**Biostatistics**

**Lab 9**

*Tasks*

More ggplot2 (geom\_jitter, violin, etc.)

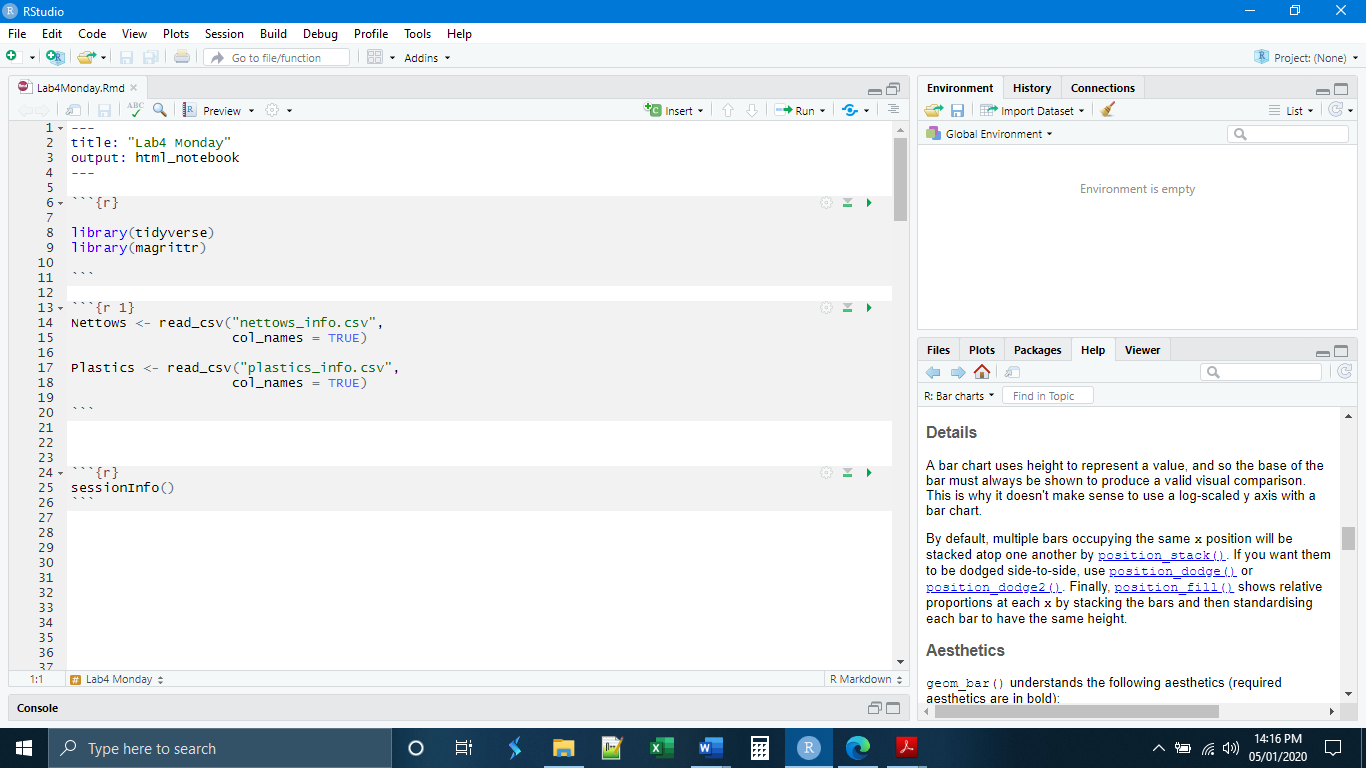
*Introduction*

Today will be the last lab devoted entirely to ggplot. The ability to convey complex information quickly, accurately, and intelligently is an important skill, which is why we are spending so much time on it. We have learned some of the basic plots already, especially those directly related to summary statistics (histograms and boxplots), but also scatter plots. The goal for today is simply to introduce a few other kinds of plots so that you have them in your back pocket.

