point(mapping = aes(y = x), size = 2, color = "white") +
  geom_point(mapping = aes(y = x), size = 0.5) +
  # Split into panels
  facet_wrap(~type, scales = "free") +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank()) +
  viridis::scale_color_viridis(option = "plasma", begin = 0.2, end = 0.8, discrete = TRU
E) +
    labs(color = "Change in\nFinancial\nStrength Index\n(Government\nCapacity)",
       y = "Expected\nConsumer Emissions Rates",
       x = "95% Confidence Interval from 1000 Simulations\nranked from lowest to highest
resulting emissions rate")
```



2.13 Interactions

Finally, we might want to estimate the effect of governance capacity in municipalities with progressively greater levels of commercial activity - something which might plausibly affect consumer spending. We can do that by modeling an interaction effect. Fortunately, there are several visualizations we can produce using simulation for this as well.

```
m2 <- dat %>%
  zelig(formula = consumer_emissions ~ fin_str_index  * value_commerce_mill +
          pop_density + income_per_capita +
          pop_age_65_plus + pop_college +
          value_agr_mill + value_manuf_mill + value_commerce_mill + year,
        model = "gamma", cite = FALSE)
```

Here, first differences are especially helpful. Let’s find out how much greater consumer emissions we would expect given different levels of governance capacity and commercial activity, relative to an average city with the lowest governance capacity and comercial activity in the sample.

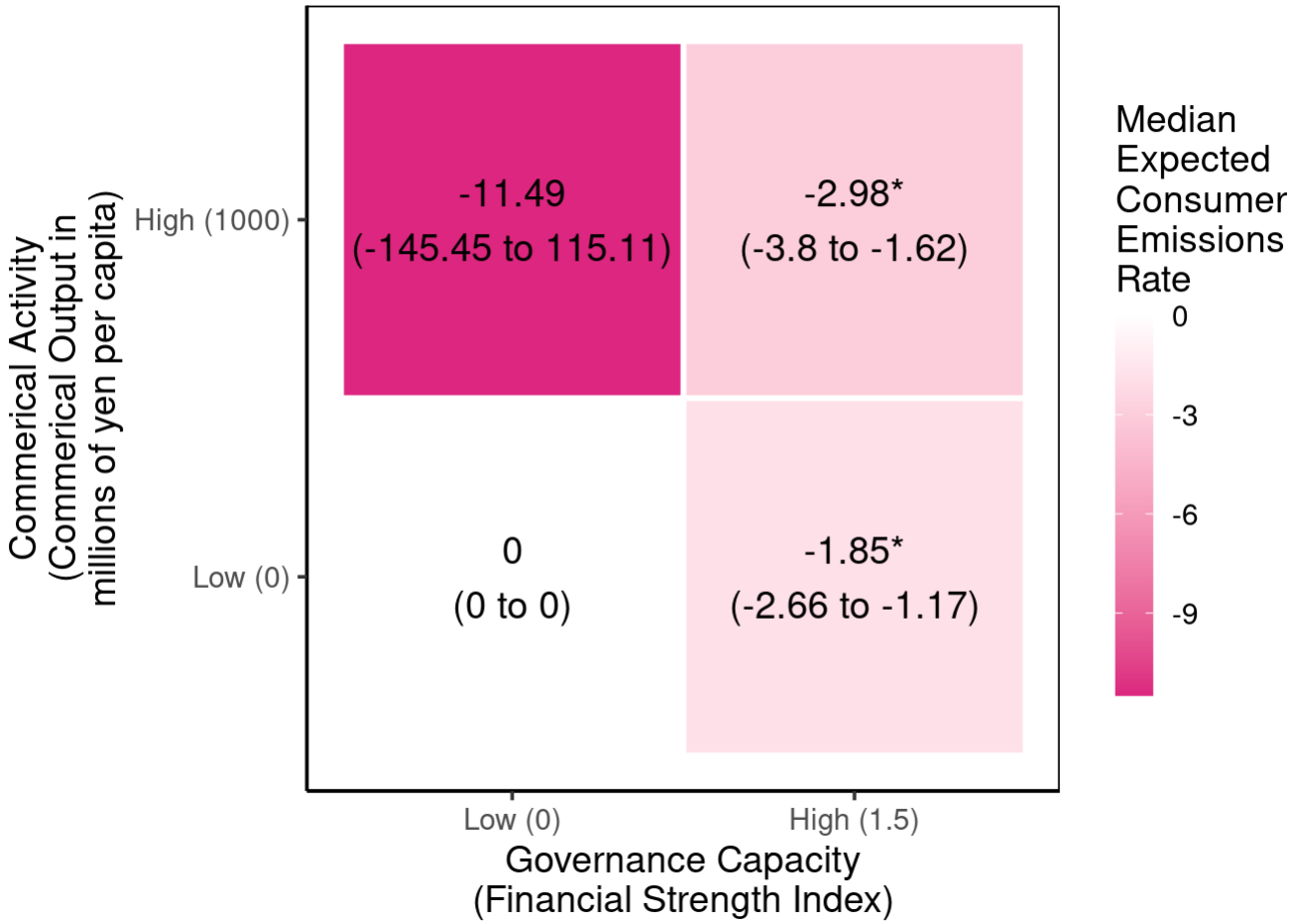
```
# We'll use our function from before for extracting quantities of interest
get_fd = function(mysimulation){

