**Data Analysis in R: What R can do for you?**

Connecting to the Science server to use R:

In this class or when you are using R on campus, you can do so by connecting to the Science Department’s computer server. This way you do not have to worry about downloading and updating R and many of the packages we need will already be available to you.

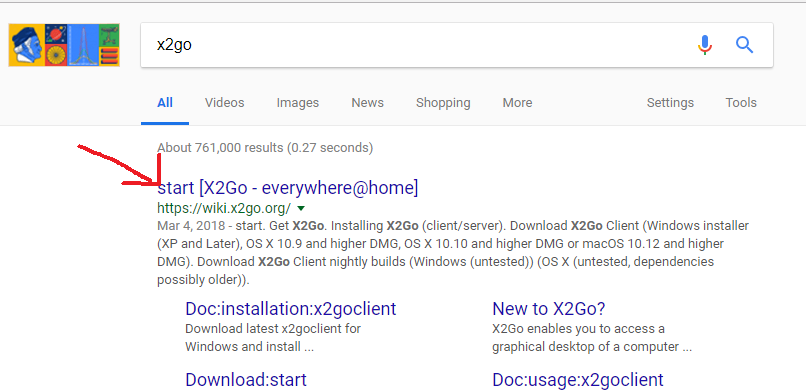
If you would like to access R from your home computer, it is a free download. I recommend R Studio, which has a graphical user interface:

<https://www.rstudio.com/>

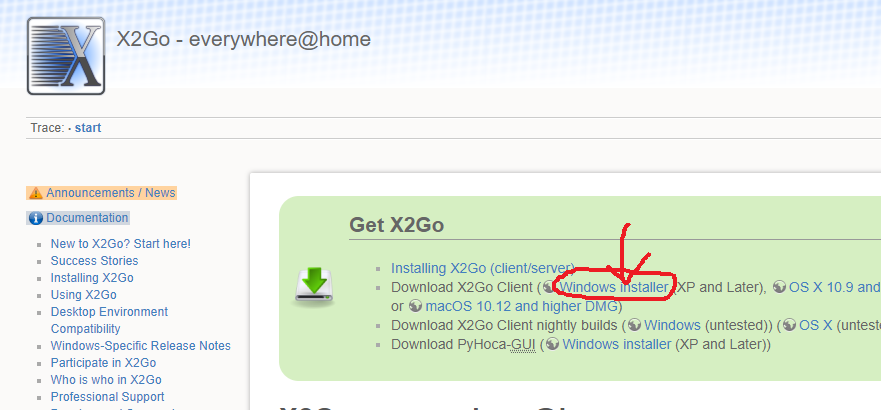
Downloading and installing x2go

For this class, we will use R on our science server. To do this, we must download a program that allows us to connect to the server. With this program, we will actually be running R remotely on the server. Let’s download this program and “enter the matrix”.

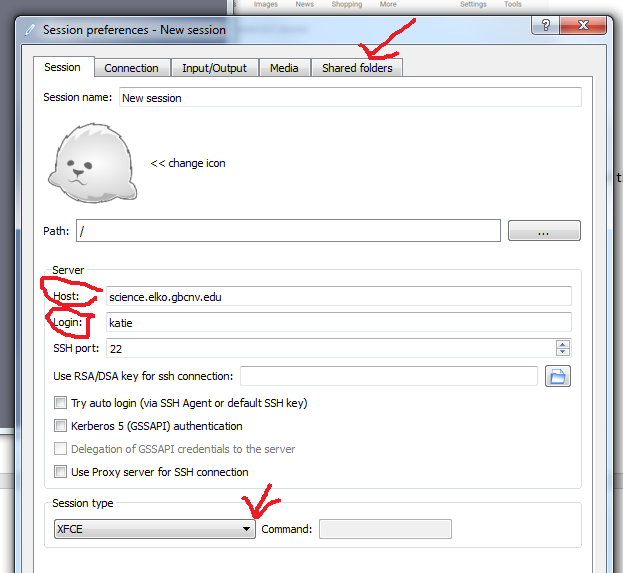
1. Type x2go into google and click on the first hit.



