# Reproducibility and Literate Programming

Rachel

2022-10-19

For this portfolio I will be using data obtained from the World Bank's DataBank on Gender Statistics [1] to demonstrate how one might use literate programming to produce code that can easily be recreated by anyone reading this report. This portfolio focuses on processing the data so it can then be ready for analysis.

We first load the packages that we'll be making use of in our analysis

```
library(tidyr)
library(dplyr)
library(readr)
library(purrr)
library(corrplot)
library(sjmisc)
library(kableExtra)
```

Since there are approximately 1000 possible metrics and 265 countries to consider for a period of >20 years, I will only be focusing on the employment data for 2020, which I have downloaded into a CSV file directly from the website. To load this into our environment we can run the following (as long as the data is in the same directory as the markdown file)

```
data <- read_csv("Raw_Gender_Data.csv", show_col_types = FALSE)
```

If we look at a summary of the first two columns we can see they are just denoting the year, which is constant and thus not useful, thus we remove it. The fourth column contains a unique series code and the third column contains a longer description of the metric. Since this is the same information, we can remove the less concise third column. If we want to retrieve it, the data comes with a CSV file containing the series name for each series column.

```
data <- data[,-c(1,2,3)]
```

Then we might want to take a brief visual inspection of our data (for simplicity we just look at the first 15 rows and 5 countries):

```
knitr::kable(data[1:15,1:6],booktabs = TRUE) %>%
  kable_styling(font_size = 8, latex_options = "HOLD_position")
```

[1] https://databank.worldbank.org/reports.aspx?source=gender-statistics#

| Series Code | United Kingdom [GBR] | Afghanistan [AFG] | Albania [ALB] | Algeria [DZA] | American Samoa [ASM] |
|---|---|---|---|---|---|
| SG.GET.JOBS.EQ | 1 | 1 | 1 | 1 | .. |
| SG.NGT.WORK.EQ | 1 | 0 | 1 | 0 | .. |
| SG.DNG.WORK.DN.EQ | 1 | 0 | 1 | 0 | .. |
| SG.IND.WORK.EQ | 1 | 0 | 1 | 1 | .. |
| SH.HIV.ARTC.FE.ZS | .. | 9 | 53 | 87 | .. |
| SH.HIV.ARTC.MA.ZS | .. | 9 | 45 | 80 | .. |
| SE.PRM.TENR | .. | .. | .. | .. | .. |
| SE.PRM.TENR.FE | .. | .. | .. | .. | .. |
| SE.PRM.TENR.MA | .. | .. | .. | .. | .. |
| SP.ADO.TFRT | 11.183 | 57.509 | 19.4332 | 9.3594 | .. |
| SH.HIV.PMTC.ZS | .. | 10 | .. | 34 | .. |
| SH.STA.BRTC.ZS | .. | .. | .. | .. | .. |
| SH.DTH.COMM.ZS | .. | .. | .. | .. | .. |
| SH.DTH.COMM.0004.ZS | .. | .. | .. | .. | .. |
| SH.DTH.COMM.0004.FE.ZS | .. | .. | .. | .. | .. |

We can see that there are many missing values, although here they are represented by an ellipses. We can then change these to NA, as this is what R recognises as missing values, and then see how many missing values there are in each row

```
data <- data %>% na_if("..")
na_freq <- tabulate(rowSums(is.na(data)))
na_mat <- rbind( "Missing Values" = 1:length(na_freq), "Frequency" =na_freq)
na_mat <- na_mat[,!(na_mat[2,]==0)]
na_mat
```

```
##                [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
## Missing Values   21   22   23   24   28   30   32   33   38    43    47    49
## Frequency         1    1    3   11   30   25    1    2    2    12     1     2
##                [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22]
## Missing Values    50    54    57    58    61    63    84    89    90   117
## Frequency          2     3     1     2     1     8     3     3     5     3
##                [,23] [,24] [,25] [,26] [,27] [,28] [,29] [,30] [,31] [,32]
## Missing Values   118   119   120   121   123   127   132   133   134   135
## Frequency          3     3     5     3     6     1     2     3     1     2
##                [,33] [,34] [,35] [,36] [,37] [,38] [,39] [,40] [,41] [,42]
## Missing Values   136   137   138   142   143   145   147   150   153   155
## Frequency          6     4     3     3     5     4     1     1     3     3
##                [,43] [,44] [,45] [,46] [,47] [,48] [,49] [,50] [,51] [,52]
## Missing Values   156   159   160   161   163   165   172   175   179   181
## Frequency          2     4     1     1     3     1     1     2     9     4
##                [,53] [,54] [,55] [,56] [,57] [,58] [,59] [,60] [,61] [,62]
## Missing Values   183   184   185   186   187   188   191   197   201   208
## Frequency          3     6     3     3     1     2     1     2     7     1
##                [,63] [,64] [,65] [,66] [,67] [,68] [,69] [,70] [,71]
## Missing Values   209   210   211   212   213   214   215   216   217
## Frequency          2     1     1    24    37    51    15   221     5
```

We can see there are many variables with 28 missing values, choosing these series to include in our analysis means we will have enough variables to hopefully find interesting patterns without dealing with too much missing data. Looking at the pattern of our data, it is plausible that the missing values for each series occurs for the same 28 countries. We can then remove the countries for which there are no entries. Finally we rotate the data frame to have each row corresponding to a country as this is more intuitive

```
data <- data[rowSums(is.na(data)) == 28, ]
data <- data[,colSums(is.na(data)) < nrow(data)]
data <-  rotate_df(data, cn = TRUE)
```

Now that we have a more workable dataset, we can save this data to a new file, so we can keep the raw data and processed data separately:

```
write_csv(data, "Processed_Gender_Data")
```