**Biostatistics**

**Lab 12**

*Tasks*

Regression in R, Plotting Regression Lines in ggplot2

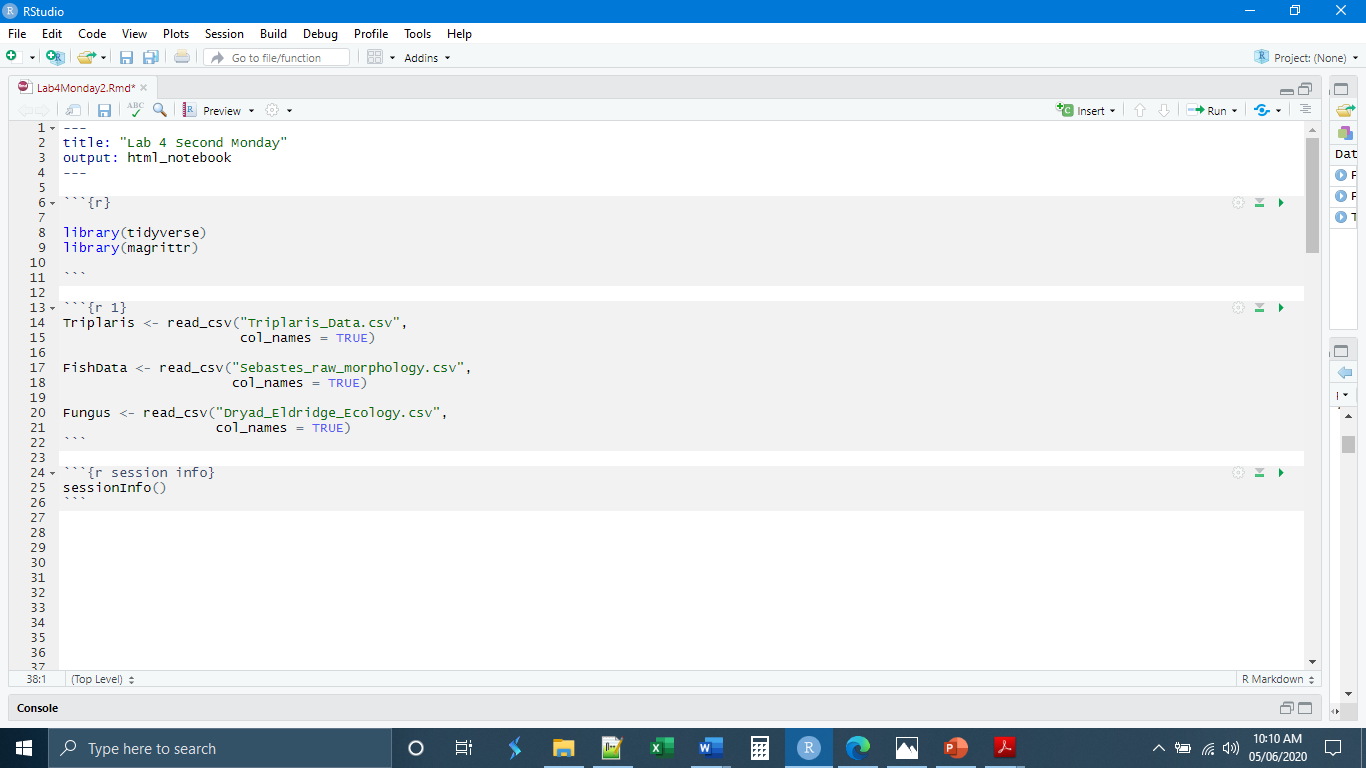
*Introduction*

Today’s lab will be reasonably short. The goal is simply to reinforce some of the major concepts associated with regression as well as introduce how to perform and plot a regression line in R. You already have the foundational skills necessary to do this, it is just a matter of learning a few other commands.

