Lab #9: Introduction to R and RStudio

Lab Instructor: Mr. Zaeem Yousaf | Mr. Tayyub Shaheen

Course Instructor: Ms. Bushra Rashid

1. Introduction

The goal of this lab is to introduce you to R and RStudio, which you'll be using throughout the course both to learn the statistical concepts discussed in the texbook and also to analyze real data and come to informed conclusions. To straighten out which is which: R is the name of the programming language itself and RStudio is a convenient interface. As an analogy,

- R is like a car's engine,
- RStudio is like a car's dashboard

2. Installation of R and RStudio

If you have used R or RStudio before, I recommend **uninstalling the old versions** and **installing the latest version of R and RStudio** since some libraries that we'll install might not be compatible with older versions of R.

See the YouTube Videos below from OpenIntro Statistics (https://www.openintro.org/book/os/) for a video tutorial of

- How to Install R and RStudio on Windows (https://youtu.be/rHZ9MGWxU5I)
- How to Install R and RStudio on Mac OS X (covers both Intel and M1 processors) (https://youtu.be/AEebOXiMyyI)

or you can install following the text instructions below.

2.1 Install the latest version of R

Download the installation file here for…

- Windows (https://cran.r-project.org/bin/windows/base/): Click "Download R x.y.z for Windows". Then run downloaded .exe file to install. Agree to all of the installation defaults (unless you already know how to customize R).
- Mac (http://cran.r-project.org/bin/macosx/): Click on one of the "R-x.y.z.pkg" file depending on **the version of your Mac OS** to begin the installation. You can leave the default options as is just like for Windows. Please note that you might have to (re)install Xquartz (http://xquartz.macosforge.org/) if you use Mac OS X.
- Linux (http://cran.r-project.org/bin/linux/): I assume Linux users know how to install software on it yourself since I don't know. :)

2.2 Install the latest RStudio

RStudio provides a friendlier working environment for R

Download RStudio Desktop (https://rstudio.com/products/rstudio/download/#download). Select an isntaller based on your OS and then install.

2.3 RStudio Layout

The RStudio interface consists of several windows (see Figure 1 below).