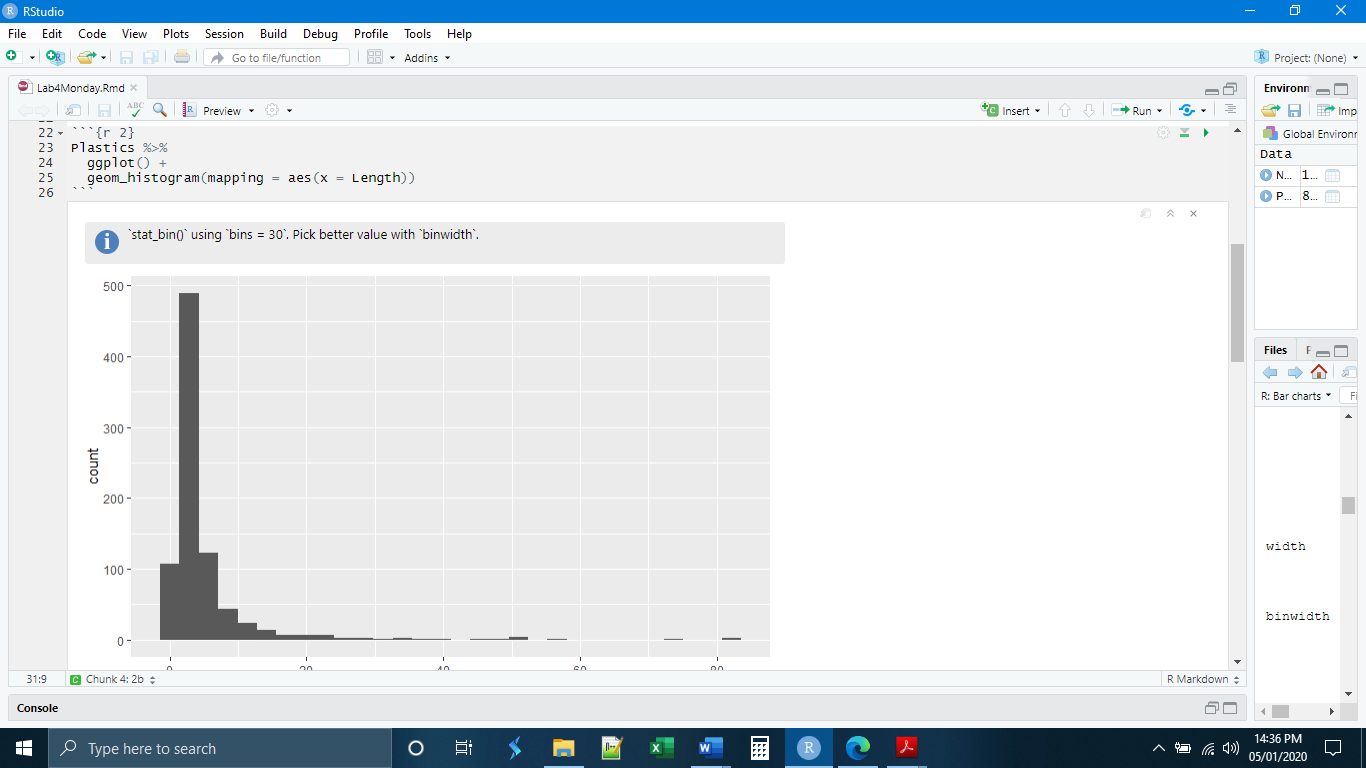
*Set-Up and Data*

If you have not done so already, download from Blackboard all of the data files necessary for the lab and save them in the folder for today. The files that will be most important for today are the plastics\_info.csv and nettows\_info.csv data.

Start R Studio and open a new R Notebook file. Save this file in today’s 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 to read in the data.



