**Transcript: R Commander Basics**

In this video, we're going to talk through some of the basics of R Commander, including what we input into various panes, as well as how to import data. To launch R Commander, you first need to open R. The console looks like this. You might also have some text here about the version and the developers, but it should look like this, with a stop sign and a little arrow over where you can type.

To launch R Commander, we're going to type “library” and then the name of the package, which is capital “R” and then “cmdr”. It's very important that you type this in exactly as I have it. It's also available to copy and paste from the lab materials, but R does not like typos and that includes having things capitalized that shouldn't be, or vice versa. If you're getting weird error messages, it's probably because you haven't typed this in correctly.

But once we type that in, we can press enter. You might get some red text here that says that it's loading packages. That's all fine. It might take a minute or two depending on the speed of your computer. But eventually you're going to see a window that looks like this.

This is actually R Commander. We're not going to use this R window anymore, but we're going to do all of our work here in R Commander. As you can see there are some options along the top. We'll be going through some of those as we progress in the lab in terms of different analysis we can do.

But right now, we're just going to look kind of at the basic window pane. So first of all, we've got up here, R script and R Markdown, and these are where the actual codes that that your analysis uses are going to be written. If you remember from the previous video, we said that you're actually not going to be writing any code in R Commander. Instead, what happens is the buttons that you push up here—so let's say we're going to select, you know, various statistical tests— are going to write the code for you based on the options that you select. You're actually not really going to have to do anything with this R script section up here, but it is maybe helpful if you're interested in programming or if you plan to use R in the future, such as in 372 or 373, just to be aware that it's there.

The thing that actually is going to matter to us here is this output. So anytime we conduct an analysis, all of our output is going to be displayed here, so that's the results of our analysis or anything else that we're doing. This is where we're going to be spending the majority of our time looking.

One last thing here. You might get some error messages sometimes. I've written some code that's produced some just so you can see what they look like. You may end up looking at these if you if you make a mistake or if you do something else that results in an error message.

So that's kind of the basics of what happens in each section of R Commander. And of course, as we go through our various specific analysis, we're going to talk in much more detail about what we're looking for in each.

The next thing we're going to do, though, is talk about how we actually get data into R Commander. Because of course, when we're conducting analysis, we have to have our data set inputted.

So, the first way that we can get some data within R Commander is there are some datasets that come kind of pre-loaded, and so these are often datasets that are really good examples, have good documentation for what the variables are, and that we can use to learn how to do statistics. The way that we access those is we can go to “Data” here, “Data in packages,” and if we want to see, you know, what datasets are available to us, we can select “List datasets in package”. This is going to bring up a whole list of datasets, and this is in the package “carData”. You can see just from A to D here. We've got a whole bunch, and it'll tell you the name of the data set and a very brief description of what it is. If you want more detail about any of these datasets you can go through the documentation for them, which is a word that we use to describe more details about things in R, and that will be linked in the lab materials below this video. But if you're looking for a particular data set that you don't remember what it's called, or you're just curious to you know what this “Pottery” data set is, you can see here that it's chemical compositions of pottery.

That sounds really interesting, so let's go and see what that actually looks like. We can close out our list here because that's just the names. Then we can go to “Data”, “Data in packages”, and then “Read data set from an attached package...”. This is where we can actually find those datasets that were listed in the previous list. So, if you remember it was in carData, we double click it. We're going to see again this whole list of all the possible datasets we could use. So, let's try to find this “Pottery” one. Here it is. You could also just type it in if you know the name. Again, be careful of typos. But once we select that we can click “OK”.

You can see here an example of what I was saying earlier about the R script. So, if you were going to actually write our code, which again you do not need to do for this class, but if you're curious, you can see here this is what the code actually looks like. So, we've read the data set in; we can see here this note that says that the data set “Pottery” has 26 rows and six columns. So that's all along good, but we might want to see what actually is the data set. What does it look like? What are the variables? To do that we can click over here at “view data set”. It loads it all the way over here on the right, and we can drag it back in to be more central. We can see here what our six variables are. “Site” and these look like different chemical elements: Aluminium, iron, magnesium, calcium, and sodium, I believe. This is really drawing on my high school chemistry knowledge and we've got you know values for each of these variables. So, if we had a bigger data set, we could scroll around to see more columns or more rows, but it looks like it's a pretty small one. This is what our data set looks like. And if we were going to do some analysis on it, we could then use this data set to conduct those analysis. Hopefully this gives you a good sense of how to use the built-in datasets in R Commander. We will be doing that for a number of the labs this term and of course, if you're interested, you can always look at all of the datasets. There's a ton of interesting stuff going on there.