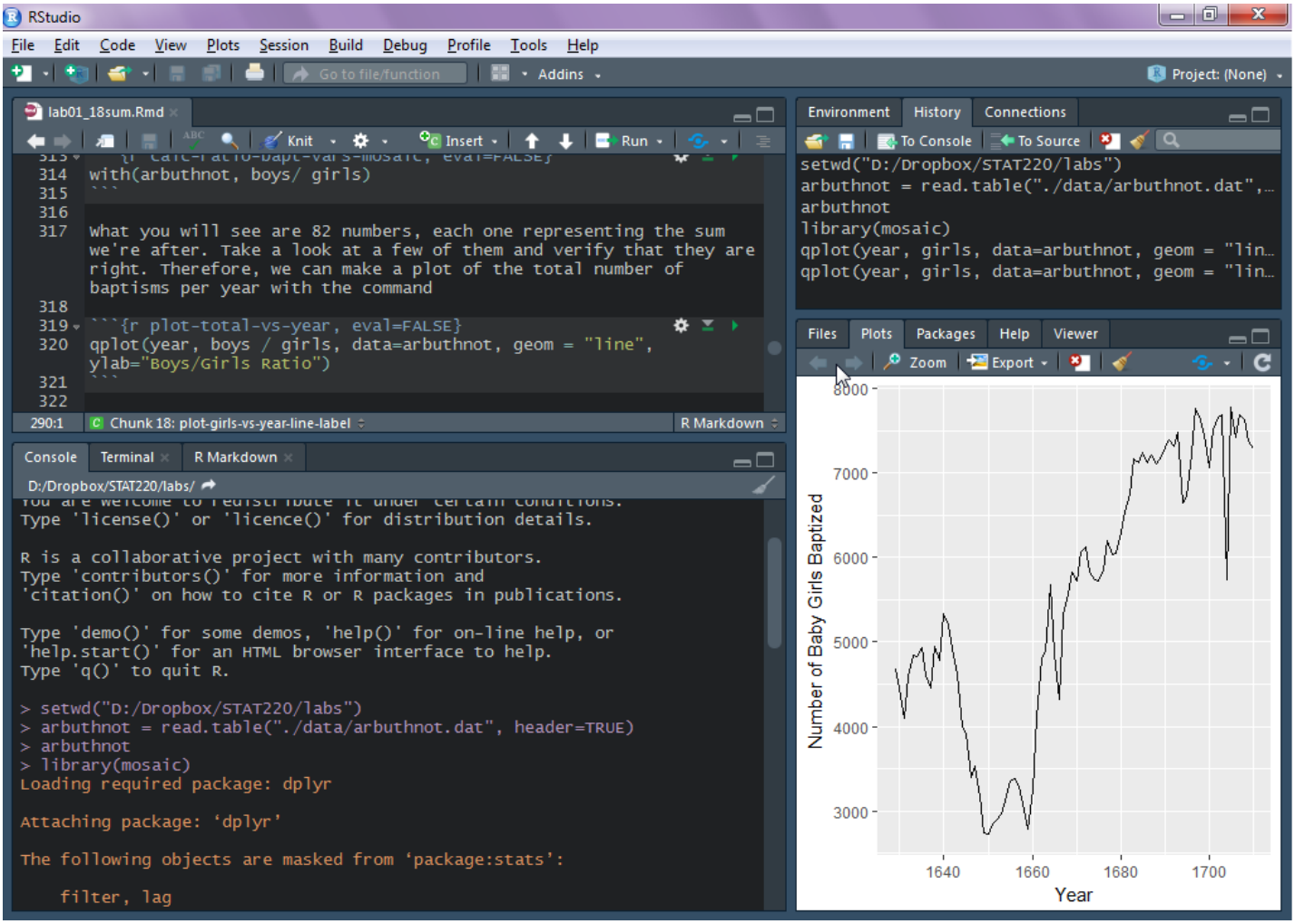


Figure 1

- Bottom left: **command window**. Here you can type simple commands after the “>” prompt and R will then execute your command. This is the most important window, because this is where R actually does stuff.
- Top left: **script window**. Collections of commands (scripts) can be edited and saved. When you don’t get this window, you can open it with [File] – [New] – [R script]. Just typing a command in the editor window is not enough, it has to get into the command window before R executes the command. If you want to run a line from the script window (or the whole script), you can click Run or press `CTRL+ENTER` to send it to the command window.
- Top right: **workspace** / **history window**. In the workspace window you can see which data and values R has in its memory. You can view and edit the values by clicking on them. The history window shows what has been typed before.

- Bottom right: **files** / **plots** / **packages** / **help window**. Here you can open files, view plots (also previous plots), install and load packages or use the help function.

You can change the size of the windows by dragging the grey bars between the windows.

2.4 The `ggplot2` Library

There are many *libraries* (also called *packages*) that can be installed to expand the standard R functions. For STAT220, we will install an additional package for R called `ggplot2`. The `ggplot2` library is a powerful R library for making fancy plots and visualizing data. After it's released in 2007, `ggplot2` was adopted worldwide, quickly replaced the build-in R functions: `plot()`, `hist()`, `boxplot()` for making plots, and become the dominating tools for data visualization. In Lab 2 we will see how useful `ggplot2` is.

To install the `ggplot2` package, copy and paste the following line to the command window and run it.

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb1-1)install.packages("ggplo
       t2")
```

This may take a few minutes to complete. During the installation process

- If R asks you to choose a CRAN mirror for downloading, we suggest choosing USA(IA) if you are in Chicago.
- If R asks to use your personal library answer yes.
- If R prompts you to update any packages, please agree to the updates.

Note that you will only have to install `ggplot2` **once**. However, once the package is installed, you will have to load it into the workspace before it can be used.

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb2-1)library(ggplot2)
```

Note that you will have to do this *every* time you start a new R session.

3. Getting Started

3.1 Working directory

Your working directory is the folder on your computer in which you are currently working. When you ask R to open a certain file, it will look in the working directory for this file, and when you tell R to save a data file or figure, it will save it in the working directory.

Before you start working, please **set your working directory** to the folder your data file is stored as follows.

First, go the [Files] pane on the bottom right, navigate to the directory you want.