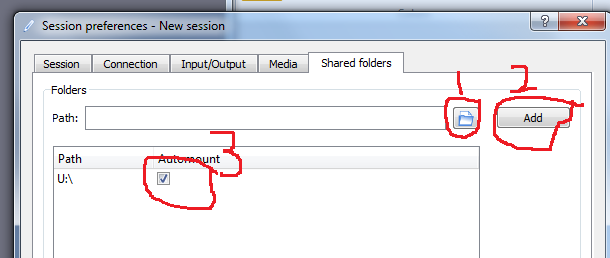
1. Click on the link for downloading the X2Go Client Windows Installer. Download the file and open the “.exe” file. Click “Run” and accept all the defaults.



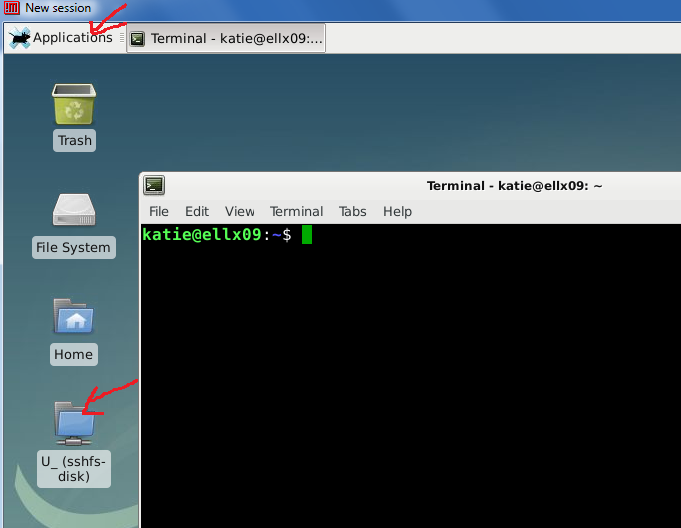
1. After the install finishes, click on the “X2Go Client” icon on your Desktop. Click “Allow Access” and “OK” on all the pop ups. You should have a window that looks like this. Enter:
   1. Host: science.elko.gbcnv.edu
   2. Login: (your name)
   3. Session: change dropdown to XFCE.
   4. Then click on “Shared Folder” on the top.



1. Under the Shared Folder tab, click on the little folder icon and navigate to your “U” drive. This is the drive that has your student ID number. This will allow us to save and share files between the lab computer and the server. Once you find your drive, click “Add” and check the “Automount” box. Then click “OK” at the bottom of the window.



1. Now, click on the little animal (baby seal?) on the top right box. It will bring up a login box that should already have your name. Enter your assigned password here and click OK. Click “Yes”, “Allow Access” and “OK” on the pop-ups. Now a window should open up and have another Desktop and a matrix-like window that says “Terminal” (not as bad as it sounds). You are connected to the server and running Linux! If your shared folder mounted correctly, you should see it on your Desktop.



Running R on the Science Server

Now we are connected to the server and ready to start running R. There are two ways you can run R, either:

1. Click on R under the applications window: Applications>Graphics>R
2. Or open the Terminal (computer screen icon at bottom of window) and type “R”.

General Operations in R

*R as a clumsy calculator*

R is not only used as a statistical package. You can also use it as a somewhat clumsy calculator. This is useful when we use R to run simulations (i.e. generate data). Try out some of the operations below to see how R is used as a calculator.

1. Type 2 + 2 into R. And hit enter. When you do so, R should return the answer to you.
2. Type 2\*10 into R. Does it return the answer?
3. Type 100/10 into R. See how R does at long division.

In addition to being a basic calculator, R is valuable because you can assign variables. Test this out below. Enter the following commands one at a time.

1. x<-2
2. y<-2
3. x +y
4. What did this do? What do the “<-“ symbols mean to R?

Now that we have the basics down, Let’s see how R can be used as a data simulator.

1. We can use the rnorm command to generate a normal distribution of data. Before we do this, let’s make sure we know how this command works. Type:

?rnorm

When you are finished looking, type “q” to exit the help screen.

