7316 - INTRODUCTION TO R

Module 4: Correction of the exercise

Teacher: Mickaël Buffart (mickael.buffart@hhs.se)

In this assignment you will replicate the main findings from the paper "Female Socialization: How Daughters Affect Their Legislator Fathers' Voting on Women's". Along the way you will also replicate a figure.

Important: The **README** file contains a description of the variables in the dataset.

1 REPLICATE FIGURE 1

Figure 1 shows the mean NOW score (a score that measures how closely a politician's votes align with recommendations of the National Organization of Women) by party and number of female children.

- Make sure the relevant variables have the correct data type.

```
# First, we load the data
df <- rio::import("data/basic.dta")

# party is a factor variable, 3 levels (see str of df):
#  1: dem
#  2: rep
#  3: indep
df$party <- as.factor(df$party)
levels(df$party) <- c("Democrat", "Republican", "Independent")

# congress is also a factor
df$congress <- as.factor(df$congress)
```

- Recode all the independent politicians to Democrat (there's only one).

```
# Recode independent
df$party[df$party == "Independent"] <- "Democrat"

# Drop the unused level
df$party <- droplevels(df$party)
```

- Remove observations where the number of daughters is missing.

```
sum(is.na(df$ngirls))
## [1] 4
# --> 4 missing values

# Removing obs where missing
df <- df[!is.na(df$ngirls),]
```

- NOW scores are only available for the 105^{th} congress. Select this subset.

```
# Removing obs where missing
df_105 <- df[c(df$congress == "105"),]
```

Figure 1 contains interesting information but it is not very pretty. Let's make it look a bit more colorful.

- Use *ggplot2* to replicate figure 1 (with slightly different coloring)

1. Filter the data for the subset of observation from the 105^{th} congress with all politicians that have either 2 or 3 children

```
df_105_2 <- df_105[c(df_105$totchi == 2 |
                       df_105$totchi == 3),]
```

2. Create a ggplot object with that data set.

```
library(ggplot2)

g <- ggplot(df_105_2)
```

3. Add an *aes* layer with the x dimension being the number of daughters and the y dimension being the total NOW score. Set the *col* and *fill* option to *party* because we want Republicans and Democrats colored differently.

```
g <- g + aes(x = ngirls, y = nowtot,
             color = party, fill = party)
```

4. Add a geometry layer: use `stat_summary`, inside which you choose the function "mean" and the geometry "bar". Also set `inherit.aes` to `TRUE`, such that the filling and shading parameters are taken over from the `aes` layer.

```
# Following the instruction, this is what is expected:
g <- g + stat_summary(fun = "mean",
                      geom = "bar",
                      inherit.aes = TRUE)

# You could have as well written this, for the same result
#  g <- g + geom_bar(stat = "summary", fun = "mean")
```

5. Add a facet grid that splits the chart along `party` (column) and `totchi` (rows). Use the `margins` option to generate the bar chart that combines democrats and republicans.