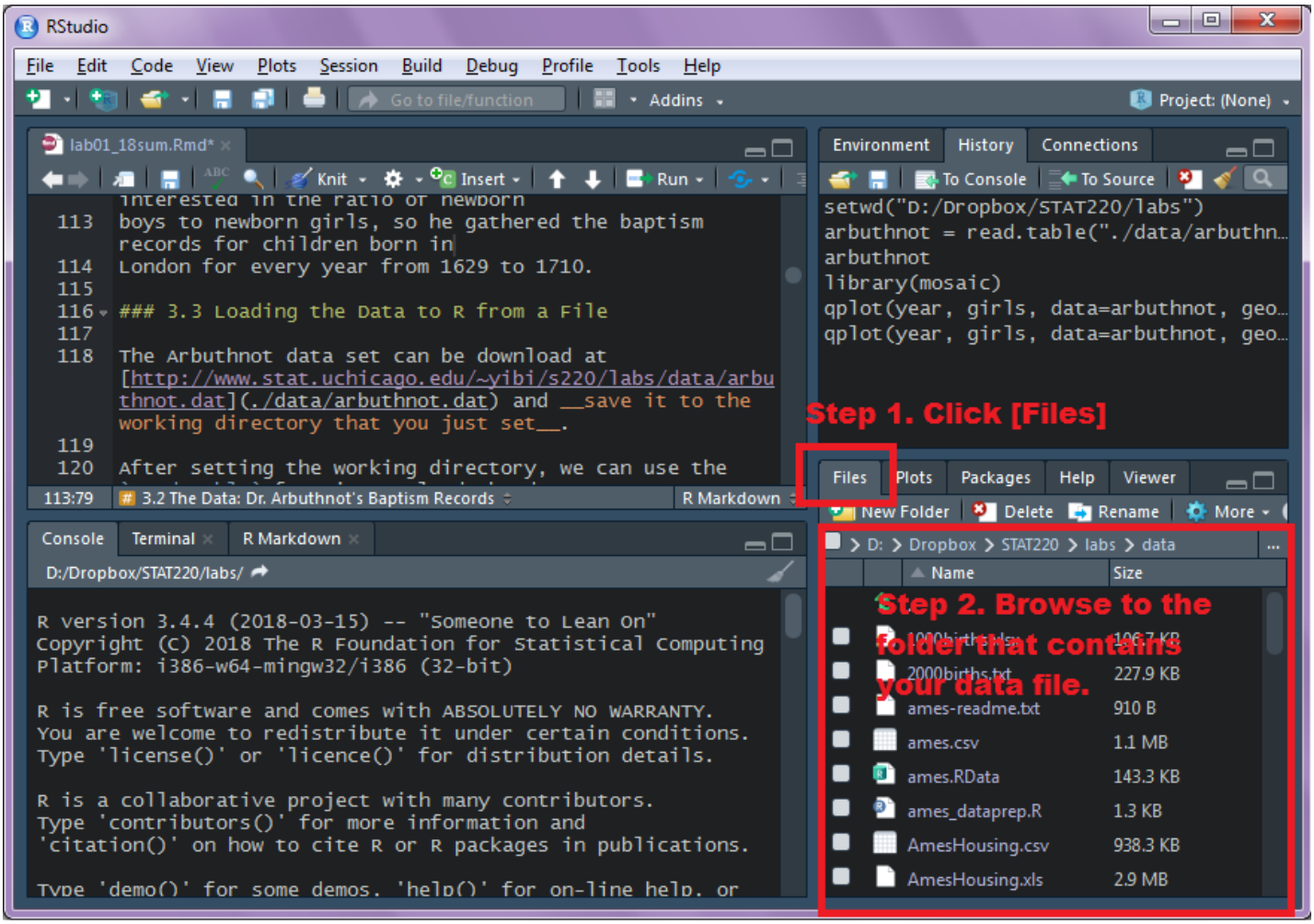
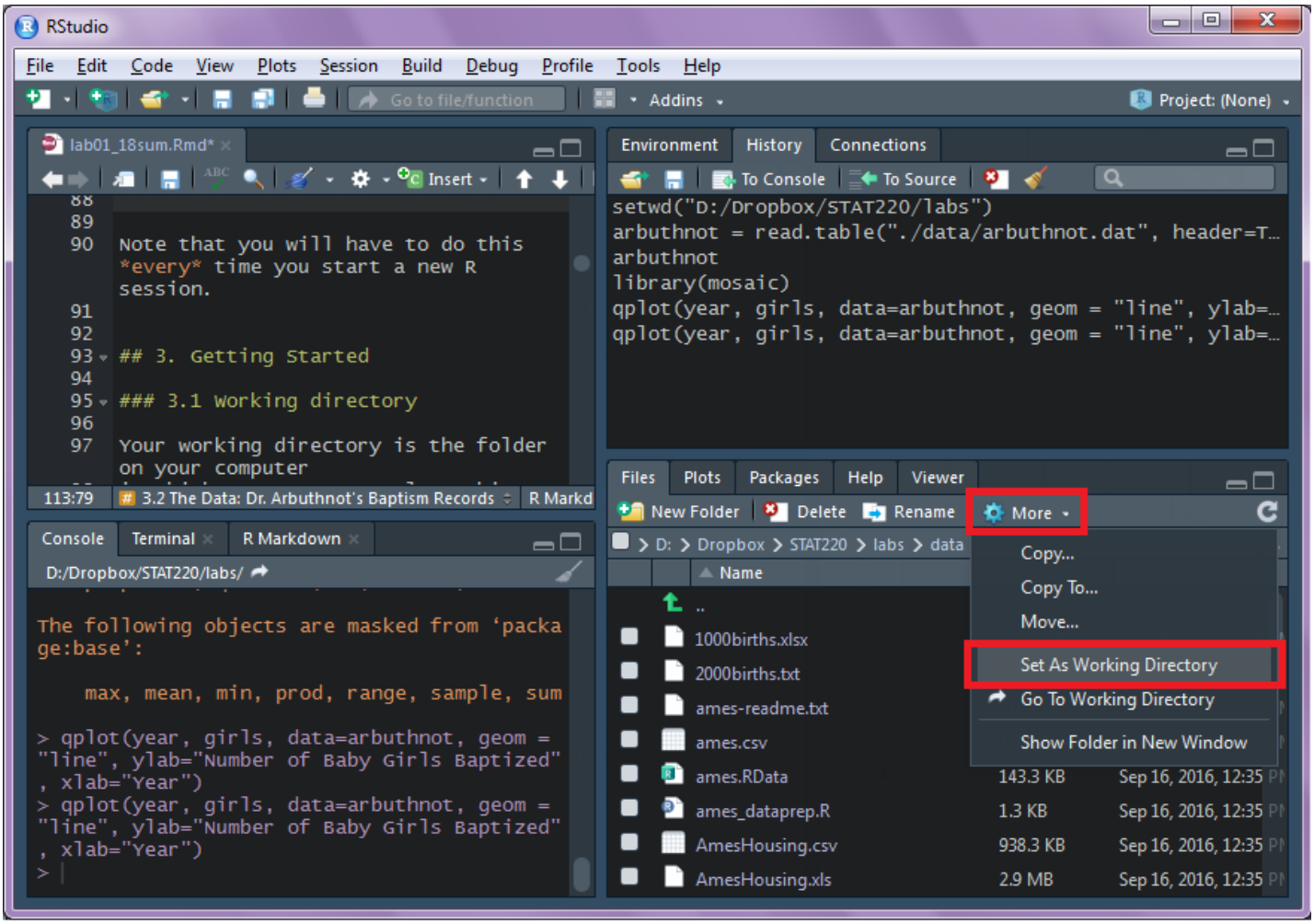


