# Module 3 Exercise

```r
library(rio)
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------------------------------------

## v ggplot2 2.2.1     v purrr   0.2.4
## v tibble  1.4.2     v dplyr   0.7.4
## v tidyr   0.8.0     v stringr 1.2.0
## v readr   1.1.1     v forcats 0.2.0

## -- Conflicts ---------------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

## Instructions

### Download and import the data

For this exercise, you are going to primarily be working with data from the US Bureau of Labor Statistics *Consumer Expenditure Survey* (CEX). You can download the data here.

- Download the SAS **2016 Interview** file. Extract the data files and keep the following two files (moving them into a project folder):
  - *fmli161x.sas7bdat*, which corresponds to 2016Q1 data about household characteristics and income.
  - *mtbi161x.sas7bdat*, which corresponds to 2016Q1 data about household expenditures.
- Set the working directory.
- Import the two data files, calling them *characteristics* and *expenditures*. You may want to convert them to tibbles.

```r
characteristics <- import("./data/fmli161x.sas7bdat") %>% as.tibble()
expenditures <- import("./data/mtbi161x.sas7bdat")%>% as.tibble()
```

### Iterate a function over column names

- Change all the column names to lower case by iterating the function tolower() over the column names of the two data frames.

```r
colnames(characteristics) <- colnames(characteristics) %>% map(tolower)
colnames(expenditures) <- colnames(expenditures) %>% map(tolower)
```

### Keep specific columns

As you may've noticed, there's a lot (over 800) variables in the dataset. Let's reduce this only to the following variables:

- **For characteristics:** newid, hh_cu_q,educ_ref, creditx,region, fincbtxm
- **For expenditures:** newid, cost, ref_mo, ref_yr

```
characteristics <- characteristics %>% select(newid,hh_cu_q,educ_ref,creditx, region, fincbtxm)
expenditures <- expenditures %>% select(newid,cost,ref_mo,ref_yr)
```

### Rename columns

Rename the following variables:

- *hh_cu_q* to *hh_size*
- *fincbtxm* to *hh_income*

```
characteristics <- characteristics %>% rename(hh_size=hh_cu_q)
characteristics <- characteristics %>% rename(hh_income=fincbtxm)
```

### Change the class of columns

Make all the variables except for *newid* into numeric for both data frames, using a loop or map function.

```
characteristics[,-1] <- characteristics[,-1] %>% map(as.numeric)
expenditures[,-1] <- expenditures[,-1] %>% map(as.numeric)
```

### Sample 80% of observations for both datasets

- To make the joins in the next step a little more interesting, first modify the datasets so that they are only a 80% sample of the full datasets.

```
characteristics <- sample_frac(characteristics, 0.8)
expenditures <- sample_frac(expenditures, 0.8)
```

### Aggregate expenditures by household

For both datasets, *newid* is a unique identifier for household. In the *expenditures* dataset, each expenditure is entered in seperately so that each household shows up many times. Using the appropriate tidyverse functions, sum up the expenditures by *newid*, replacing *expenditures* with this aggregated information.

```
expenditures <- expenditures %>%
  group_by(newid) %>% summarize(cost = sum(cost))
```

### Practice different joins

Try using the different *join* functions covered in the module.

- In particular, first perform a traditional join that keeps all of the observations from *expenditures* and the columns of *expenditures* and *characteristics*. Save the result of this join as *cex_data*.
- Also try a join that keeps the columns of *expenditures* and *characteristics*, but only the observations in both datasets.
- Try a join that keeps only the columns of *expenditures*, with only the observations of *expenditures* that are not matched in *characteristics*.

```
cex_data <- left_join(expenditures, characteristics, by="newid")
cex_data_inner <- semi_join(expenditures, characteristics, by="newid")
cex_data_semi <- semi_join(expenditures, characteristics, by="newid")
cex_data_anti <- anti_join(expenditures, characteristics, by="newid")
```

### Use conditional statements to create indicators for region

Starting from *cex_data*, create indicators for each region value. You might find the unique() function helpful.

```
cex_data <- cex_data %>% mutate(region1 = ifelse(region == 1,1,0),
                    region2 = ifelse(region == 2,1,0),
                    region3 = ifelse(region == 3,1,0),
                    region4 = ifelse(region == 4,1,0),
                    region4 = ifelse(is.na(region),1,0)
                    )
```

### Write your own simple linear regression function

Finally, try your hand at writing a function. In particular, try to write a function that produces the coefficient in a linear regression. In matrix notation, the formula for $\hat{\beta}_{OLS}$ is:

$$\hat{\beta}_{OLS} = (X'X)^{-1}(X'y)$$

You will need some more matrix multiplication operators for this:

- solve(A) yields the inverse of matrix A.
- t() provides the transpose of matrix A.

Also, remember to add a column of ones to include an intercept in the model. You can make a vector of ones by using the rep() inside of vector or matrix definition.

Once you've finished writing the function, try running it to produce the parameter estimates from a regression of expenditures on any of the other variables in *cex_data*.

```
# Define OLS function
my_ols <- function(indvars,depvar){

  # Keep only observations with no missing values for indvars and depvar
  X <- indvars[(!is.na(indvars)) & (!is.na(depvar))]
  y <- depvar[(!is.na(indvars)) & (!is.na(depvar))]

  # Create constant vector
  ones_vec <- matrix(rep(1), length(X))

  # Create matrix X equal to constant vec and indvars
  X <- cbind(ones_vec, X)

  # Name constant column "constant"
  colnames(X)[1] <- "constant"

  # Solve for coefficients
  beta <- solve(t(X)%*%X) %*% (t(X)%*%y)
  colnames(beta) <- "Estimate"
```

```r
  # Convert to data frame
  beta <- as.data.frame(beta)

}

# Estimate Coefficients
coeffs_schooling <- my_ols(cex_data$educ_ref, cex_data$cost)
# Display results
coeffs_schooling
```

| | Estimate |
|---|---|
| constant | -92195.58 |
| X | 10720.39 |