  data.frame(fd = mysimulation$sim.out$x1$fd %>% unlist(),
             x1 = mysimulation$sim.out$x1$ev %>% unlist(),
             x = mysimulation$sim.out$x$ev %>% unlist()) %>%
    return()
}

# Now let's get th
mysim <- list(
  m2 %>%
    sim(., setx(., fin_str_index = 0,
                value_commerce_mill = 0),
        setx1(., fin_str_index = 1.5,
              value_commerce_mill = 1000)),
  m2 %>%
    sim(., setx(., fin_str_index = 0,
                value_commerce_mill = 0),
        setx1(., fin_str_index = 1.5,
              value_commerce_mill = 0)),
  m2 %>%
    sim(., setx(., fin_str_index = 0,
                value_commerce_mill = 0),
        setx1(., fin_str_index = 0,
              value_commerce_mill = 1000)),
  m2 %>%
    sim(., setx(., fin_str_index = 0,
                value_commerce_mill = 0),
        setx1(., fin_str_index = 0,
              value_commerce_mill = 0))
) %>%
  map_dfr(~get_fd(.), .id = "id") %>%
  mutate(fin_str_index = case_when(
    id %in% c(1,2) ~ "High (1.5)",
    id %in% c(3,4) ~ "Low (0)"),
    value_commerce_mill = case_when(
      id %in% c(1,3) ~ "High (1000)",
      id %in% c(2,4) ~ "Low (0)")) %>%
  mutate(fin_str_index = factor(fin_str_index, levels = c("Low (0)", "High (1.5)")),
         value_commerce_mill = factor(value_commerce_mill, levels = c("Low (0)", "High
 (1000)")))
```

```
## Warning in rgamma(nrow(ev), shape = simparam$simalpha, scale = theta): NAs
## produced
```

This plot highlights the expected change in consumer emissions for a city with the specified traits, compared to a city with both low governance capacity and low consumer spending. The lower left quandrant is naturally 0, since it has low capacity and spending, so no change. We see that the strongest reduction in consumer emissions comes from areas with more consumer spending, the upper left. However, both cities with greater capacity and spending also see reductions in emissions. It does suggest that the most shopping-heavy parts of Tokyo emit many fewer consumer emissions, which is interesting.