The other thing we need to do though is we need to learn how to import data that's not built into R Commander. So, if you were doing real research, of course your data would not be one of the datasets that's already available in R Commander. It would be data that you get say from Qualtrics if you're doing a survey or from an in-person lab study that you get from the research assistants who are running the lab. And in that case, you would have your data typically as an Excel file or CSV. Some sort of file on your computer that you then need to read into R Commander.

To do that, we're going to go again to “Data”. But this time, we're going to go to “Import data”, so that means that we're bringing in data from outside of R Commander. We're going to import it into R Commander. There's a whole bunch of different ways that you can import data. The one that we're going to be using in this class though, is “from text file, clipboard or URL...”. So, if we click on that. It's going to bring up this window. We can change the name of our data set if we want, so I might call this, you know, “Data1”. You could call it anything you like.

We've got some other options here. We don't really need to look at any of these except for one. One thing that might be useful, we're not going to use it in this particular video, but it’s this “Missing data indicator”. Often when you have a data set, you know that some data are missing. It could be because the participant left the question blank, equipment could malfunction, or anything like that. We need to have some way of indicating that these data are missing. Oftentimes we'll use “NA”, or not available. Sometimes people will also use something like 9999, or something else. And in that case, it's really important that we know which values are missing so that we're not, for example, not saying that you know 9999 is a real value. You can imagine if we asked people for their age and some people didn't answer the question, we wouldn't want to incorrectly in our results assume that the person was 9999 years old. For this class generally this NA default setting is going to work well, but it is important that you know what that value is and that you can change it if you need to.

We're going to leave all of the rest of these the same, except for this “Field Separator”. The type of file that we have this case [and that we're going to be using throughout the term] is called a CSV or comma separated value, which means that it's got commas in between the different sets of data. It isn't really important that you understand how these files work, but we are going to switch this to comma just to make sure that it reads in correctly.

From there I'm going to click “OK”. And that's going to bring up this window where I can select the file that I want to use. Now I've already set it up so that this is in the folder of the data set that I want to import, but you're probably going to have to navigate around your computer a little bit to wherever you've stored the file—whether that's in your downloads on your desktop, in documents. You will find it the same way that you would find a file any other time. We can click on this and click “Open”. And it's going to do the same thing.

So again, we can see the code here. We don't need it, but it's there if you're interested. And if we want to view the data set, we can click on “View data set”. In this case we've got a different data set. This isn't about pottery. It's actually about people who've experienced a coma. We can see here what we've got various different variables about their age, their sex, their IQ, and some stuff about their coma.

So that's how you would import a data set from a CSV file. And from there it's imported in R. It's there just the same whether it was built into a package or whether you read it in from a file.

OK, so now that we have our data set, there's one other thing that we want to show you, which is that R Commander sometimes will use scientific notation. So scientific notation is a convention that's used in the sciences and in statistics and other fields to represent either very large or very small numbers. If you need a review on scientific notation, I've linked a YouTube video in the lab materials that's really great as a quick refresher. But it is important to know that R Commander often uses scientific notation and we will be encountering it in the output this term.

I'm going to demonstrate this to you in R. This is really just for demonstration purposes. You don't need to follow along or anything. If I put in a very, very small number, let's say .000… [I don't even know how many zeros] and then a three, what we're going to get out is this “3 e – 15”. So instead of writing out all of these zeros, what this means is 3 times 10 to the negative 15th power, and again you can review the video on scientific notation if you're not quite up to speed on what that means, but this is how R represents 3 \* 10 to the negative 15th power. And if we had, you know more decimals after the three, we would see 3.16 in this case times 10 to the negative 13.

We're going to see numbers like this a fair bit at this term, so it's just important that you don't panic when you see them. Seeing a letter in the middle of a number can be confusing. Just remember it's scientific notation and you can go back and review what that means if you if you need it.

One thing we're going to see definitely less often (I don't think we'll encounter it this term) is the flip side where we have very, very large numbers. So, let's say we've got 316, followed by a bunch of zeros. We're going to see 3.16 \* 10 to the positive 16 or 10 to the 16th. Again, this is, you know, a scientific notation, which is something we're going to run into in R Commander so it's important that you are familiar with it.

So those are the basics of R Commander. Hopefully that's helpful for you in getting started, as well as throughout the term. If you forget how to do something, you can come back and reference this video or the transcript.