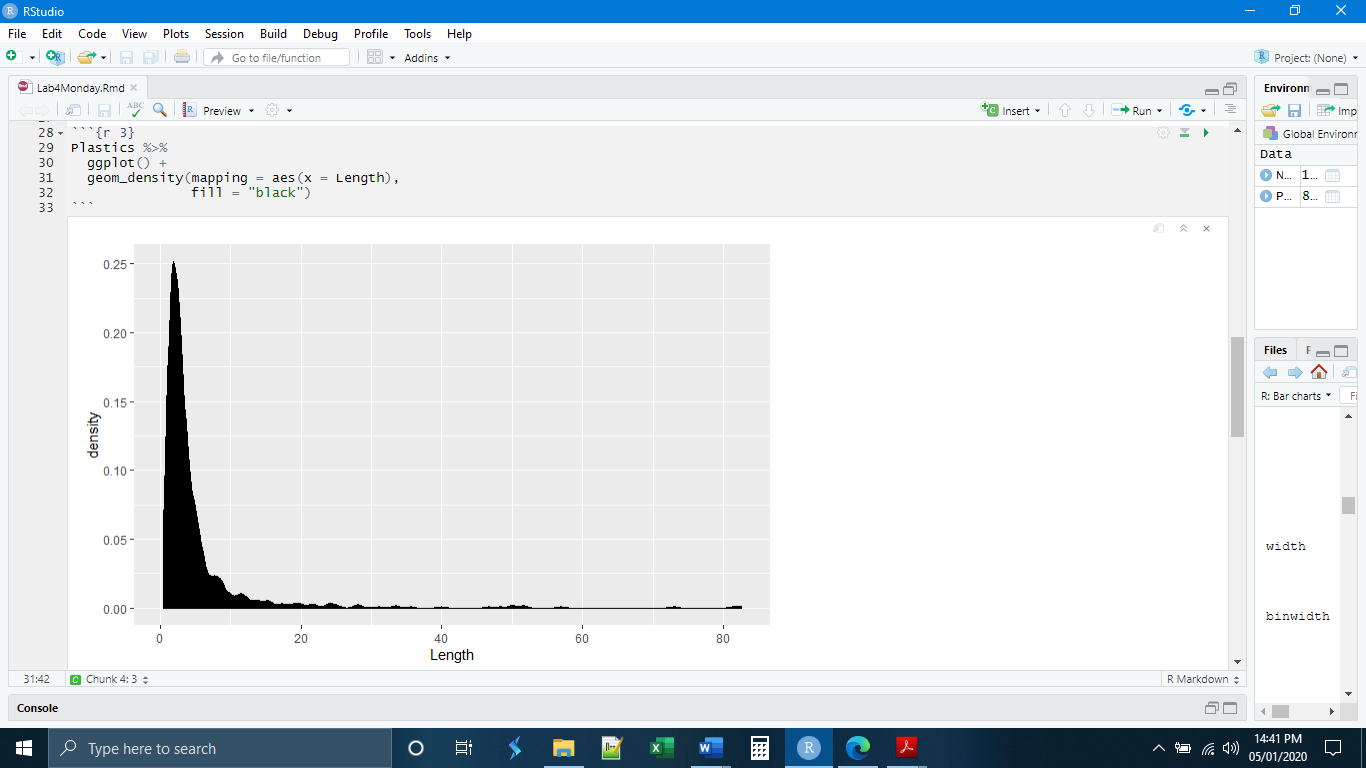
This plastics data catalogs all of the plastics recovered by the net tows data. Let’s begin by making a simple histogram of the lengths of the plastic pieces recovered.