```
mysim %>%
  group_by(id, fin_str_index, value_commerce_mill) %>%
  summarize(median = median(fd),
            se = sd(fd),
            lower = quantile(fd, 0.025),
            upper = quantile(fd, 0.975)) %>%
  # check whether the lower or upper intervals cross 0
  mutate(sig = case_when(
    lower > 0 & upper > 0 ~ "*",
    lower < 0 & upper < 0 ~ "*",
    lower < 0 & upper > 0 ~ "",
    TRUE ~ "")) %>%
  mutate(estimate = paste(round(median, 2), sig,
                          "\n(", round(lower, 2), " to ",
                          round(upper, 2), ")", sep = "")) %>%
  ggplot(mapping = aes(x = fin_str_index, y = value_commerce_mill,
                       fill = median, label = estimate)) +
  geom_tile(color = "white", size = 1) +
  geom_text(size = 5, color = "black") +
  scale_fill_gradient2(low = "#DC267F", high = "#648FFF", mid = "white", midpoint = 0) +
  guides(fill = guide_colorbar(barheight = 10, barwidth = 1)) +
  labs(fill = "Median\nExpected\nConsumer\nEmissions\nRate",
       x = "Governance Capacity\n(Financial Strength Index)",
       y = "Commerical Activity\n(Commerical Output in\nmillions of yen per capita)")
```

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


-11.49
(-145.45 to 115.11)
-2.98*
(-3.8 to -1.62)
0
(0 to 0)
-1.85*
(-2.66 to -1.17)
High (1000)
Low (0)
Commerical Activity
(Commerical Output in
millions of yen per capita)
Low (0)
High (1.5)
Governance Capacity
(Financial Strength Index)
Median
Expected
Consumer
Emissions
Rate
0
-3
-6
-9

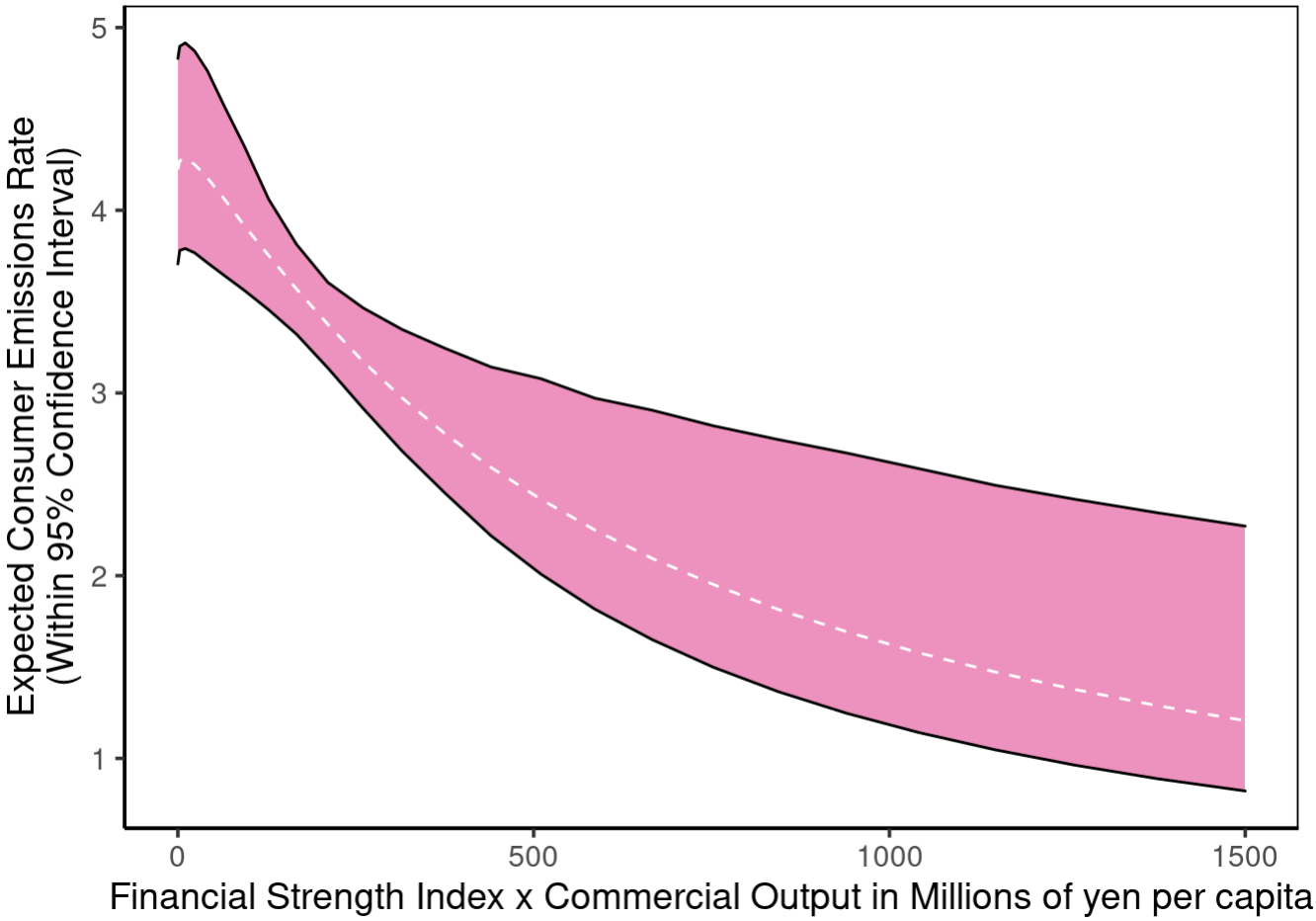
2.14 Interactions with Ribbon Plots

Alternatively, we can visualize the product of these two predictors using expected values and ribbons. This describes the interaction effect pretty well.

