SPEC REU R Resources: Visualizing Regression Results with dot-and-whisker Plots – Groupwork

Yeiyoung Choo and Claudia Salas Gimenez

Summer 2024

In this groupwork, we will practice how to generate dot-and-whisker plots with results stored in regression objects. This assignment aims to familiarize you with the process of extracting and manipulating regression outputs, and to enhance your skills in creating visualizations using `ggplot2` and the `dotwhisker` package.

Initial Setup

Before starting the exercises, ensure your working directory is set to where your data files are located and that you've loaded the required packages and dataset. This assignment uses the `aslaksen2010.rds` data from Aslaksen's 2010 study on the relationship between oil income and democracy.

Please note that for part of this module, we will use the `dotwhisker` package. As of April 2024, `dotwhisker` has been removed from CRAN, so you need to install its dependencies, `prediction` and `margins`, from GitHub to load it.

```
# Set working directory
#setwd("YourFolderPath")

# Load required libraries
library(tidyverse)
library(ggplot2)
library(broom)

# As of April 2024, "dotwhisker" is no longer on CRAN and must be installed
# with its dependencies "prediction" and "margins" from GitHub:
# remotes::install_github("leeper/prediction", force = TRUE)
# remotes::install_github("leeper/margins", force = TRUE)
# remotes::install_github("fsolt/dotwhisker", force = TRUE)

library(dotwhisker)

# Load the dataset
as2010 <- readRDS("aslaksen2010.rds")
```

Exercise 1: Understand the Data and Variables

First, let's explore the dataset to understand what the variables stand for:

Exercise 1.1: Oil-Related Variables

Identify and count the variables related to oil. How many are there, and what do they represent? We will revisit these variables for further analysis.

```r
# Identify and list variables containing 'oil'
oil_variables <- select(as2010, contains("oil"))

# List oil-related variables
names(oil_variables)
```

```
## [1] "ccode"         "oil_wb"        "oilproduction" "oilprice"
```

```r
# The variables related to oil are oil_wb, oilproduction, oilprice
```

Exercise 1.2: Democracy Index

Determine which index is used to measure the level of democracy in the dataset.

```r
# The dataset uses the Political Rights Index to measure democracy
```

Exercise 1.3: Logarithmic Transformations

Transform the population and real GDP per capita data into their logarithmic scales to simplify the analysis and manage scale disparities. Name these new variables `lpop` for the logged population and `lrgdppc` for the logged real GDP per capita.

Helpful Hint: Use `mutate()` to create the variables.

```r
# Transform population and real GDP per capita into their logarithmic scales
as2010 <- as2010 %>%
  mutate(lpop = log(population),
         lrgdppc = log(rgdppc))
```

Exercise 2: Run the Regression Models

Now let's start running the regression models. In our analysis, the outcome variable is `pr_lead`, which captures the predicted value of political rights in the subsequent year, using the Political Rights Index to measure democracy. As you may have guessed, our key predictor variable will capture the country's income from oil extraction.

Exercise 2.1: Create `oilshare` Variable

Construct the `oilshare` variable, which measures oil income as a percentage of the country's GDP. Start by multiplying the volume of oil extraction (`oil_wb`) by the price of oil (`oilprice`). Then, divide this product by the GDP and multiply the result by 100 to express `oilshare` as a percentage.

Helpful Hint: Use `mutate()` to create the variable.

```r
# Create 'oilshare' variable
as2010 <- as2010 %>%
  mutate(oilshare = 100*(oil_wb*oilprice/gdp))
```

Exercise 2.2: Replace NA values

Substitute NA values with 0 for countries where `oil_wb` is 0.

```r
# Replace NAs with 0 where oil_wb == 0
as2010$oilshare[as2010$oil_wb == 0] <- 0
```

Exercise 2.3: Simple Regression Model

Now that we have prepared the dataset, let's first run a simple linear regression model to analyze the impact of `oilshare` alone on political rights in the upcoming year. Use `oilshare` as predictor, and `pr_lead` as our outcome variable, and store the regression model as an object called `n1`. This will be our Model 1. Then use the `summary()` function to check your regression results.