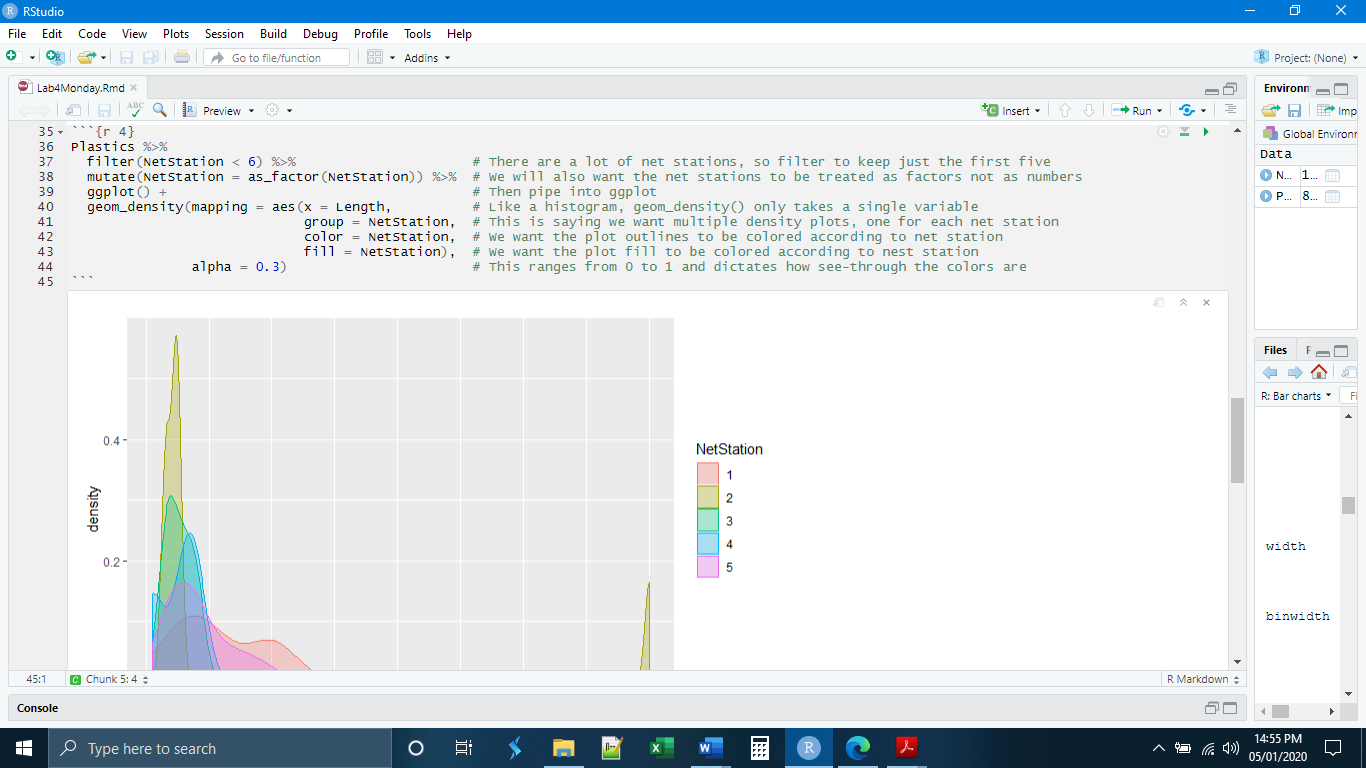
*geom\_density()*

An issue with histograms is that they often look blocky. This makes interpreting distributions from them difficult. Additionally, they are not always attractive. Density plots solve these problems by smoothing the bars to approximate a continuous distribution. Warning: density plots are not to be used to make data appear more impressive than they are.

Notice that in this case I have used the fill option. If we do not use the fill option the plot will simply be an outline of the shape.