Figure 2

Then click [More] and then in the drop down menu click [Set As Working Directory].



3.2 The Data: Dr. Arbuthnot's Baptism Records

The Arbuthnot data set we are going to work on today comes from Dr. John Arbuthnot, an 18th century physician, writer, and mathematician. He was interested in the ratio of newborn boys to newborn girls, so he gathered the baptism records for children born in London for every year from 1629 to 1710.

3.3 Loading the Data to R from a File

The Arbuthnot data set can be download at http://www.stat.uchicago.edu/~yibi/s220/labs/data/arbuthnot.txt (https://www.stat.uchicago.edu/~yibi/s220/labs/data/arbuthnot.txt) and **save it to the working directory that you just set**.

After setting the working directory, we can use the `read.table` function to load the data set.

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb3-1)arbuthnot = read.table
       ("arbuthnot.txt", header=TRUE)
```

The argument `header=TRUE` means that the first line of the data file is the header of the data file, which contains the names the variables, and the data stars from the second line.

If the data set is loaded successfully, you should see that the workspace area in the upper righthand corner of the RStudio window now lists a data set called `arbuthnot` that has 82 observations on 3 variables.

Setting working directory and loading data file are two steps many students struggle with. Ask TA for help if you have any trouble.

3.4 A First Look at Arbuthnot's Data

We can take a look at the data by typing its name into the console.

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb4-1)arbuthnot
```

What you should see are four columns of numbers, each row representing a different year: the first entry in each row is simply the row number (an index we can use to access the data from individual years if we want), the second is the year, and the third and fourth are the numbers of boys and girls baptized that year, respectively. Use the scrollbar on the right side of the console window to examine the complete data set.

Note that the row numbers in the first column are not part of Arbuthnot's data. R adds them as part of its printout to help you make visual comparisons. You can think of them as the index that you see on the left side of a spreadsheet. In fact, the comparison to a spreadsheet will generally be helpful. R has stored Arbuthnot's data in a kind of spreadsheet or table called a *data frame*.

You can see the dimensions of this data frame by typing:

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb5-1)dim(arbuthnot)
```

This command should output `[1] 82 3`, indicating that there are 82 rows and 3 columns (we'll get to what the `[1]` means shortly), just as it says next to the object in your workspace. You can see the names of these columns (or variables) by typing:

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb6-1)names(arbuthnot)
```

You should see that the data frame contains the columns `year`, `boys`, and `girls`. At this point, you might notice that many of the commands in R look a lot like functions from math class; that is, invoking R commands means supplying a function with some number of arguments. The `dim` and `names` commands, for example, each took a single argument, the name of a data frame.

One advantage of RStudio is that it comes with a built-in data viewer. Click on the name `arbuthnot` in the *workspace* panel (upper right window) that lists the objects in your workspace. This will bring up an alternative display of the data set in the *Data Viewer* (upper left window). You can close the data viewer by clicking on the *x* in the upper left corner.

We can access a single variable of a data frame using the dollar-sign `$` notation. For example, the command below

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb7-1)arbuthnot$boys
```

```
##  [1] 5218 4858 4422 4994 5158 5035 5106 4917 4703 5359 5366 5518 5470 5460 4793
## [16] 4107 4047 3768 3796 3363 3079 2890 3231 3220 3196 3441 3655 3668 3396 3157
## [31] 3209 3724 4748 5216 5411 6041 5114 4678 5616 6073 6506 6278 6449 6443 6073
## [46] 6113 6058 6552 6423 6568 6247 6548 6822 6909 7577 7575 7484 7575 7737 7487
## [61] 7604 7909 7662 7602 7676 6985 7263 7632 8062 8426 7911 7578 8102 8031 7765
## [76] 6113 8366 7952 8379 8239 7840 7640
```

will return the `boys` variable in the data frame `arbuthnot`, which shows the number of boys baptized each year. Likewise `arbuthnot$girls` will return just the counts of girls baptized each year.

Notice that the way R has printed these data is different. When we looked at the complete data frame, we saw 82 rows, one on each line of the display. These data are no longer structured in a table with other variables, so they are displayed one right after another. Objects that print out in this way are called *vectors*; they represent a set of numbers. R has added numbers in [brackets] along the left side of the printout to indicate locations within the vector. For example, `5218` follows `[1]`, indicating that `5218` is the first entry in the vector. And the second line starts with `[15]` means that first number `4793` in the second line is the 15th entry in the vector, and so on for the rest of the output.

To see the element in the 56th row and the 3rd column (in this case, `girls`) of the table, we can type

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb9-1)arbuthnot[56,3]
```

