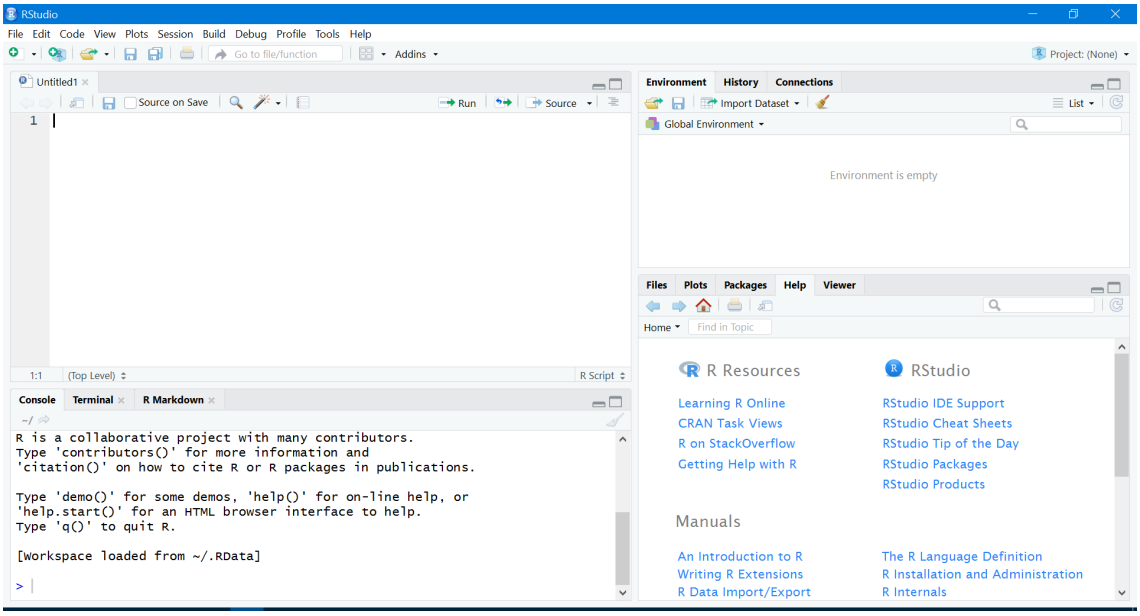
Quick Introduction to R and R Markdown

Jingyu Sun

R is a language and free software environment. It provides a set of powerful tools for data manipulation, statistical computing and graphical display.

Getting Started with R and RStudio

1. Install R by downloading the software from http://lib.stat.cmu.edu/R/CRAN/ and follow the installation instructions.
2. Install RStudio by downloading the software from https://www.rstudio.com/products/rstudio/download/ and follow the installation instructions.
3. Start RStudio by double-clicking on the icon, or choosing RStudio from the Start menu. The RStudio interface consists of a few panes.



- The top left pane shows the source files. If no files were open this pane is hidden. From the top menu, choose **File > New File > R script**. The top left pane will now show a blank document. For each file that you open, a new tab will appear in the top left pane, for easy navigation.
- The bottom left pane contains the console. You can enter commands and the output will be displayed there. For example, type `date()` and press the "Enter" key. Note that the ">" prompt indicates that R is ready to process the next commands. Some commands or functions may take time to process. While they are processing R will not process new commands.
- The top right pane contains a few tabs: usually you will see Environment and History, but other tabs may appear.
  - The **Environment** tab contains a table with all the variables, functions, and other data structures that were defined during the R session. When you start R you may see "Environment is empty"
  - The **History** tab contains a list of all the previously-entered commands.

- The bottom right pane contains a few tabs.
  - The **Files** tab shows the content of the current working directory
  - The **Plots** tab shows the plots that were generated during the session
  - The **Packages** tab shows a list of all installed packages, including a short description and the version
  - The **Help** tab shows the help page on commands. In the top right corner there is a text-box where you can enter the command name. Enter `date` in that box and read the documentation for the `date` function. To obtain documentation on a function `f` you can also enter `?f` in the Console. For example, enter `?date`.

Working with R

- In the console (bottom left pane), try the following

```
rnorm(5,0,1)
```

The `rnorm` command draws n random numbers from a normal distribution. Type `?rnorm` in the console to get more information about the command `rnorm`.

Now try the following:

```
x <- rnorm(5)
```

Note a few things:

1. This time no output was produced. The 5 random numbers were assigned to a variable called `x`. In the Environment tab you can find `x` and the 5 random numbers.
2. The `<-` operator is used to assign values.
3. Use meaningful variable names. When you write a long program, it's easy to forget what `x,y,z` stand for. User-defined variables and function names should not conflict with built-in commands. It causes confusion and may cause errors.
4. When you type `x` in the console, the content of `x` will be displayed in the console.
5. Any object (variable, function, etc) which you define remains in the program's memory for the duration of the session. When you close RStudio, you can choose to save the session, in which case all the defined objects can be available when you start Rtudio again.