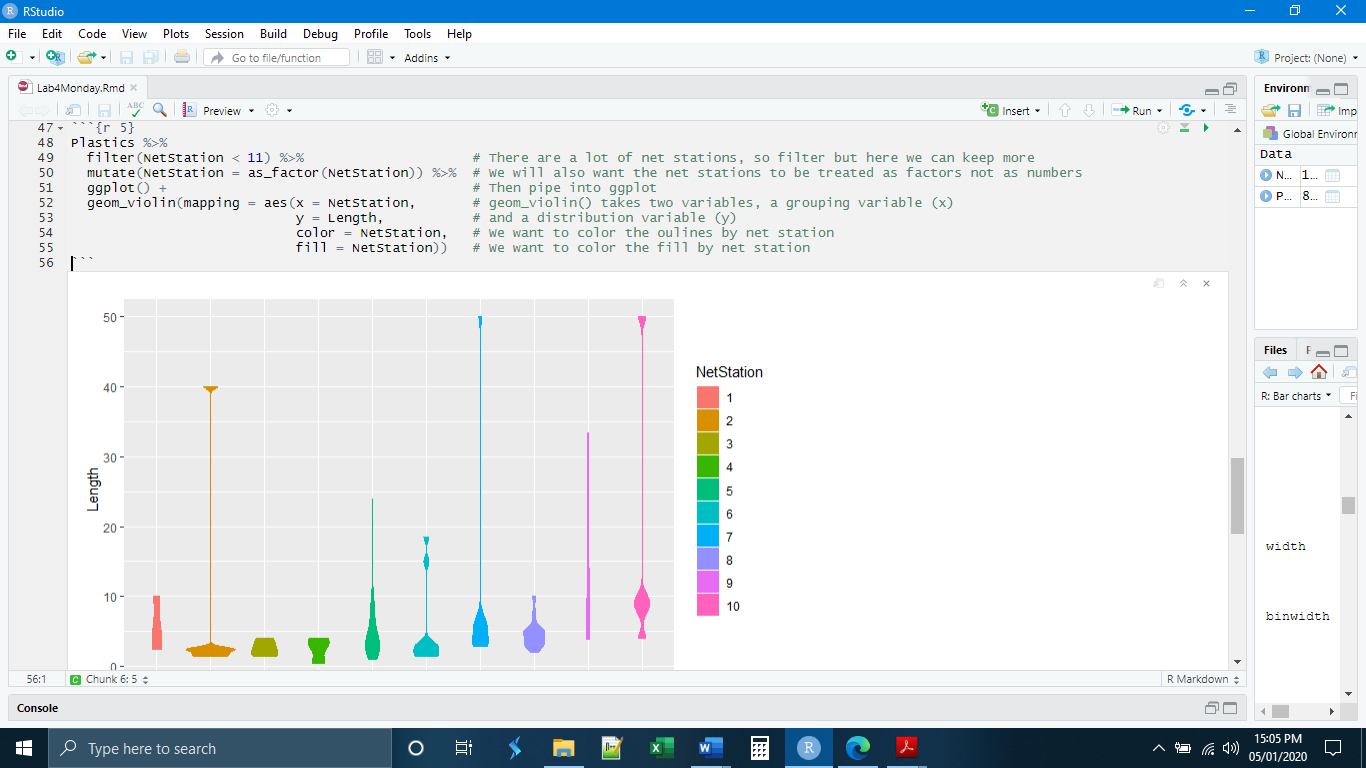


An additional benefit to the density plot is that it can be used to essentially layer histograms on top of each other. This makes it easy to compare the distributions of sections of the data. As you can see, there is a limit to the number of layers that can be applied and the plot still be readable.