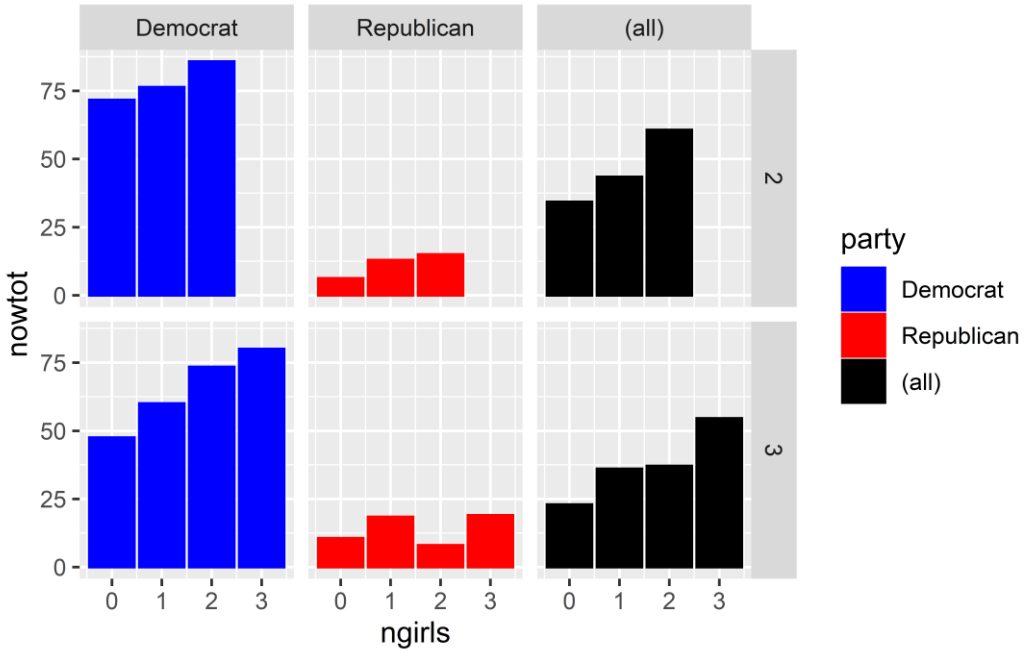
```
g <- g + facet_grid(cols = vars(party),
                    rows = vars(totchi),
                    margins = "party")
```

6. Add a color layer using *scale_color_manual* where the breaks are the parties (incl. "(all)") and the values are the colors you want to assign each party. Use the *aesthetics* option to specify that you want to change both the color and the filling.

```
g <- g + scale_colour_manual(breaks = c("Democrat", "Republican", "(all)"),
                             values = c("blue",     "red",        "black"),
                             aesthetics = c("colour", "fill"))
```

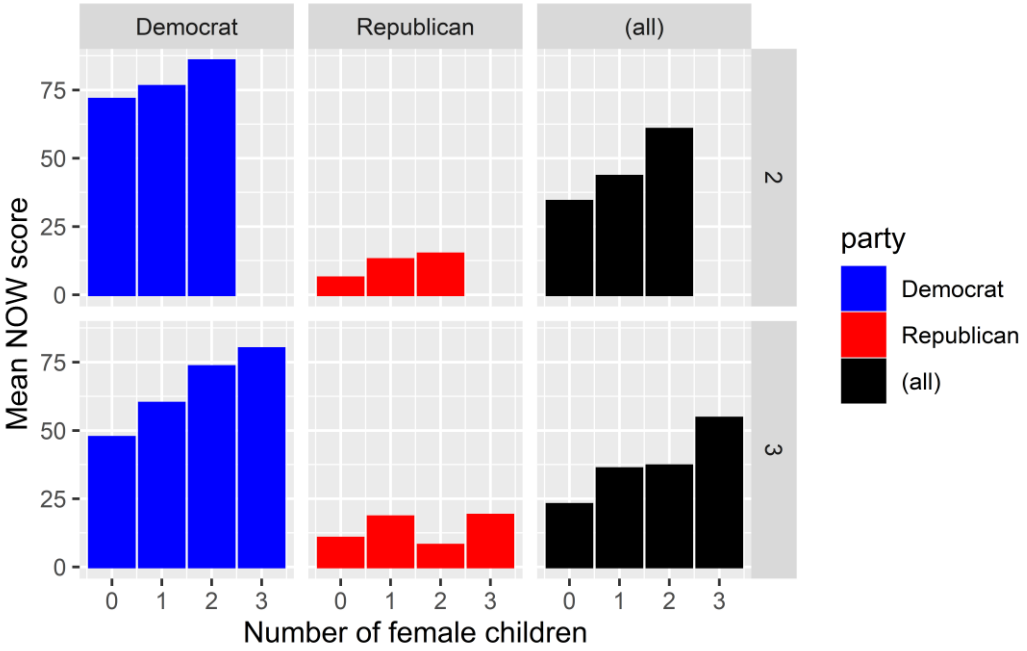
- Display the figure.

```
g
```



This is just an example of how this could look like. Feel free to play around with the parameters to make the plot prettier.

```
# For example, you could add labels
g + labs(x = "Number of female children",
         y = "Mean NOW score")
```