- Try the following commands. We investigate the relationship between age and weight in a small sample.

```
age <- c(1,3,5,2,11,9,3,9,12,3)
weight <- c(4.4,5.3,7.2,5.2,8.5,7.3,6.0,10.4,10.2,6.1)
mean(weight)
sd(weight)
cor(age,weight)
plot(age,weight)
```

In this example we also introduce very useful commands: `mean, sd, cor`, and `plot`. To get more information about each one, use the `?` command to get more information about each command.

- If you want to use the output from a function to do other things, you may save it to a variable. For example, if you type `sdWgt <- sd(weight)` the output from the `sd` function will be saved to a variable called `sdWgt`. You can now use it to obtain the variance of the weights by using:

```
sdWgt <- sd(weight)
sdWgt^2
```

- Note that R is case-sensitive, which means that `qqplot` and `qqPlot` are not the same thing.

Comments in the R code

You can add comments to your R code to explain what the code is doing. Any characters that appear after the # symbol in the same line are considered comments and are ignored by R. For example,

```
age <- c(1,3,5,2,11,9,3,9,12,3)  # the age of the mice in our experiment (in weeks)
```

You should add clear comments to your code in order to help readers (including yourself) understand the program.

Workspace

The workspace contains your working environment, including user-defined objects, and the history of the session (the commands entered in the console). A useful feature of the user interface is that you can use the up and down arrows to scroll through the command history.

The `getwd()` command tells you the current working directory. It's important to know your working directory, because this is used as the default place to look for input files, and this is where your output will be saved. It's a good idea to use different directories for different projects. To set the working directory, use the `setwd("mydirectory")` command (replace `"mydirectory"` with a real directory name). In RStudio you can also do this by using the top menu: **Session > Set Working Directory**.

Other useful commands to manage the R workspace are listed in the following table:

Function	Action
`ls()`	Lists the objects in the current workspace
`rm(object list)`	Deletes object(s) from the workspace
`save(object list, file="filename")`	Saves specific object(s) to a file
`load("filename")`	Loads previously saved objects or a session to the current session
`q()`	Quits R

Installing and Using Packages

- The base installation of R contains many useful functions. However, you may augment the basic functionality by installing contributed **packages**. R packages are a collection of R functions. To install a package, use the command `install.packages("package_name")`.

For example, install the package `MASS` by typing in the console `install.packages("MASS")`

We will use the `rmarkdown` package extensively. Install the package on your computer.

- Every time you start an R session you have to load the packages that you will be using. Installing them does not make them automatically available! To load the `rmarkdown` package, type `library(rmarkdown)` in the console, and hit the "Enter" key.
- Packages come with detailed documentation, and some come with a demo, and/or vignettes. To obtain information about a package, type `help(package="package_name")` in the console. The documentation is also available online (on the CRAN website).

Dynamic documents and reproducible research

Knowing how to do statistical analysis is very important, but it's equally important to communicate results to others. This often involves three components:

1. Effective graphical representation (figures, tables)
2. Clearly stating the methods, results, and conclusions (text)
3. Submit the results in a convenient and aesthetic format.

R has several packages and built-in functions to produce nice graphs and tables. For example, we will use the **ggplot2** and **xtable** packages.

Not long ago, the only way to include R output in a report was to use copy and paste. Now, using knitr, you can generate reports in html, pdf, or Word format, and include all the data, code, and output, in one document. In this course, you are required to submit all your homework and the project using `knitr`, and the goal of this section is to give you the basic instructions.

`knitr` is a package that allows you to create reports and presentations in different formats (pdf, html, Word). With `knitr`, you can combine R code and the output from the execution of the R code with formatted text. The lecture and lab notes are written using `knitr`, and your homework have to be done using `knitr`.

- If you want to create pdf documents, you need to install a LaTeX compiler. A LaTeX compiler converts a LaTeX document into a high-quality typeset pdf document. We recommend that you install TinyTeX. TinyTeX is a lightweight and cross-platform LaTeX distribution. Type the following commands in the console to install TinyTeX.

```
install.packages("tinytex")
tinytex::install_tinytex()
```

- To create a new `knitr` file, click on **File > New File > R Markdown** in the main menu. A small window will open. Choose "Document", give it a title, and choose the output format. When you click "OK", RStudio will show a new file and it will include a short sample document. Click on the **Knit** button to generate the document.
- For a quick R Markdown reference guide, see https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-reference.pdf

Writing Math Equations

You can write mathematical expressions using LaTeX syntax. For instance, the simple linear regression model is:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

To obtain the above display, the corresponding LaTeX command is:

```
$$Y=\beta_0+\beta_1X+\epsilon$$
```

You can find an overview of the basic commands (subscripts and superscripts, fractions, integrals, sums and limits, list of Greek letters and math symbols, ...) here: https://www.overleaf.com/learn/latex/Mathematical_expressions