To see the first 10 numbers in the 3rd column, we can type

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb10-1)arbuthnot[1:10,3]
```

In this expression, we have asked just for rows in the range 1 through 10. R uses the `:` to create a range of values, so 1:10 expands to 1, 2, 3, 4, 5, 6, 7, 8, 9, 10. You can see this by entering

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb11-1)1:10
```

Finally, if we want all of the data for the first 10 years, type

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb12-1)arbuthnot[1:10,]
```

By leaving out an index or a range (we didn't type anything between the comma and the square bracket), we get all the columns. When starting out in R, this is a bit counterintuitive. As a rule, we omit the column number to see all columns in a data frame. Similarly, if we leave out an index or range for the rows, we would access all the observations, not just the 56th, or rows 1 through 10. Try the following to see the 3rd variable for all 82 years

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb13-1)arbuthnot[,3]
```

Recall that column 3 represents the number of girls baptized in that year, so the command above should output the same list of numbers as `arbuthnot$girls`.

We see the number of girls baptized for the 56th year by typing

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb14-1)arbuthnot$girls[56]
```

Similarly, for just the first 10 years

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb15-1)arbuthnot$girls[1:10]
```

The command above returns the same result as the `arbuthnot[1:10,3]` command. Both row-and-column notation and dollar-sign notation are widely used, which one you choose to use depends on your personal preference.

3.5 Your First R Plot

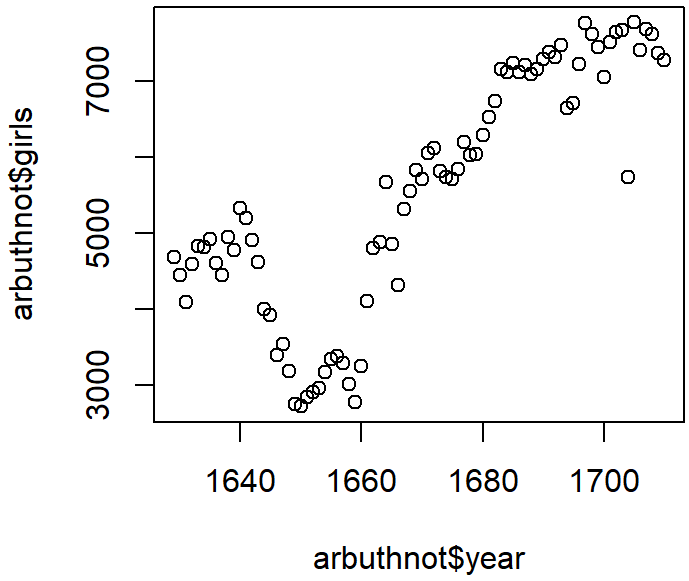
The `plot()` function in R can to make scatter plots. For example, we can create a simple plot with `year` as the x-variable and the number of `girls` baptized in a year as the yy-variable with the command below.

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb16-1)plot(x=year, y=girls)
```

```
## Error in plot(x = year, y = girls): object 'year' not found
```

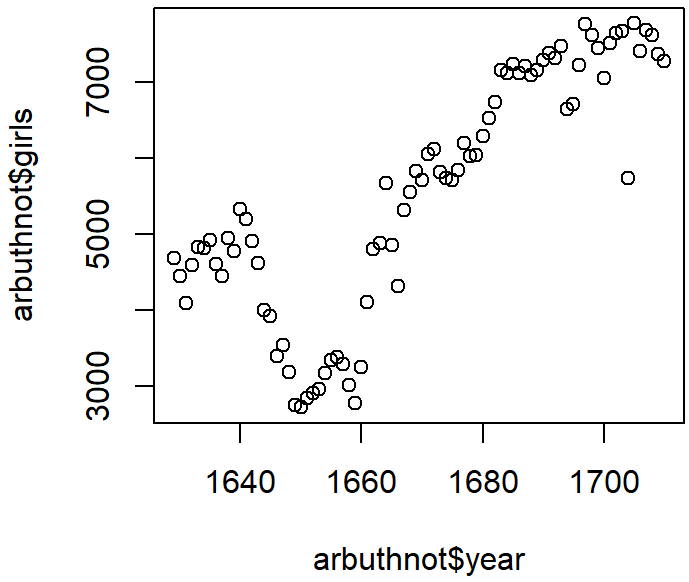
We get an error message because R cannot find the two variables `year` and `girls`, which are both under the `arbuthnot` data frame. We need to tell R that the two variables are under the `arbuthnot` data frame as follows.

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb18-1)plot(x=arbuthnot$year,
       y=arbuthnot$girls)
```