*Set-Up and Data*

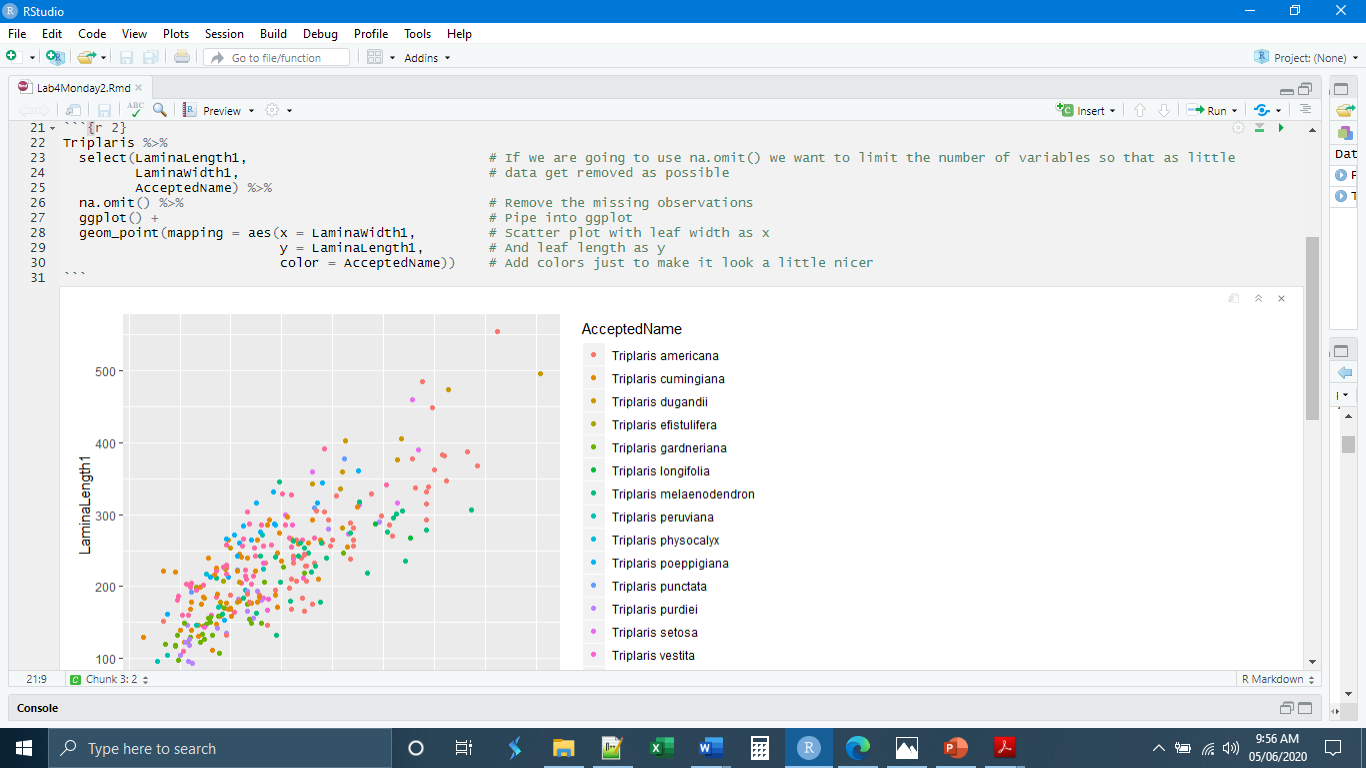
Today will be considered the last part of Lab 4. If you have not done so already, download from Blackboard all of the data files necessary for the lab this week and save them in the Lab4 folder you made last Monday. The files that will be most important for today are the Triplars\_Data.csv, the Sebastes\_raw\_morphology.csv, and the Dryad\_Eldridge\_Ecology.csv data.

Start R Studio and open a new R Notebook file. Save this file in the Lab4 folder where the data files are. Delete the text that comes in the notebook template, and use the first chunk to load libraries. It is good practice to also present your session info, make a last chunk to do this. Make a chunk 1 to read in the data and save them as new data objects.



*Regression Relationships geom\_point()*

Remember that, while we can do a regression and correlation analysis on any pair of continuous variables, there may not always be a relationship. The value or magnitude of one variable may not change as the value or magnitude of another variable does. In general, when two variables are not related to one another, this is manifested as a flat line in a scatter plot. As a result, if we are interested in doing a regression analysis, it is often best to begin by simply potting the two variables against each other in a scatter plot, to make sure that there is some relationship present. We have already learned how to make a scatter plot in ggplot. For now, let’s make a scatter plot of the leaf lengths and widths in the Triplaris data, removing any missing data.



