BS2280 - Econometrics
R Workshop 1 - Introduction to R

In this computer lab we will have to achieve the following tasks/learning outcomes:

- import data and view the data
- label the variables
- create new variable
- sort the data

Preparing your workspace

Before you do each task, you need to prepare your workspace first.

Step 1. Create a folder called RWorkshop1

Step 2. Go to Blackboard – Learning and Teaching Activities – Week 2 – R Workshop 1, download datafile: crime.xls, save it in the RWorkshop1 folder you created in step 1

Step 3. Open Rstudio and set working directory

Menu bar → Click Session → Set Working Directory → Choose Directory → Select RWorkshop1 folder you created in step 1

Step 4. Create an R script

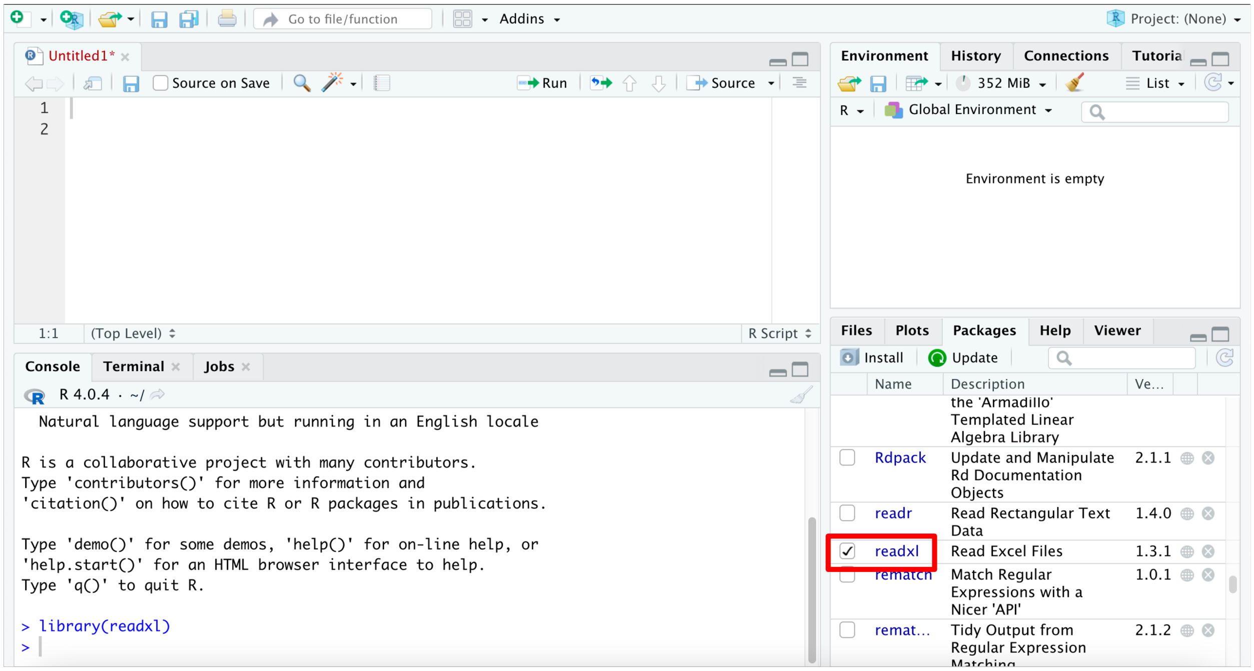
!! If you forget how to prepare your workspace, please review R Workshop 0 first.

For each task, replace the missing part XXXX to make these codes work.

Task 1. Import the Excel data set into R.

The native data format of R is .Rdata, however, you can also open other formats, such as .xlsx, .csv, etc. Non-native data formats have to be imported rather than just opened. Before we can import Excel spreadsheets directly into R, we have to activate a R-library first.

- You can use the package window (that's the window with the yellow border in Figure 1) and tick the box next to the package name.



```
R 4.0.4 · ~/
 Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> library(readxl)
> 
```

	Name	Description	Ve...
		the 'Armadillo' Templated Linear Algebra Library	
□	Rdpack	Update and Manipulate Rd Documentation Objects	2.1.1
□	readr	Read Rectangular Text Data	1.4.0
☑	readxl	Read Excel Files	1.3.1
□	rematch	Match Regular Expressions with a Nicer 'API'	1.0.1
□	remat...	Tidy Output from Regular Expression Matching	2.1.2

- Or you type the following command into the R script and click Run

```
library(readxl)
```

This line loads the necessary readxl library.