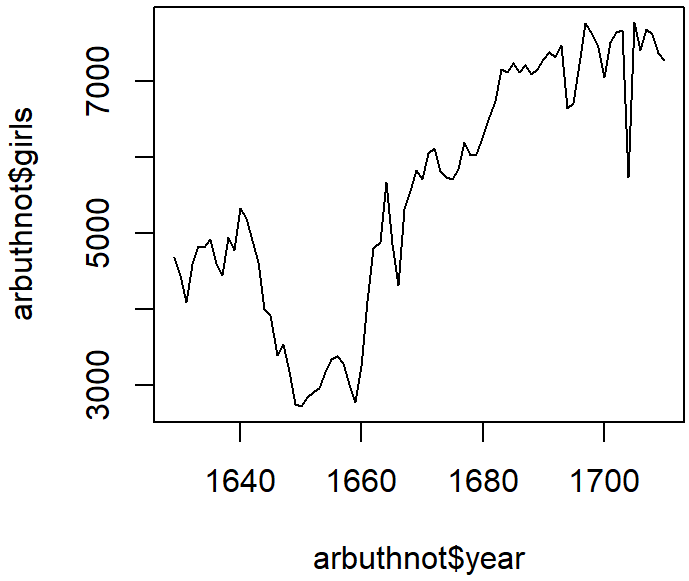
You can omit `x=` and `y=` in the command and get the same plot as R knows the first variable is the xx-variable and the 2nd is the yy-variable.

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb19-1)plot(arbuthnot$year, a
       rbuthnot$girls)
```



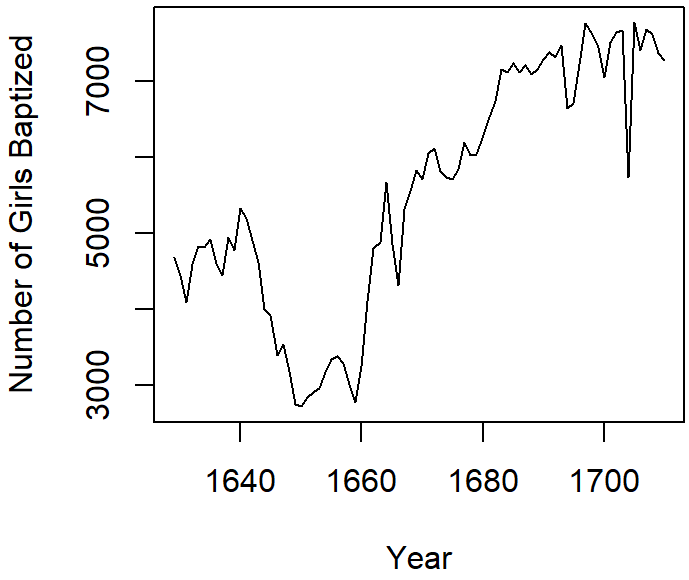
You can connect the points with a line by adding `type="l"` within `plot()`.

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb20-1)plot(arbuthnot$year, a
       rbuthnot$girls, type="l")
```



It's a good habit to label the axes of the plot.

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb21-1)plot(arbuthnot$year, a
       rbuthnot$girls, type="l", xlab="Year",
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb21-2)  ylab="Number of Girl
       s Baptized")
```



3.6 Further Exploration of Arbuthnot's Data

Now, suppose we are interested in the ratio of the number of boys to the number of girls baptized in 1629 To compute this, we could use the fact that R is really just a big calculator. We can type in mathematical expressions like

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb22-1)5218 / 4683
```

to see the boys/girls ratio in 1629. We could repeat this once for each year, but there is a faster way. If we take the ratio of the vector for baptisms for boys and girls, R will compute all ratios simultaneously.

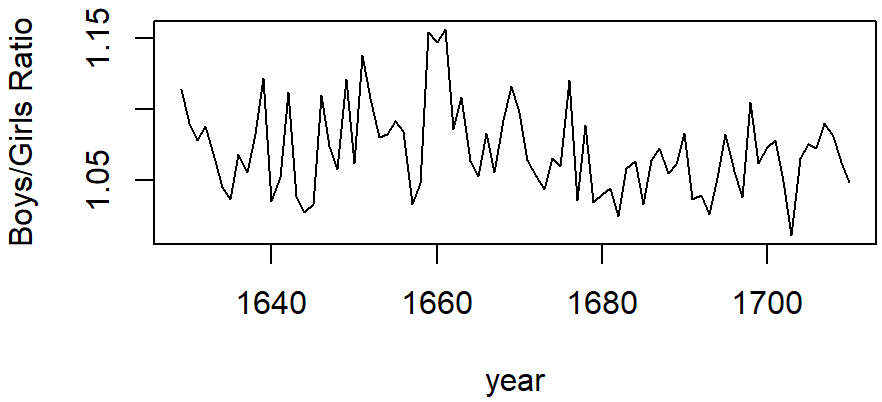
```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb23-1)arbuthnot$boys/arbuthn
       ot$girls
```

You can also use `with` to avoid repeatedly typing the name of the data frame. `with` instructs R to interpret everything else from within the data frame that you specify.