```
mysim <- m2 %>%
    setx(fin_str_index = seq(from = 0, to = 1.5, length.out = 25),
         value_commerce_mill = seq(from = 0, to = 1000, length.out = 25)) %>%
    sim() %>%
    zelig_qi_to_df(.) %>%
            qi_slimmer(qi_type = "ev", ci = 0.95) %>%
  select(fin_str_index, value_commerce_mill,
         median = qi_ci_median, lower = qi_ci_min, upper = qi_ci_max)
```

```
## Slimming Expected Values . . .
```

```
mysim %>%
  ggplot(mapping = aes(x = fin_str_index * value_commerce_mill,
                       y = median, ymin = lower, ymax = upper)) +
  geom_ribbon(alpha = 0.5, color = "black", fill = "#DC267F") +
  geom_line(color = "white", linetype = "dashed") +
    labs(x = "Financial Strength Index x Commercial Output in Millions of yen per capit
a",
       y = "Expected Consumer Emissions Rate\n(Within 95% Confidence Interval)")
```


Expected Consumer Emissions Rate
(Within 95% Confidence Interval)
5
4
3
2
1
0
500
1000
1500
Financial Strength Index x Commercial Output in Millions of yen per capita

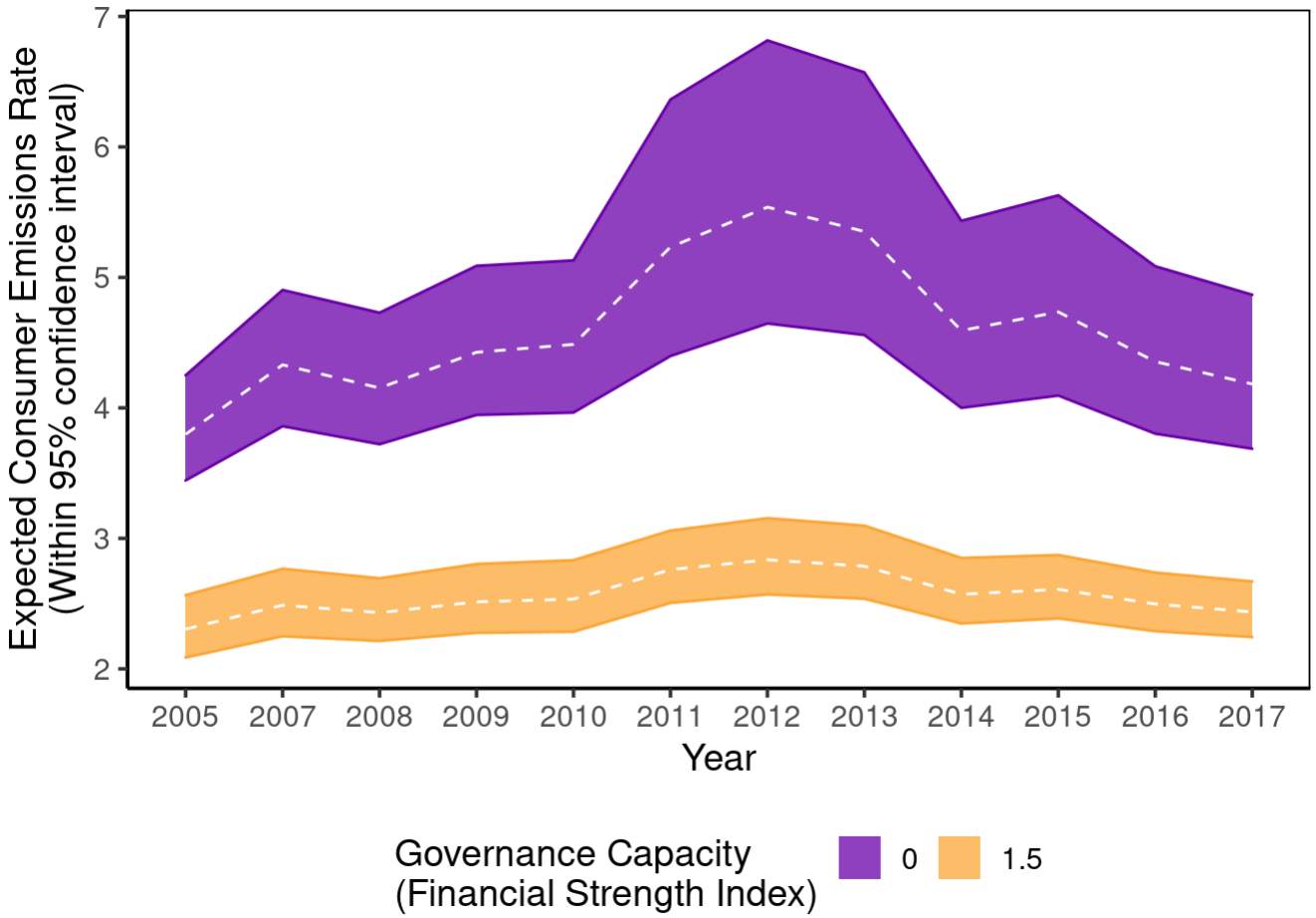
2.15 Visualizing Effects in Context

Finally, sometimes you may want to tell a story about a specific set of cases in your data. For example, we can simulate the expected consumer emissions rates for cities with greater government capacity in each specific year, taking into account the different fixed impacts of different years. We are able to do this because we added fixed effects.