Helpful Hint: Remember the basic syntax for linear regression: `lm(y ~ x,data = data)`.

```
# Run simple regression model
n1 <- lm(pr_lead ~ pr + oilshare, data = as2010)

# Check results
summary(n1)
```

```
##
## Call:
## lm(formula = pr_lead ~ pr + oilshare, data = as2010)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.98518 -0.01966 -0.00455  0.01476  0.81061
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0265457  0.0027657   9.598  < 2e-16 ***
## pr           0.9586945  0.0042674 224.657  < 2e-16 ***
## oilshare    -0.0003052  0.0001051  -2.903  0.00371 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1006 on 4394 degrees of freedom
##   (1238 observations deleted due to missingness)
## Multiple R-squared:  0.9243, Adjusted R-squared:  0.9243
## F-statistic: 2.684e+04 on 2 and 4394 DF,  p-value: < 2.2e-16
```

Exercise 2.4: Multiple Regression Model

Next, expand Model 1 by running a multiple linear regression that also includes `pr`, logged population (`lpop`), and GDP per capita (`lrgdppc`). This model assesses their collective impact on democracy alongside oil income. Save this regression model as `n2`, which is our Model 2. Review the regression results after running the model.

```
# Run multiple regression model
n2 <- lm(pr_lead ~ pr + oilshare + lpop + lrgdppc , data = as2010)

# Check results
summary(n2)
```

```
##
## Call:
## lm(formula = pr_lead ~ pr + oilshare + lpop + lrgdppc, data = as2010)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.98118 -0.02211 -0.00128  0.01230  0.80286
##
```

```
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0847748  0.0206169  -4.112 4.00e-05 ***
## pr           0.9318744  0.0056244 165.684  < 2e-16 ***
## oilshare    -0.0006740  0.0001210  -5.568 2.73e-08 ***
## lpop         0.0012616  0.0009902   1.274    0.203
## lrgdppc      0.0129660  0.0017616   7.360 2.19e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09944 on 4238 degrees of freedom
##   (1392 observations deleted due to missingness)
## Multiple R-squared:  0.9262, Adjusted R-squared:  0.9261
## F-statistic: 1.329e+04 on 4 and 4238 DF,  p-value: < 2.2e-16
```

Exercise 2.5: Expand Multiple Regression Model

Lastly, expand Model 2 by adding education (`educ`) and openness (`open`) to the list of predictors. Run the linear regression and store it as an object called `n3`. This is our Model 3. Check your regression results.

```r
# Run multiple regression model
n3 <- lm(pr_lead ~ pr + oilshare + lpop + lrgdppc + educ + open, data= as2010)

# Check results
summary(n3)
```

```
##
## Call:
## lm(formula = pr_lead ~ pr + oilshare + lpop + lrgdppc + educ +
##     open, data = as2010)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.97980 -0.02469 -0.00333  0.01592  0.78150
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.290e-03  3.348e-02   0.098 0.921727
## pr           9.076e-01  7.223e-03 125.648  < 2e-16 ***
## oilshare    -5.974e-04  1.684e-04  -3.547 0.000395 ***
## lpop        -1.754e-03  1.429e-03  -1.228 0.219599
## lrgdppc      8.368e-03  3.143e-03   2.662 0.007804 **
## educ         4.818e-03  1.130e-03   4.263 2.08e-05 ***
## open        -1.119e-04  4.727e-05  -2.366 0.018030 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1024 on 3143 degrees of freedom
##   (2485 observations deleted due to missingness)
## Multiple R-squared:  0.9188, Adjusted R-squared:  0.9186
## F-statistic:  5928 on 6 and 3143 DF,  p-value: < 2.2e-16
```

Exercise 3: Generate a Dot-and-Whisker Plot

Exercise 3.1: Dot-and-Whisker Plot Using `ggplot2`

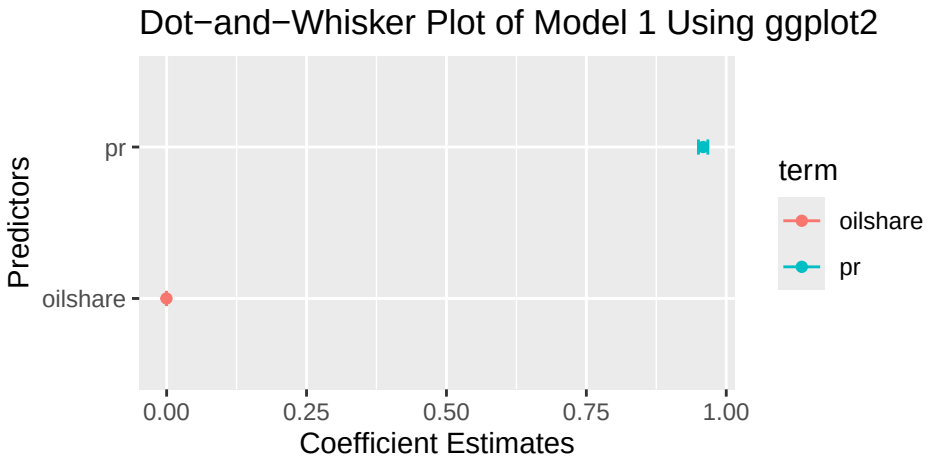
Let's dive into creating dot-and-whisker plots to better understand the regression results. We'll start by manually creating a dot-and-whisker plot for Model 1 using `ggplot2`.