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb24-1)with(arbuthnot, boys/
       girls)
```

What you will see are 82 numbers, each one representing the sum we're after. Take a look at a few of them and verify that they are right. Therefore, we can make a plot of the total number of baptisms per year with the command

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb25-1)with(arbuthnot,
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb25-2)  plot(year, boys / gi
       rls, type="l", ylab="Boys/Girls Ratio")
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb25-3))
```



Similarly to how we computed the proportion of boys, we can compute the ratio of the number of boys to the number of girls baptized in 1629 with

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb26-1)5218 / (4683+5218)
```

or we can act on the complete vectors with the expression

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb27-1)with(arbuthnot, boys/
       (boys + girls))
```

Note that with R as with your calculator, you need to be conscious of the order of operations. Here, we want to divide the number of boys by the total number of newborns, so we have to use parentheses. Without them, R will first do the division, then the addition, giving you something that is not a proportion.

Now, we make a plot of the proportion of boys over time. What do you see?

```
(https://www.stat.uchicago.edu/~yibi/s220/labs/lab01.html#cb28-1)with(arbuthnot, plot(y
       ear, boys/(boys + girls), type = "l", ylab="Proportion of Boys"))
```

3.7 How to Export R Plots and R Outputs to a Word Document?

To export a plot, go to the [Plots] tab in the bottomright panel, click on [Export], and then select [Copy to Clipboard].

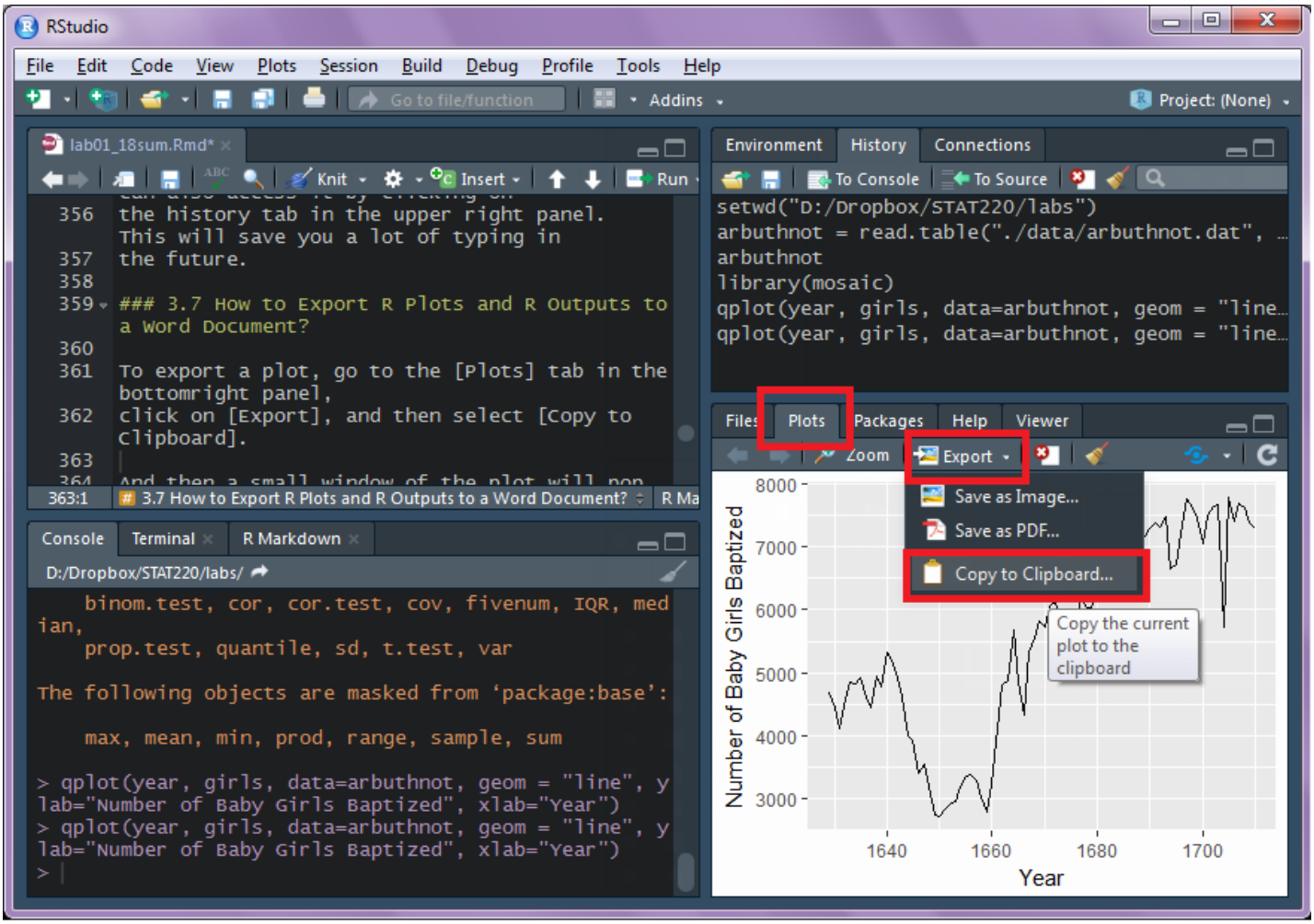


Figure 4

And then a small window of the plot will pop up. In the pop up window (you may adjust the size of the plot at this point), select the option [Copy as Metafile] and then click [Copy Plot] and then paste it directly into your Word document.