After you activate the readxl library, there are two ways to import:

Option 1. Use codes

We import the data set with the read_excel() and store this dataset as an object with the name 'crime'.

```
crime <- XXXX("crime.xls")
```

Option 2. Use menu bar

Menu bar → Click File → Import Dataset → From Excel

Don't forget to tick the "First Row as Names" box if it is unticked!

Task 2. View the data set in R's data viewer.

To open the data viewer, type in the following command:

```
XXXX(crime)
```

View(), note the capital 'V' (remember, R is case-sensitice, so view does not work) lets you view the data in read-only mode.

Task 3. Label the variables using the definitions given above.

You have to attach a variable label to each variable. A function called “apply_labels” could be used to achieve this goal.

“apply_labels” can facilitate the allocation of lables to variablesis and it is stored in package “**expss**”.

Considering this is the first time for us to use “**exprss**”, we need to install package “**expss**” and then activate it using library() command.

```
install.packages("expss")
XXXX(expss)
```

```
crime <- apply_labels(crime,
                      pop = "actual population in number",
                      crimes = "total number of crimes",
                      unem = "unemployment rate (%)",
                      officers = "number of police officers",
                      pcinc = "per capita income, $",
                      area = "land area, square miles",
                      lawexpc = "law enforcement expenditure per capita, $")
```

Task 4. Create a new variable which measures the population density for each city.

To generate a new variable and add it to the existing crime data-set, we use the following command:

We know that:

population density = actual population/land area

```
crime$popdens <- crime$XXXX / crime$XXXX
```

crime$pop: $ doller operator here means we only pick variable pop from dataset crime

crime$area: $ doller operator here means we only pick variable areea from dataset crime

crime$popdens: we create a new variable called popdens in dataset crime, assign the value of using crime$pop divided by crime$area to it

You may wonder why we add crime$ in front of every variable. The reason is that R can store more than one data frame, matrix, list, vector etc., at the same time, so the prefix crime$ is necessary to avoid ambiguity.

Think of crime$ as an address where e.g. the variable pop stays. If you have loaded another data frame that contains a pop variable, R would know that we only want to use the variable from the crime dataset and not from the other data frame. There are library packages that can facilitate the process, however, we will not cover them in this module.

Note that the variable label was copied from the pop variable. I recommend to update the label with the method we introduced in Task 3.

Task 5. Sort the data with respect to the population density of each city.

Sorting data is a useful action to get a general feeling for the data, e.g. are there any outliers in the dataset? Are there any unusual patterns?

To change the order of the rows in a data frame, we will apply the order() command.

We first rank all observations with respect to the population density in crime dataset and store this information in a vector called rank. The rank vector contains indices that we can use to sort the crime data frame.

```
rank <- XXXX(crime$popdens)
```

We use rank to sort popdens from the smallest to the largest value and assign it to a new created dataset called crime.popens1

```
crime.popens1 <- crime[XXXX,]
```

To sort the data from the largest to the smallest number, we set the order argument decreasing to TRUE.

```
crime.popens2 <- crime[order(crime$popdens, decreasing = XXXX),]
```

We can save some time and space by merging the two steps into one line, however, it is sometimes easier to understand a command if it is split into separate stages.

Task 6. What is the minimum and maximum value for population density in the dataset?

The minimum and maximum values can be produced by generating standard descriptive statistics of the variables using summary() in R.

```
XXXX(crime$popdens)
```

Final comments:

Before you finish, save the dataset. Never overwrite your original data!

```
save(crime, file = "crime2.Rdata")
```

This command tells R to use the crime dataset and save it as ‘crime.Rdata’. Rdata is an R specific format. R can also save data in .csv format, that can be opened with any text editor or spreadsheet software:

```
write.csv(crime, file = "crime2.csv", row.names = FALSE)
```

Save you R scipt file by clicking disc icon.

Further Exercises

Now you are ready to answer the following questions on your own:

1. Find the minimum and maximum number of police officers in the data set.
2. Create a new variable which measures the crime rate per 1,000 of population.
3. Is the city with the highest number of police officers also the city with the highest crime density?
4. How many crimes occurred in the richest city?
5. Is the richest city also the one with the highest number of police officers?
6. What is the average unemployment rate across these 46 U.S. cities?
7. Does the city with the highest unemployment rate also have the highest crime level?