```
mysim <- m1 %>%
  setx(fin_str_index = c(0, 1.5),
       year = unique(dat$year) %>% sort()) %>%
  sim() %>%
  zelig_qi_to_df() %>%
  select(fin_str_index, year, ev = expected_value) %>%
  group_by(fin_str_index, year) %>%
  summarize(median = median(ev),
            lower = quantile(ev, 0.025),
            upper = quantile(ev, 0.975))
```

```
## `summarise()` has grouped output by 'fin_str_index'. You can override using the `.gro
ups` argument.
```

```
mysim %>%
  ggplot(mapping = aes(x = year, y = median, ymin = lower, ymax = upper,
                       color = factor(fin_str_index), group = fin_str_index,
                       fill = factor(fin_str_index))) +
  geom_ribbon(alpha = 0.75) +
  geom_line(color = "white", linetype = "dashed") +
  labs(x = "Year",
       y = "Expected Consumer Emissions Rate\n(Within 95% confidence interval)",
       fill = "Governance Capacity\n(Financial Strength Index)") +
  theme(legend.position = "bottom") +
  guides(color = FALSE) +
  viridis::scale_fill_viridis(option = "plasma", begin = 0.2, end = 0.8, discrete = TRUE
) +
  viridis::scale_color_viridis(option = "plasma", begin = 0.2, end = 0.8, discrete = TRU
E)
```


Expected Consumer Emissions Rate
(Within 95% confidence interval)
7
6
5
4
3
2
2005
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
Year
Governance Capacity
(Financial Strength Index)
0
1.5

3. Conclusion

Good news: there's lots of ways to visualize your results. My personal trick is to first sketch by hand a few drafts of the chart I want to make, in order to figure out what to do to make it. The big question though is where to go from here? What resources are out there to help you make your own visualizations with statistical simulation, among other techniques?

- For more ideas for graphing in R, check out the **RGraphGallery** (https://www.r-graph-gallery.com/ggplot2-package.html), a tremendous resource of visualization ideas.
- For more information about statistical simulation, check out the original paper on statistical simulation: Gary King, Michael Tomz, and Jason Wittenberg. (2000). Making the most of statistical analyses: Improving interpretation and presentation. American Journal of Political Science 44, 341-365. https://gking.harvard.edu/files/abs/making-abs.shtml (https://gking.harvard.edu/files/abs/making-abs.shtml)
- If you liked this tutorial, take a look at **my other tutorials** (https://www.timothyfraser.com/tutorials-in-r/) as well! Also, please do check out the **Zelig team's vignettes here** (https://zeligproject.org/).
- For more tutorials of visualizing regression results, check out my other tutorials on *general techniques for visualizing regression results.* (https://www.timothyfraser.com/tutorials-in-r/visualizing-statistical-model-results) and *how statistical simulation works* (https://www.timothyfraser.com/tutorials-in-r/statistical-simulation-with-zelig).

Have more questions? Reach out to me at timothy.fraser.1@gmail.com (mailto:timothy.fraser.1@gmail.com) or on ResearchGate!