1. Now, let’s generate a normal distribution with 20 observations.

rnorm(20)

1. This shows you some of the useful for R to simulate data. But if these numbers are just displayed on the screen, this is not too useful to us. Using what we learned earlier, we can get R to save a hole distribution. Try this:

data<-rnorm(20)

data

1. Now that the data is saved in R, we can do anything we want with it. Try the following:

mean(data)

sd(data)

min(data)

max(data)

range(data)

**\*\*\*When you find commands that are useful and you are likely to use again, it is a good idea to copy them and save them in a text document. To do this on the server, go to Application>Accessories>Text Editor. You can save your file as a “.R” and Text Editor will know it is an R file and help you with formatting, too (i.e., tell you if you are missing parentheses).\*\*\***

Plotting Data in R

Now that we know the basics, we can practice using R as a graphics program.

*Histograms*

If you have a large dataset of continuous values, one of the best ways to visualize it is by building a histogram. For example, imagine you were studying tree age and had age values for 20 trees. A histogram will help you quickly visualize the patterns in this data.

1. Let’s first get to know the command:

?hist

1. There are a lot of ways you can tweak this command. Not all of these arguments are necessary to get the command to run. In general, I always use:

hist(x, xlab=”Name of x variable”, main=”Name of chart”)

1. Let’s test this out, using the normal distribution we already generated.

hist(data, xlab=”X Value”, main=”My Random Data”)

1. Does it look like a normal distribution to you?

*Scatterplots*

Suppose you collected more than one variable and you would like to determine whether or not they are correlated. For example, you are studying lizards and you would like to know if their length is correlated with their mass. Before running statistics, it is a good idea to visualize this data using a scatterplot. Let’s explore this.

1. The function that allows us to do this is called plot. Using what you already know, explore the help menu for this function and see if you can determine the minimum arguments you will need to run it.
2. Let’s generate some x and y data and run plot. See if you can do this on your own, given what you’ve learned.
3. What commands did you use? How would you describe your plot?

Later, we will explore how to add trendlines to the plot.

*Boxplots*

Suppose you want to compare the average of two different groups. Before doing statistics, it is useful to visualize the data and see if they appear to have differences. One way to do this is using a boxplot. This type of graph displays the median, range, and quantiles data (the upper and lower values that describe most of the data) of the data.

1. Using what you already know, investigate the boxplot command. What arguments does it need? How is its formula different from the other plot functions that we’ve looked at?
2. To test boxplot, we can use one of the example datasets in R called InsectSprays. There are many datasets available in R and they are useful for practicing. Type:

?InsectSprays

1. To see what other practice datasets are available in R, load the MASS package into the library and type the data command:

library(MASS)

data()

1. Now that we know more about this dataset and the boxplot command, let’s create a boxplot of this data. Type the following command on all one line. Before hitting enter, go through each part of the command and make sure you understand what it is doing. If you are unsure, ask!

boxplot(count ~ spray, data=InsectSprays, xlab=”Type of Spray”, ylab=”Insect count”, main = Insect Spray Data”)

Can you tell just by looking at this plot if some sprays were more effective?

*Barplots*

Barplots are similar to histograms but are useful for examining several different variables at once. Let’s explore how to use this command in general, and then we will use it to plot our own data.