Helpful Hint: Remember to tidy the model using the `tidy()` function from the `broom` package before plotting the graph. Let's call them `n1df`. Then, calculate the 95% confidence interval for `n1df`.

```
# Display regression results and save them as `summary_n1`
summary_n1 <- summary(n1)

# Create a dataframe for plotting based on the results stored in `summary_m1`
results_df <- data.frame(
  term = rownames(summary_n1$coefficients)[-1],
  estimate = summary_n1$coefficients[-1, "Estimate"],
  std.error = summary_n1$coefficients[-1, "Std. Error"],
  conf.low = summary_n1$coefficients[-1, "Estimate"] -
    1.96 * summary_n1$coefficients[-1, "Std. Error"],
  ## Calculate the lower bound of the 95% confidence interval for each coefficient.
  conf.high = summary_n1$coefficients[-1, "Estimate"] +
    1.96 * summary_n1$coefficients[-1, "Std. Error"])
  ## Calculate the upper bound of the 95% confidence interval for each coefficient.

# Creating the dot-and-whisker plot using ggplot2
ggplot(results_df, aes(x = estimate, y = term, color = term)) +
  geom_point() +
  geom_errorbar(aes(xmin = conf.low, xmax = conf.high), width = 0.1) +
  ## Sets the horizontal span of the error bars based on the confidence intervals calculated,
  ## showing the estimate's uncertainty.
  labs(title = "Dot-and-Whisker Plot of Model 1 Using ggplot2",
       x = "Coefficient Estimates",
       y = "Predictors")
```



Exercise 3.2: Dot-and-Whisker Plot Using `dotwhisker` Package

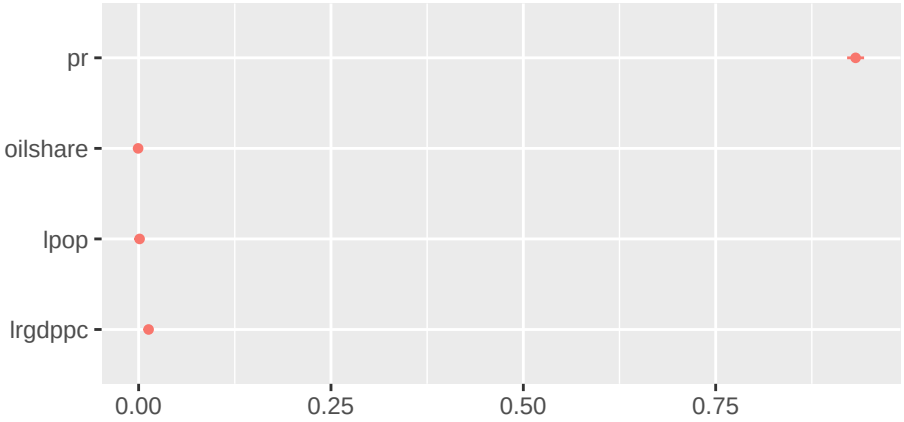
Next, we'll use the `dotwhisker` package to quickly generate dot-and-whisker plots for model `n2`. This package simplifies the process, especially when handling multiple regression models.

Helpful Hint: Remember to tidy the model using the `tidy()` function from the `broom` package before

plotting the graph. Let's call them `n2df`.

```
# Convert regression results into tidy dataframes
n2df <- tidy(n2)

# Generate a dot-and-whisker plot for Model 2
dwplot(n2df)
```

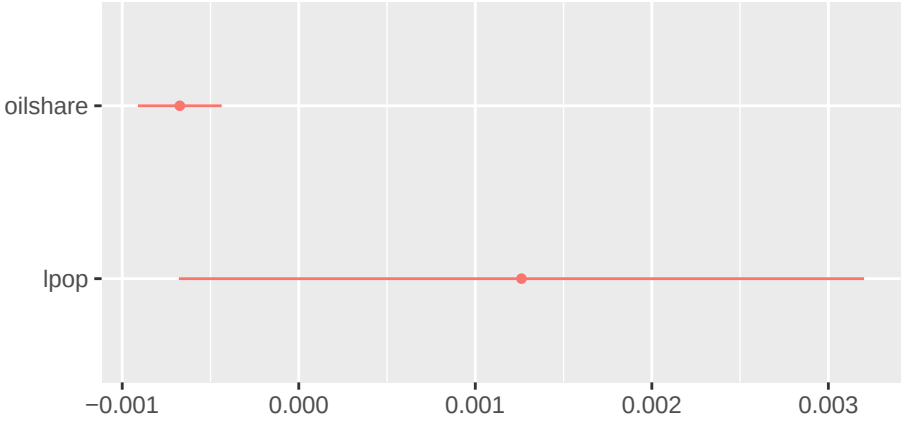


Exercise 3.3: Focused Dot-and-Whisker Plot on `oilshare` and `lpop`.