*Linear Regression*

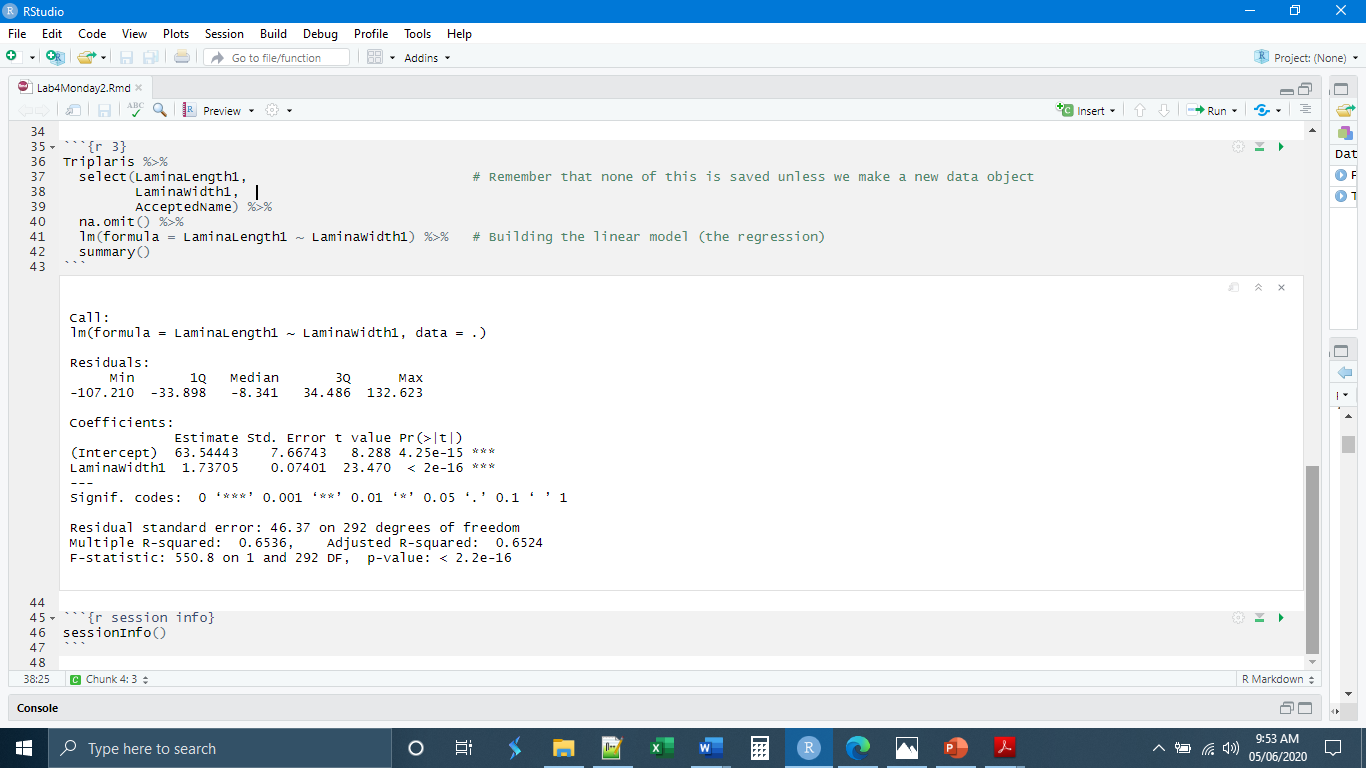
We can see that our data do have a relationship. The leaf length does vary with the leaf width. Moreover, it appears that the relationship is a linear one. The change in leaf length is a constant proportion of the change in leaf width. It is alright for us to proceed with constructing the linear regression line. The general syntax for this is

DataFrame %>%

lm(formula = y ~ x) %>%

summary()

It is important to include the summary() at the end in order to get all of the information such at the r2 value and the significance tests for the constants.



*Plotting Linear Regression*

Now we can turn to plotting the line on our scatter plot. This will require us to introduce a new geom, geom\_smooth(). geom\_smooth() is a multifaceted geom. It can be layered over a scatter plot or used on its own. It requires its own aes() because you must define the relationship between the variables in each line. The basic syntax is