1. Let’s get familiar with the barplot function:

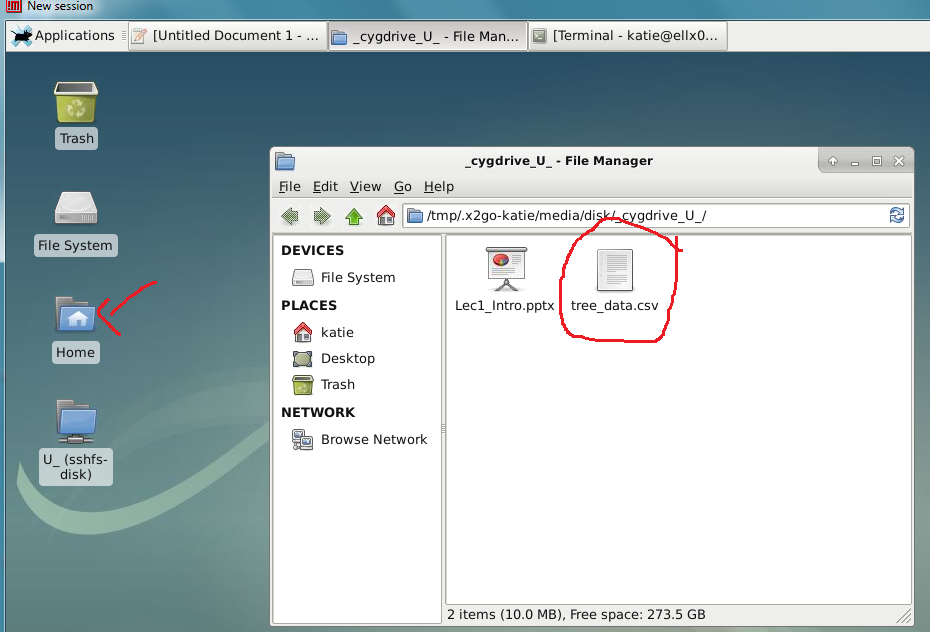
?barplot

1. Use barplot to plot your randomly generated dataset or one of R’s example datasets. How does the barplot compare to the histogram?

Importing Your Data to R

Now that we are familiar with how to use R in general, let’s bring our data into R and start to explore our results.

1. To get your data in the server, go back to your Windows desktop and copy your .csv to the U drive, if you have not already saved it there. Now, on your server window, you should be able to click on the U drive folder on the Desktop and find your file. Now, drag this file from the U drive to the Home folder. This makes it easier for R to find it. Now we can access that data on the server and load it into R.



1. Type the following command into R. Again, pay attention to what each part of the command is doing and ask if you are not sure.

inverts<-read.csv(“inverts\_totals.csv”)

1. Now that we’ve imported the data, we can type in its name to see what it looks like:

inverts

1. Now let’s see if we can make a fancy barplot. Take a look at the following command. It is a tricky one! Count the “()“ and “[]”. If you miss just one, R will be stubborn and refuse to run your data. See if you can figure out what each means.

barplot(t(inverts[c('Count\_Riffle','Count\_Pool')]), beside=T,names.arg=inverts$Order, xlab="Invertebrate Order", ylab="Count", legend.text=c('Riffle','Pool'))

1. What do you think? Does diversity differ between Riffles and Pools?
2. Once you have your beautiful plot, don’t forget to save it. The command below will save this plot as a jpg in your Home directory. You can drag it to your U drive to access it from any campus computer.

dev.copy(jpeg, 'inverts\_plot.jpg', height=700, width=1200)

dev.off()

Bonus Challenge: Use R to calculate a Diversity Index for Riffles and Pools

Based on our plot, we already have a good idea of how diversity differs between Riffles and Pools. To quantify this difference, Ecologists often use what is called a diversity index. The most common such index is the Shannon-Weiner Index which measures both the number of species and their evenness. Take a moment to read about this index in your Ecology textbook (pg. 346) or online.

1. Think about how you would calculate the Shannon-Weiner Index by hand or using Excel. This is a good exercise for learning the index, but might get tedious.
2. Don’t worry! You can calculate a diversity index in one step using R! To do this, we will use a package called “vegan”. Packages are R programs that other people have created and made freely available to everyone. This is one of the many reasons scientists love R! We have already downloaded vegan on the server. To load it, type:

library(vegan)

1. Inside the vegan package, there is a function called diversity that can calculate the Shannon-Weiner Index for us. Explore this command:

?diversity

1. Now run the command for Riffles and Pools

diversity(inverts$Count\_Riffle)

diversity(inverts$Count\_Pool)

1. How do the numbers compare for Riffles and Pools? Remember, this is a log scale.

Congratulations! You have entered the matrix and communicated with R! Now you can exit the matrix and go back to the real world ☺

1. On the server window, click on your name on the top right and click “Log Out”.