2 Replicate Table 2

2.1 Data Preparation

- Create all extra variables that you need for the regression. Don't forget the two fixed effects for *region* and *number of children*. (Fixed effects in this context means simply coding these variables as factors)

```r
# We need the square length of service
df$srvlng2 <- df$srvlng^2

# a dummy for republicans
df$republican <- c(df$party == "Republican")

# region is a factor
df$region <- as.factor(df$region)

# we also need a factor for the N of children
df$nchild_fac <- as.factor(df$totchi)
```

- Check the data type of the variables that you will need. Convert them if necessary.

```r
# Some of the dummy are coded as numeric (0/1), not as logical. We could recod
e them, but for R, it doesn't matter. We can keep them as they are.

# Others, we recoded above already.
```

- Recode the Religion as a factor variable that it aligns with the regressors in the table. Set *protestant* as the reference group.

```r
# First, setting religion as factor
df$rgroup <- factor(df$rgroup,
                    levels = c(0, 1, 2, 3, 4),
                    labels = c("No Religion", "Protestant",
                               "Catholic", "Other Christian",
                               "Other Religion"))

# relevel allows you to set a reference level (the baseline)
df$rgroup <- relevel(df$rgroup, "Protestant")

# Age is squared
df$age2 <- df$age^2
```

2.2 Regressions

Table 2 shows the coefficients from the model $Score = \beta_0 + \beta_1 ngirls + \beta_2 female + \beta_3 White + \ldots + \beta_{13} Dem.vote.share$ where score is either the NOW for the 105^{th} Congress or the AAUW score for each congress from the 105^{th} to the 108^{th}.

- Run the regression from the first column where the dependent variable is the total NOW score using the $lm(\ \ )$ function. **Hint:** if you selected only the relevant variables beforehand, you can just type $nowtot \sim .$ as the formula.
- Save the regression output as an object.

```r
# Select 105th again with the new variable from df
df_105 <- df[df$congress == "105",]

# Select the relevant subset of variables
#  Note: the variables will appear in your regression model in the order you l
ist them:
data_reg <- df_105[, c("nowtot",
                       "ngirls", "female", "white", "republican",
                       "age", "age2", "srvlng", "srvlng2",
                       "rgroup", "demvote", "nchild_fac", "region")]

model_now <- lm(nowtot ~ ., data = data_reg)
```

- Write a loop that runs the regression with the AAUW score as the dependent variable for each congress separately.

```
# create a vector of the congress
congress <- unique(df$congress)

# Creating an empty list that we will append with
list_models_aauw <- list()

# The loop
for (i in 1:length(congress)) {
  # Select the congress one by one
  specific_congress <- df[df$congress == congress[i],]

  # Select variables in the model. Note: the DV is "aauw" here
  data_reg <- specific_congress[, c("aauw",
                                    "ngirls",
                                    "female", "white", "republican",
                                    "age", "age2",
                                    "srvlng", "srvlng2",
                                    "rgroup", "demvote",
                                    "nchild_fac", "region")]

  # Fit model
  model_tmp <- lm(aauw ~ ., data = data_reg)

  # We append the list for each new model
  list_models_aauw[[i]] <- model_tmp
}
```

- Combine the NOW score regression and the AAUW regressions into one table using *stargazer*.
- omit the region and number of children fixed effects from the output
- keep the number of observations as the only statistic
- set the type to html

```
stargazer::stargazer(
  # NOW score regression
  model_now,

  # AAUW regressions
  list_models_aauw,

  # omit the region and number of children fixed effects from the output
  omit = c("region", "nchild_fac"),

  # keep the number of observations as the only statistic
  keep.stat = c("n"),

  # set the type to html
  type = 'html',

  # save it somewhere
  out = 'output_table.html'
)
```