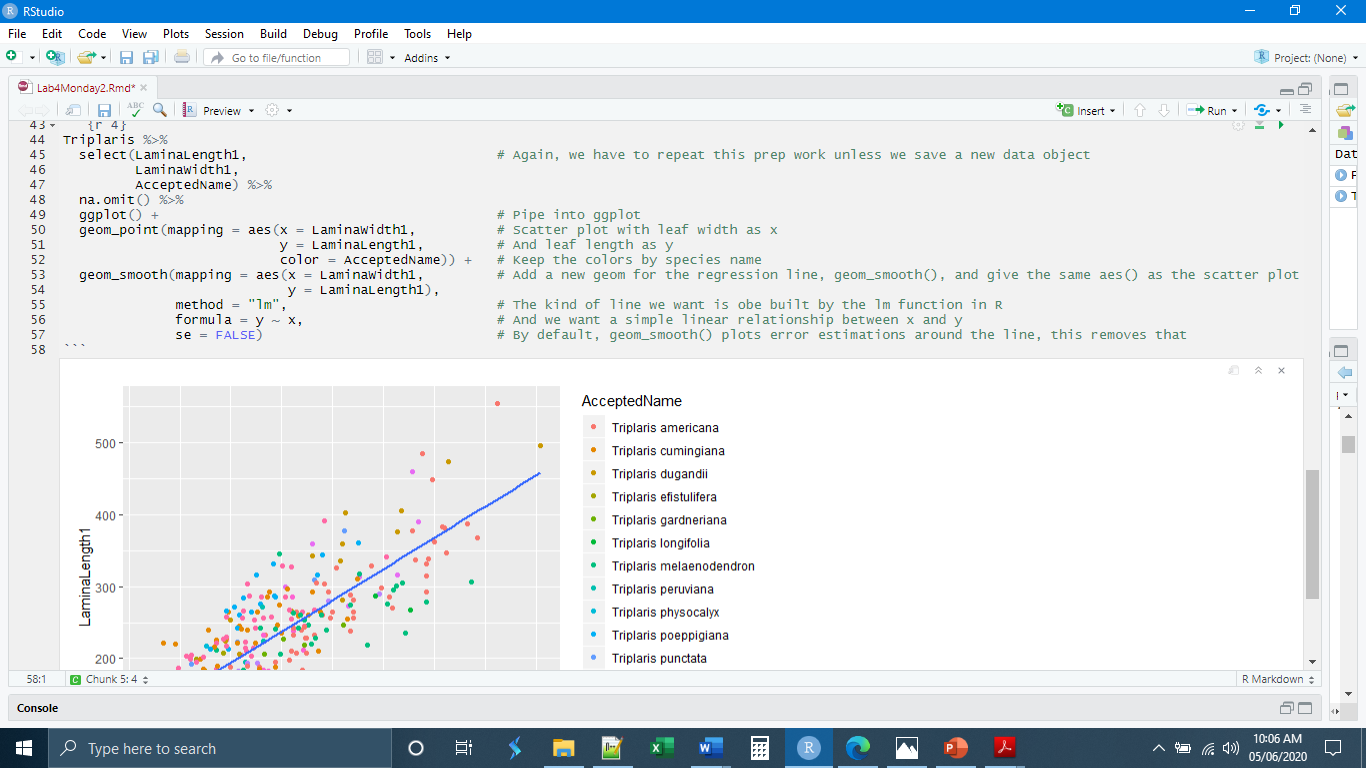
geom\_smooth(mapping = aes(x = xVariableName,