Let's focus on the influence of key variables in model `n2` by generating a dot-and-whisker plot that zooms in on the logged population (`lpop`) and `oilshare`.

```
# Filter 'oilshare' and 'lpop'
n2df <- tidy(n2) %>%
  filter(term == "oilshare" | term == "lpop")

# Generate a dot-and-whisker plot for Model 2 focusing on focused plot on 'oilshare' and
# 'lpop'
dwplot(n2df)
```



Exercise 4: Plotting Results from Multiple Regression Models

Finally, let's display results from all three regression models (`n1`, `n2`, `n3`) in one plot, focusing on key variables `oilshare`, `lrgdppc`, and `educ`. Remember to tidy the regression models `n1`, `n2` and `n3`, filter the variables `oilshare`, `lrgdppc`, and `educ`, and add a column called `Model (model number)` before combining the dataframes with `rbind()`. Save the last version of your plot as a `.png` file using `ggsave()`.

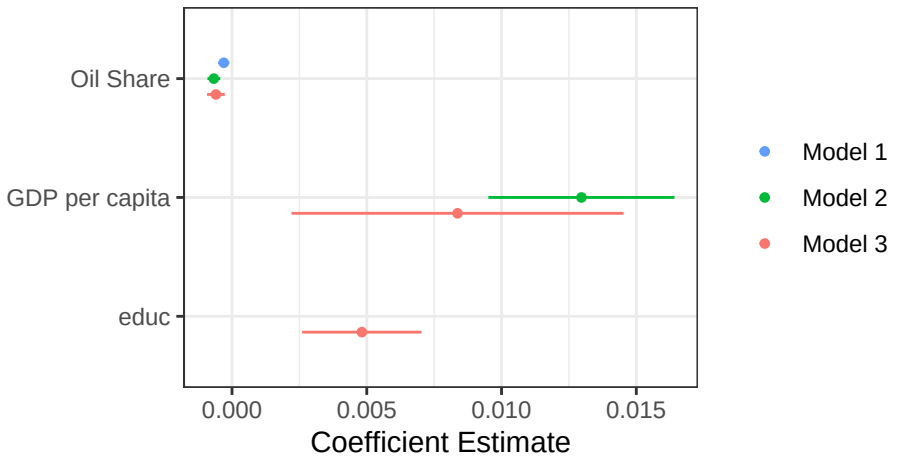
Helpful Hint: The `ggsave()` structure is: `ggsave("your filepath/your filename.png"...)`

```
# Tidy up the results from model n1, n2 and n3
n1df <- tidy(n1) %>%
  mutate(model = "Model 1")
  # Assigning the label 'Model 1' to the rows
n2df <- tidy(n2) %>%
  mutate(model = "Model 2")
  # Assigning the label 'Model 2' to the rows
n3df <- tidy(n3) %>%
  mutate(model = "Model 3")
  # Assigning the label 'Model 3' to the rows

# Combine the tidied dataframes into one
models <- rbind(n1df, n2df, n3df)

# Create a dot-and-whisker plot for variable 'oilshare','lrgdppc', and 'educ' across models
plot3 <- dwplot(models,
       model_order = c("Model 1","Model 2","Model 3"),
       # Specify the order of models in the plot
       vars_order = c("oilshare","lrgdppc","educ")) %>%
       # Select the order of the variables in the plot
  relabel_predictors(c(oilshare = "Oil Share",
                       lrgdppc = "GDP per capita",
                       open = "Education")) +
                       # Relabel the variables' names
  theme_bw() +
  # Applies a minimalistic black and white theme
  xlab("Coefficient Estimate") +
  ylab("") +
  # Label the axis
  theme(legend.title=element_blank())
  # Remove the legend title

plot3
```




```
# Save the plot
ggsave("dwplot1_oildem.png", plot3, width=6.5, height=4.5, dpi=400)
```

Before we finish this groupwork assignment, save the final version of your dataset as a `.rds` file called

`reg_aslaksen2010.rds`. We will use it for homework!

```
# Save the dataset to an RDS file
saveRDS(as2010, "reg_aslaksen2010.rds")
```