	Dependent variable:				
	nowtot	aauw			
	(1)	(2)	(3)	(4)	(5)
ngirls	2.300**	2.385**	1.688	2.421**	2.250*
	(1.038)	(1.124)	(1.136)	(1.092)	(1.145)
female	10.838***	9.194***	10.440***	7.564***	6.909**
	(2.690)	(2.910)	(2.881)	(2.618)	(2.730)
white	1.860	0.144	2.594	-2.626	1.939
	(3.451)	(3.676)	(3.833)	(3.150)	(3.211)
republican	-44.903***	-60.468***	-55.927***	-63.216***	-63.932***
	(2.109)	(2.280)	(2.335)	(2.120)	(2.436)
age	0.661	0.854	2.034**	1.298	2.300***
	(0.800)	(0.860)	(0.902)	(0.799)	(0.860)
age2	-0.006	-0.006	-0.018**	-0.012	-0.021**
	(0.008)	(0.008)	(0.009)	(0.007)	(0.008)
srvlng	0.235	-0.208	-0.726*	-0.100	-0.139
	(0.300)	(0.324)	(0.380)	(0.346)	(0.327)
srvlng2	-0.009	0.004	0.025*	-0.0002	0.004
	(0.010)	(0.011)	(0.013)	(0.011)	(0.010)
rgroupNo Religion	7.262	5.671	5.353	7.027	-7.137
	(7.022)	(7.606)	(7.791)	(7.176)	(7.499)
rgroupCatholic	-3.974**	-4.505**	-2.277	-4.016**	-5.468***
	(1.941)	(2.091)	(2.129)	(1.987)	(2.077)
rgroupOther Christian	0.769	3.204	1.690	1.646	3.874
	(4.600)	(4.983)	(4.912)	(4.491)	(4.684)
rgroupOther Religion	10.866***	9.683**	11.890***	10.286***	3.156
	(3.752)	(4.047)	(4.343)	(3.785)	(3.959)
demvote	84.158***	62.148***	57.437***	56.206***	66.950***
	(10.869)	(11.568)	(12.018)	(9.092)	(10.889)
Constant	17.025	36.451	9.384	27.445	1.870
	(21.662)	(23.401)	(25.004)	(21.914)	(24.069)
Observations	430	434	434	434	433
Note:					*p**p***p<0.01

- *Hint:* set *results = 'asis'* in the code chunk header to properly knit the html code.
- This hint is useful when you use Rmarkdown to produce documents. We will see what it means during the next class.
- To go further, you can play with the design of your table: change the style, add labels, try different options…
- **Note:** Once you saved your table into `.html`, you can open the file into Word, and modify the table, or copy-paste it into any Word document, as a Word table.

3 Table 4

The NOW score is composed of several sub-scores for different women's rights issues, such as reproductive rights, equal pay or violence against women. For each issue, politicians are rated either zero (does not agree with NOW on this issue) or one (does agree).

Take a look at Table 4. In each row we run the regression from above but we replace the dependent variable $nowtot$ with a sub component $now1$ to $now20$. Each row then shows the coefficient of $ngirls$ from that regression.