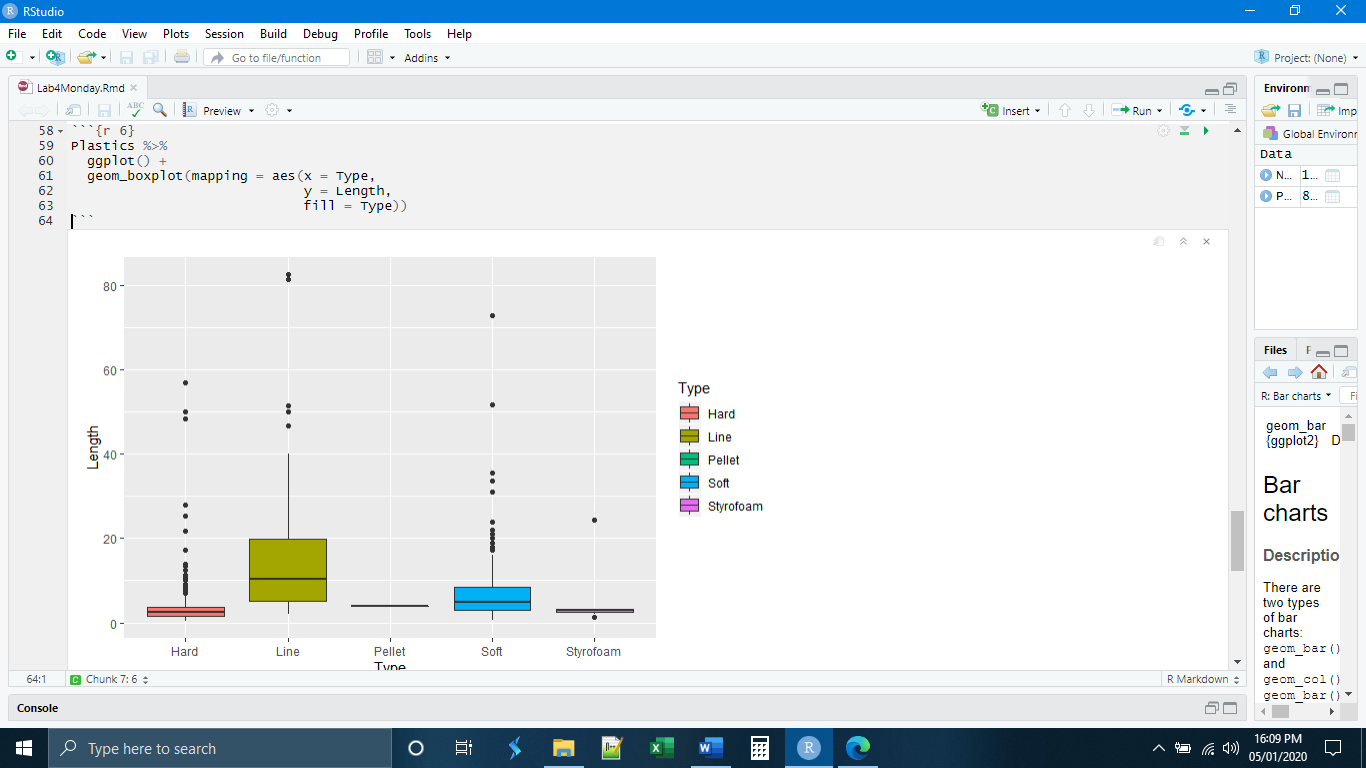


*geom\_violin()*

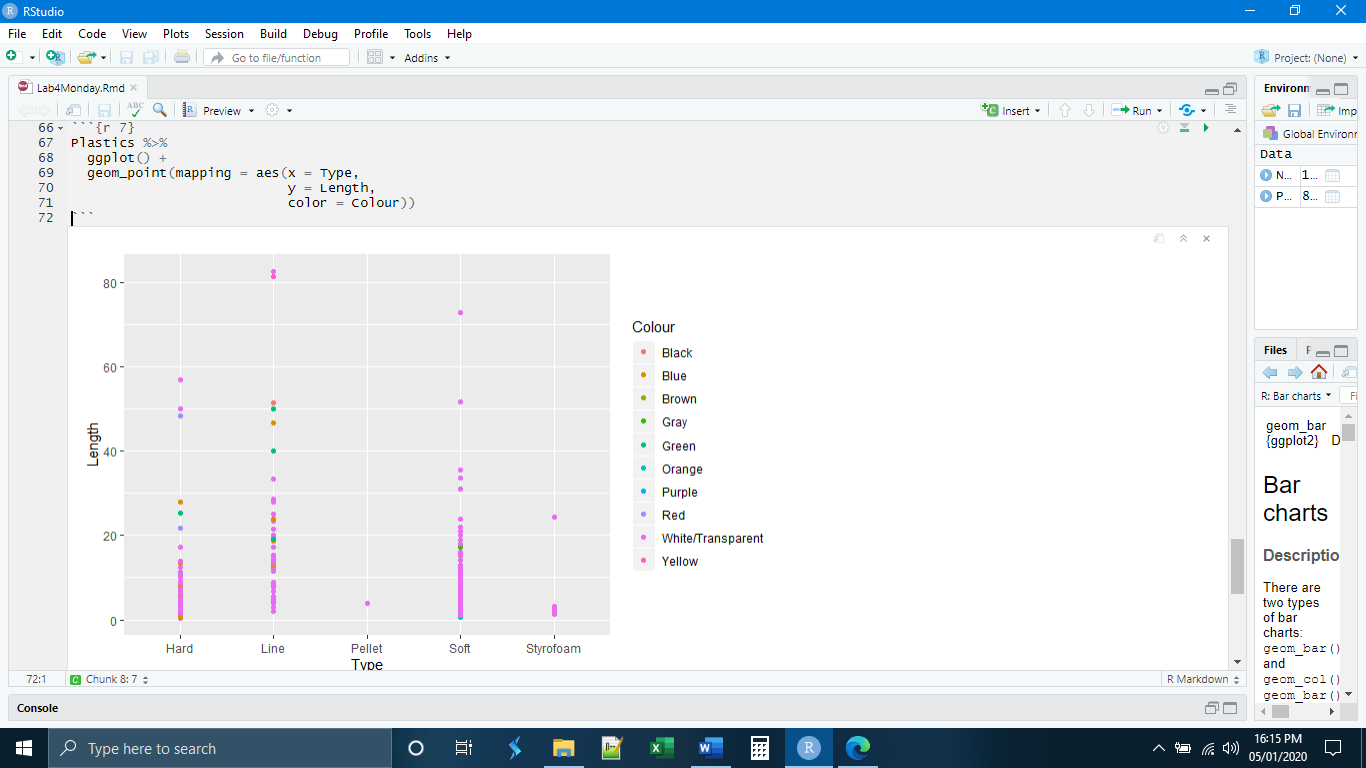
If the overlapping distributions are hard to interpret or you would like more than five, another option is geom\_violin(). Here a density plot is mirrored on either side of an axis. Each mirrored density plot is then set beside all of the others. This means that you can compare many distributions at once.