y = yVariableName,

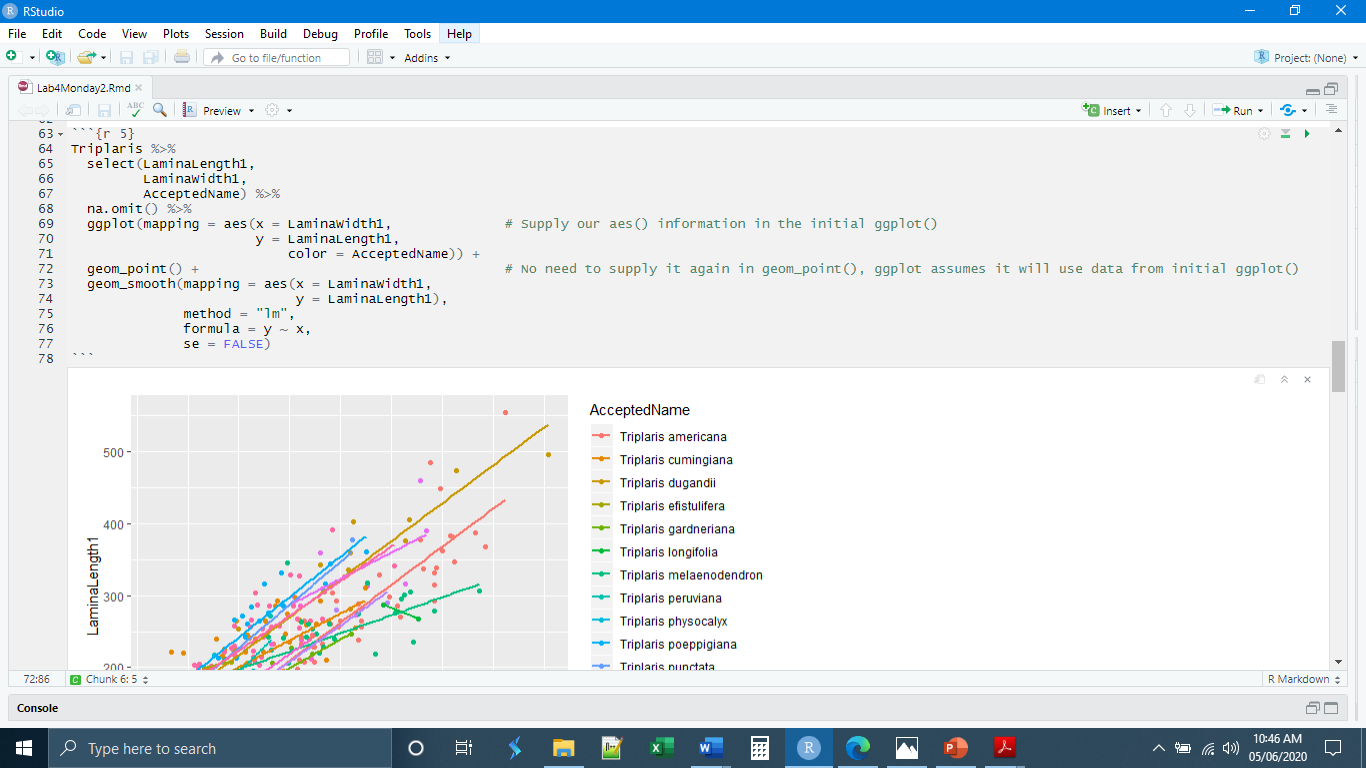
method = “lm”, # We want the same lm() function

formula = y ~ x, # For a linear relationship

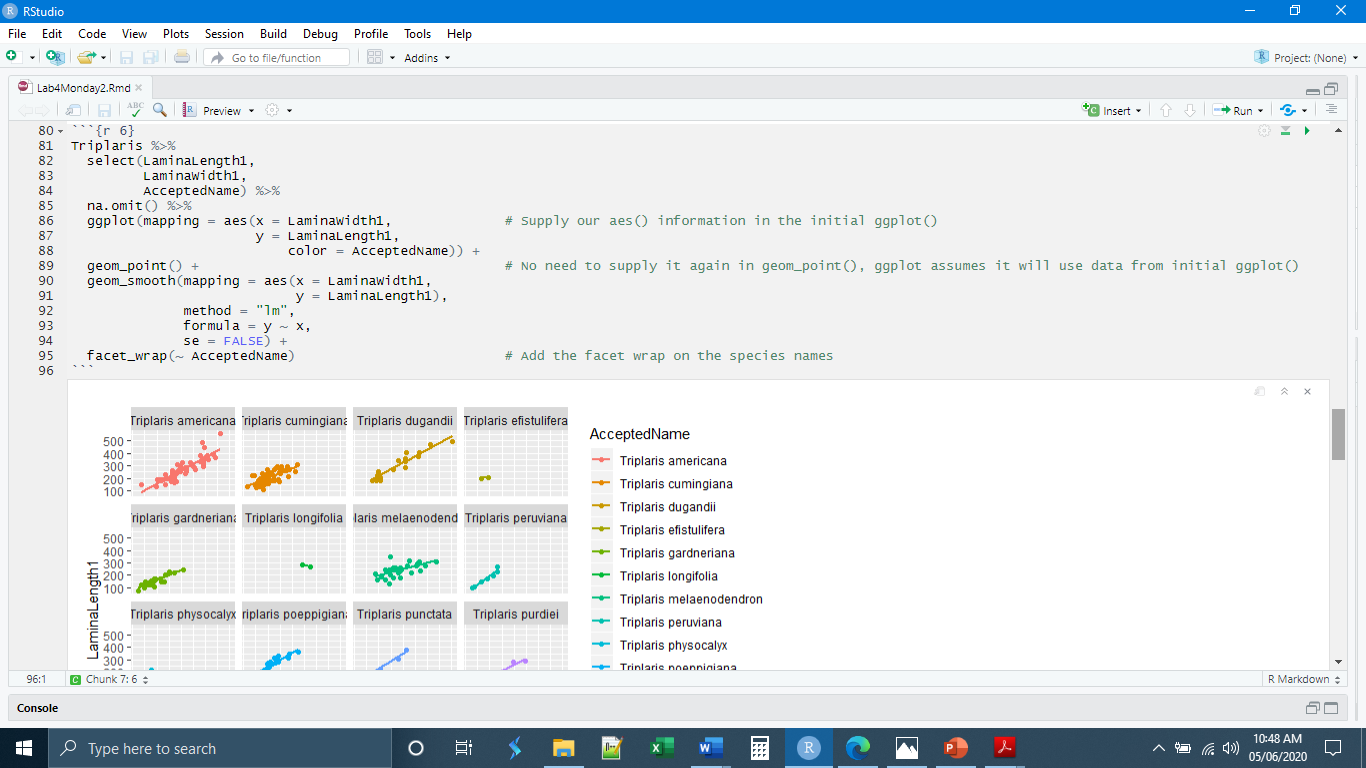
se = FALSE) # Prevents plotting of error shading