- Use the subset from the 105^{th} congress.

```
# We saved it in df_105.
```

- Write a loop that runs the regressions from Table 4. Each regression should be the same model but with a different now subscore on the left hand side (now1 - now20).

```
# Creating an empty list
list_models_now <- list()

# We fit models from now1 to now20
for (i in 1:20) {
  dv <- df_105[, paste0("now", i)]

  # Select the ivs in the model. Note: the DV is OUT of the df
  data_reg <- df_105[, c("ngirls",
                         "female", "white", "republican",
                         "age", "age2",
                         "srvlng", "srvlng2",
                         "rgroup", "demvote",
                         "nchild_fac", "region")]

  # Here, it is more tricky, because we do not know the name of the DV,
  #  from now1 to now20. But we can extract it from the dataset while
  #  concatening "now" and the increment i.
  model_tmp <- lm(dv ~ . , data = data_reg)

  # We append the list for each new model
  list_models_now[[i]] <- model_tmp
}
```

- Extract the coefficients of $ngirls$ from the regressions.
- Extract the standard errors of $ngirls$ from the regressions.

```
# You don't need to use tidy here, you can simply use R base

# Setting up the vectors
ngirls_coef <- numeric()
ngirls_se <- numeric()

# 20 coefficient for 20 models
for (i in 1:20) {
  # Extract the output of the model
  tmp <- summary(list_models_now[[i]])

  # Extracting the coefficient of ngirls from tmp
  estim <- tmp$coefficients["ngirls", "Estimate"]
```

```
  # Extracting the coefficient of ngirls from tmp
  std_err <- tmp$coefficients["ngirls", "Std. Error"]

  # Append the vectors
  ngirls_coef <- c(ngirls_coef, estim)
  ngirls_se <- c(ngirls_se, std_err)
}
```

- Calculate the significance of each coefficient using a standard t-test $\frac{coefficient}{st.error}$. Add a column where you print one asterisk for significance at the 10 % level, two for significance at the 5 % level and three asterisks for 1% significance.

```
# No significance: empty character
significance <- character(length = 20)

# Sig level, 10%
significance[c(ngirls_coef / ngirls_se) > 1.65] <- "*"

# Sig level, 5%
significance[c(ngirls_coef / ngirls_se) > 1.96] <- "**"

# Sig level, 1%
significance[c(ngirls_coef / ngirls_se) > 2.58] <- "***"

# Warning: order is important, otherwise, previous condition is cancelled
#  by the next one.
```

- Bind the coefficients and standard errors into one table as in Table 4 in the paper. Round the coefficients and standard errors to three digits after the decimal.
- Add a column where you just number each score (unfortunately, the data does not contain labels for the now scores)

```
# Round the coefficients and standard errors to 3 digits
ngirls_coef <- round(ngirls_coef, 3)
ngirls_se <- round(ngirls_se, 3)

# Bind in a table
table_4 <- data.frame(now_score = paste0("now", 1:20),
                      coefficient = ngirls_coef,
                      std.error = ngirls_se,
                      significance = significance)
```

- Print the table using *stargazer*. Set *summary = FALSE* to print the data frame instead of the summary statistics.

```
# To make is look more like in the article, we need to concatenate
#   the significant and the standard errors

table_4$std.error <- paste0("(", table_4$std.error, ")",
                            table_4$significance)

# Significance can be removed: it is reported with standard errors
table_4$significance <- NULL

# print using stargazer
stargazer::stargazer(table_4,
                     summary = FALSE,
                     type = 'html',
                     out = 'table_4.html')
```

	now_score	coefficient	std.error
1	now1	-0.002	(0.019)
2	now2	0.003	(0.021)
3	now3	0.035	(0.019)*
4	now4	0.037	(0.02)*
5	now5	0.032	(0.024)
6	now6	0.030	(0.023)
7	now7	0.049	(0.023)* *
8	now8	0.034	(0.022)
9	now9	0.047	(0.025)*
10	now10	0.034	(0.021)
11	now11	0.027	(0.022)
12	now12	0.016	(0.023)
13	now13	0.030	(0.018)*
14	now14	-0.007	(0.02)
15	now15	0.011	(0.02)
16	now16	0.017	(0.023)
17	now17	0.033	(0.015)* *
18	now18	0.025	(0.022)
19	now19	0.013	(0.021)
20	now20	-0.006	(0.015)