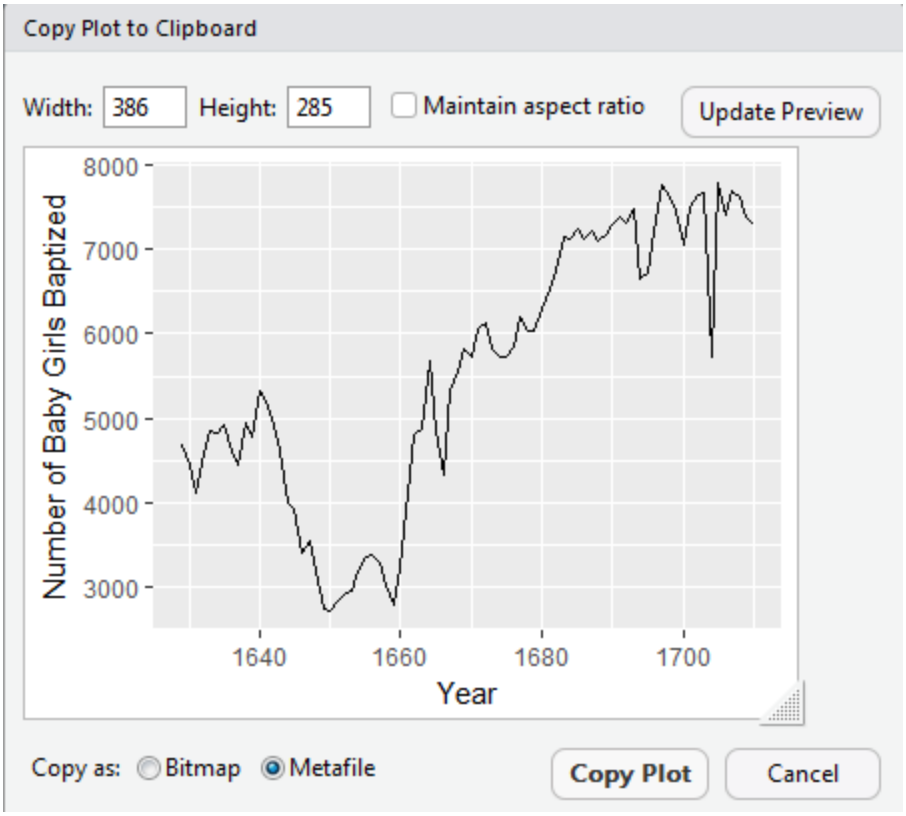


Figure 5

Many students do STAT220 homework by typing answers in a Word document, copy-and-pasting R outputs (including plots, numerical outputs, and possiblly some R codes) to where they are needed from R Studio, and then submitting the doc file. This way is simple. One downside of this method is that it can be bothersome to copy and paste back forth between RStudio and Word document when the R work is long. Another downside is that the copied numerical R outputs look ugly in Word document since they don't line up so well as they were in RStudio.

A more advanced way (and still simple) is to write the text answers and R codes in a **R markdown** file, and then RStudio can convert the R markdown file to a Word documents, with the R outputs and R plots inserted in the text. For R markdown tutorial, visit the page http://shiny.rstudio.com/articles/rmarkdown.html (http://shiny.rstudio.com/articles/rmarkdown.html) or ask CAs for a demonstration.

4. Ending

This seems like a fair bit for your first lab, so let's stop here. To exit RStudio you can click the *x* in the upper right corner of the whole window.
You will be prompted to save your workspace. If you click *save*, RStudio will save the history of your commands and all the objects in your workspace so that the next time you launch RStudio, you will see `arbuthnot` and you will have access to the commands you typed in your previous session. For now, click *save*, then start up RStudio again.

On Your Own

In the previous few pages, you recreated some of the displays and preliminary analysis of Arbuthnot's baptism data. Your assignment involves repeating these steps, but for present day birth records in the United States. The "present" data set can be downloaded at http://www.stat.uchicago.edu/~yibi/s220/labs/data/present.txt (https://www.stat.uchicago.edu/~yibi/s220/labs/data/present.txt)

- What years are included in this data set? What are the dimensions of the data frame and what are the variable or column names?
- Make a plot that displays the boy-to-girl ratio for every year in the data set. What do you see? Does Arbuthnot's observation about boys being born in greater proportion than girls hold up in the U.S.? Include the plot in your response.
- In what year did we see the most total number of births in the U.S.? You can refer to the help files or the R reference card http://cran.r-project.org/doc/contrib/Short-refcard.pdf (http://cran.r-project.org/doc/contrib/Short-refcard.pdf) to find helpful commands.