As a general rule, we have not been supplying the aes() information inside the initial ggplot() call. Supplying aes() information inside of the initial ggplot() call gives you much less flexibility because it means that ggplot will assume that everything you do after that point will be using those aes() data. If you supply the aes() information in each geom specifically, you can actually combine different variables in the same plot. For example, we could plot a regression of one set of variables on top of a scatter plot of a different set of variables in order to compare the relationships. Or we could use geom\_smooth() twice to plot lines for two different sets of variables, to compare the relationships. Yet, there are some instances where it is better to supply the aes() information within the initial ggplot() call. This is one of those cases. Here, geom\_smooth() will recognize that we have used a grouping variable (color = AcceptedName) and plot lines for each of our species separately.

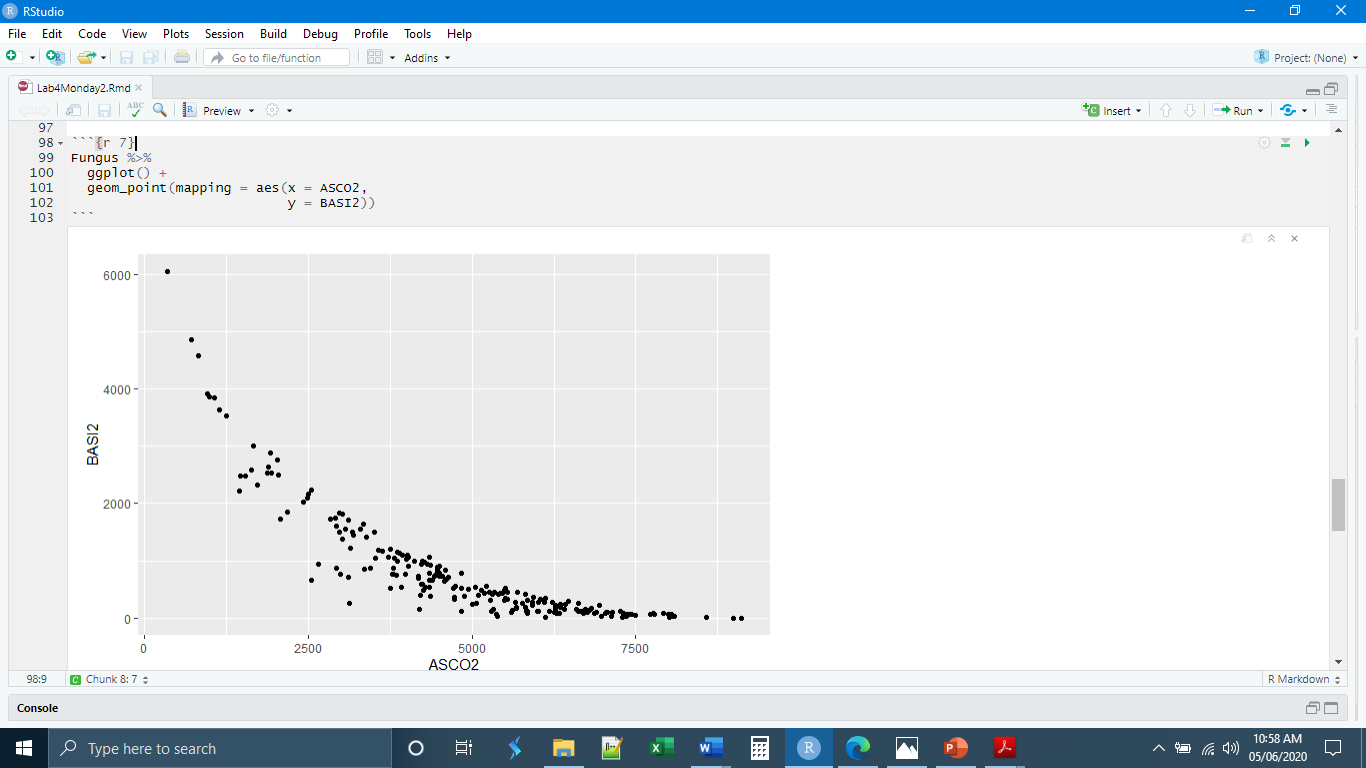


This is a bit chaotic, but if we add a facet\_wrap() it can look pretty nice

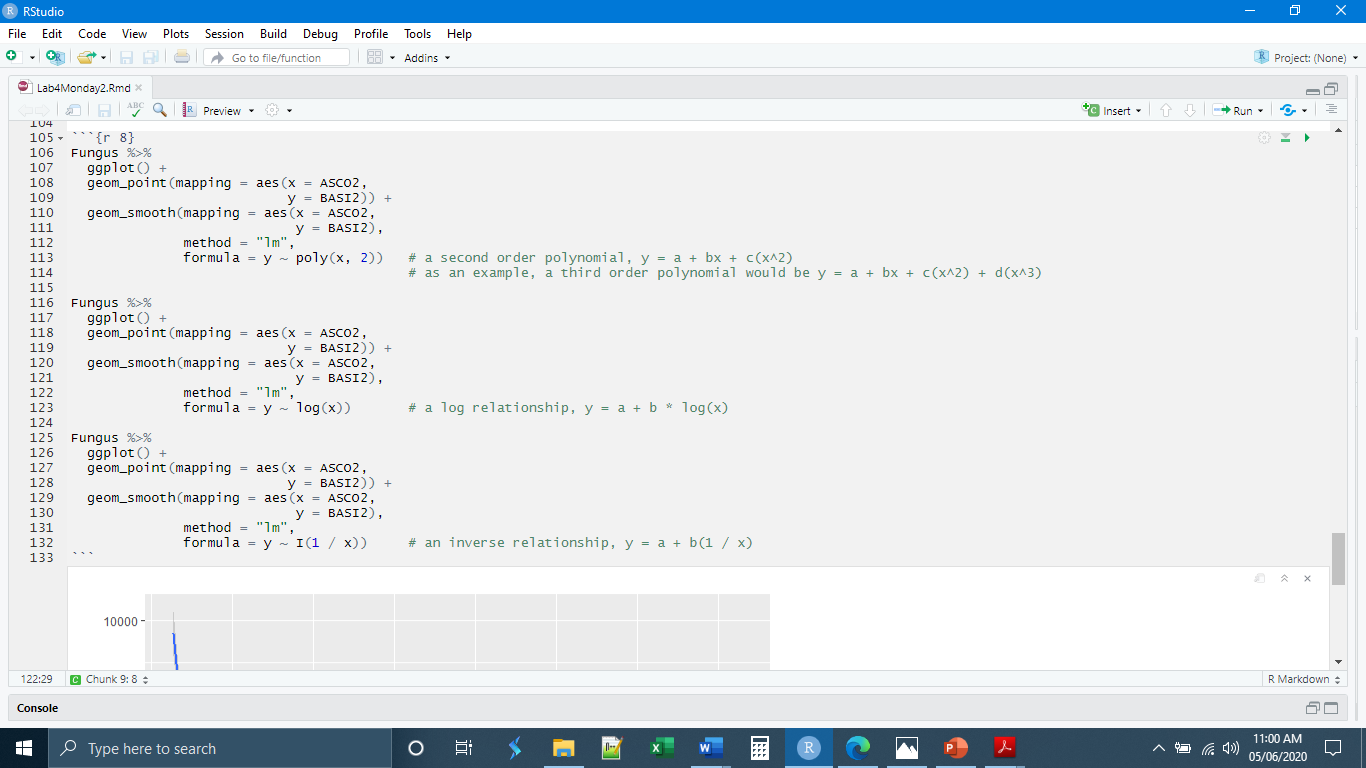


*Fitting Other Regression Lines*

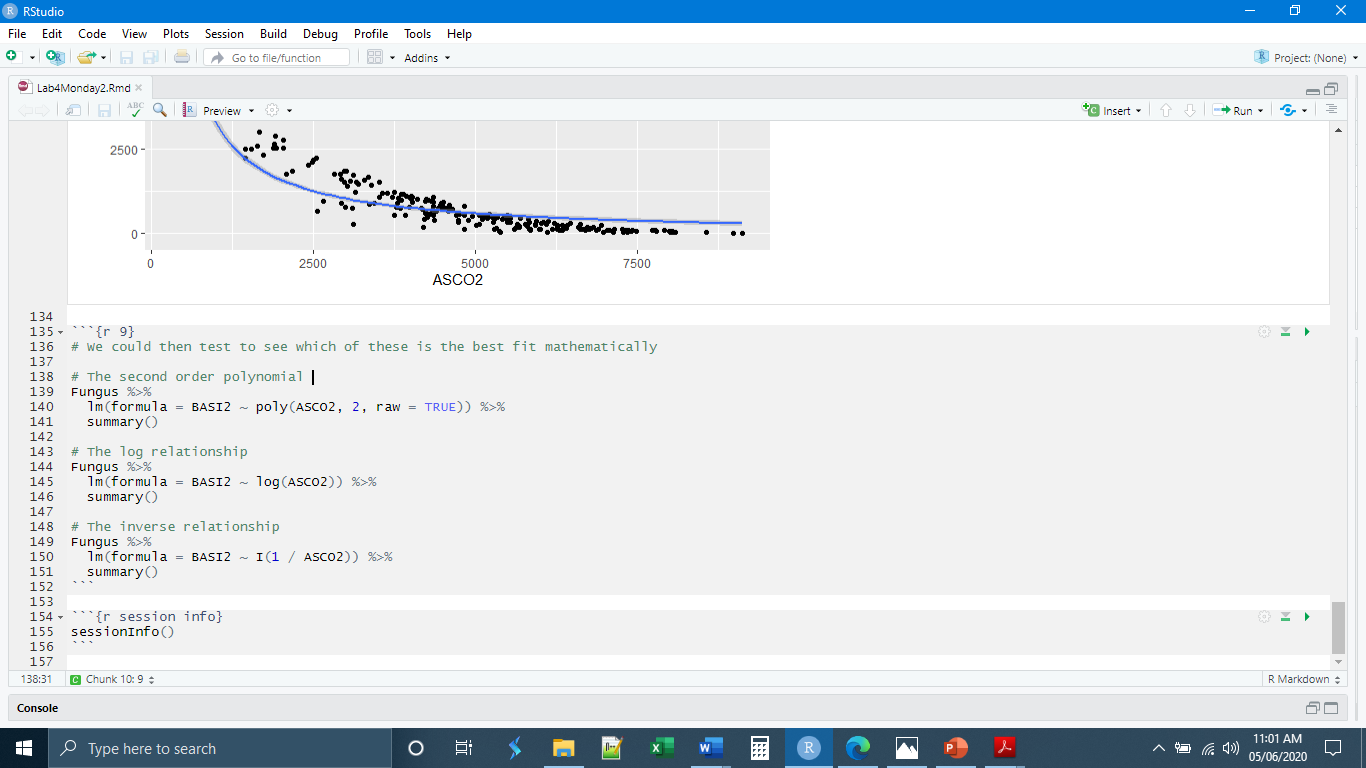
Both lm() and geom\_smooth() are capable of producing lines where the variables are related in a non-linear fashion. We have not covered this in lecture so I will only touch on it briefly here. If you recall from the Fungus data, we saw the relationship between the ascomycete abundance and the basidiomycete abundance appeared to be an inverse one.



Even though this looks both biologically and mathematically like an inverse relationship, for the sake of illustration, we can use geom\_smooth() to plot lines from a few different kinds of relationships.



Further, we could then use lm() to model and test those same relationships.



*Independent Exercises*

**1.** Make a new chunk called “chunk 7” and write code to

- Read in the Sebastes\_raw\_morphology.csv data and save it as a new data object

- Out of this data object select the variables TL (fish length) and PECTLEN (pectoral fin length), remove any missing observations, and use lm() to find the linear regression formula.

(Hint: the whole second part can be done in one piped expression)

**2.** Make a new chunk called “chunk 8” and, using the Sebastes data object created in chunk 7, write code to

- Select the variables TL (fish length) and PECTLEN (pectoral fin length)

- Remove any missing observations

- Use geom\_point() to make a scatter plot of TL (x axis) and PECTLEN (y axis)

- Use geom\_smooth() to add a linear regression line

(Hint: all of this will be done in a single piped expression)