The plots we have covered so far are variations on the histogram. The next plot that we will cover is, in some ways a variation on a boxplot. Let’s begin by making a boxplot of the lengths of plastics by the plastic type.

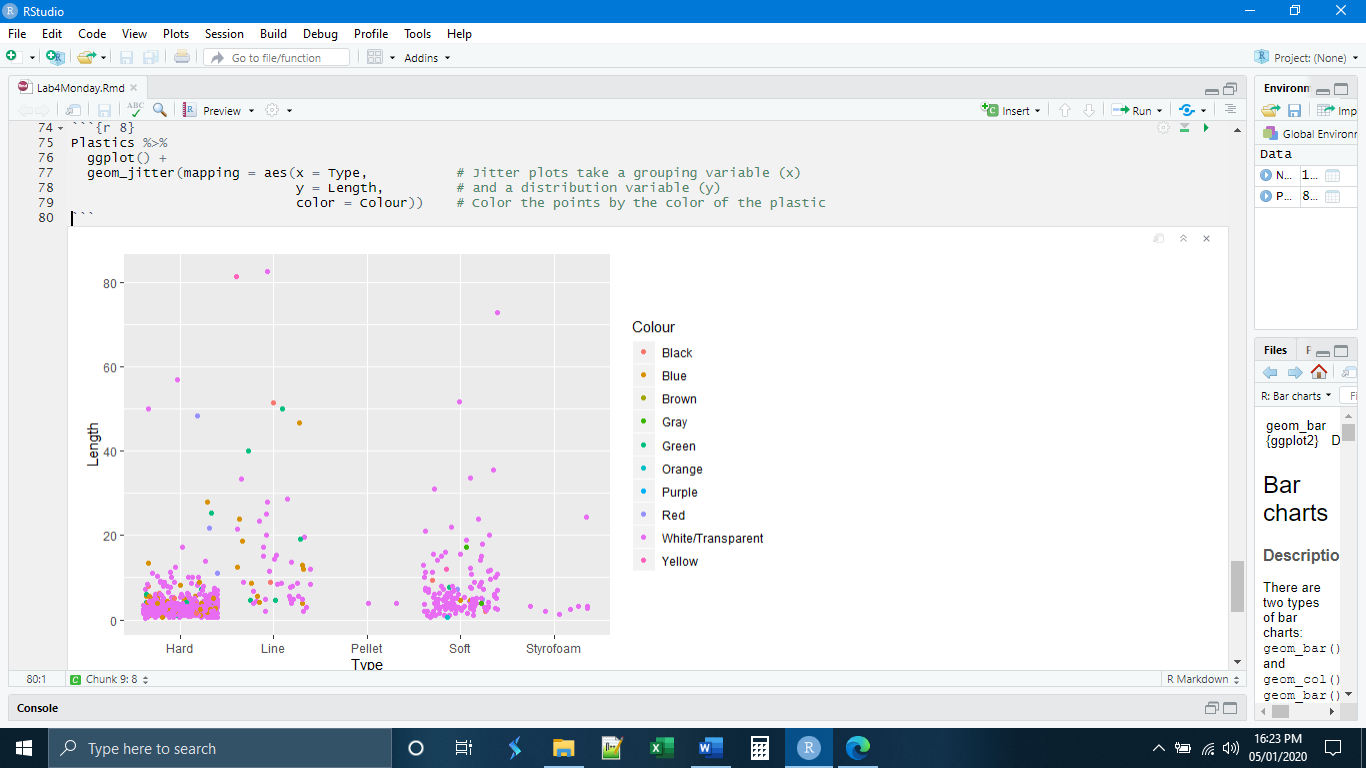


This is a fine plot, but what if we wanted to add another layer of information, for example the color of the plastic. We would not be able to do this with a boxplot. Moreover, a boxplot can also misrepresent the number of observations in a given category. We already know how to add color using a scatter plot, and it will show all the observations. But the geom\_point() stacks all of the points on top of each other and we lose some information.



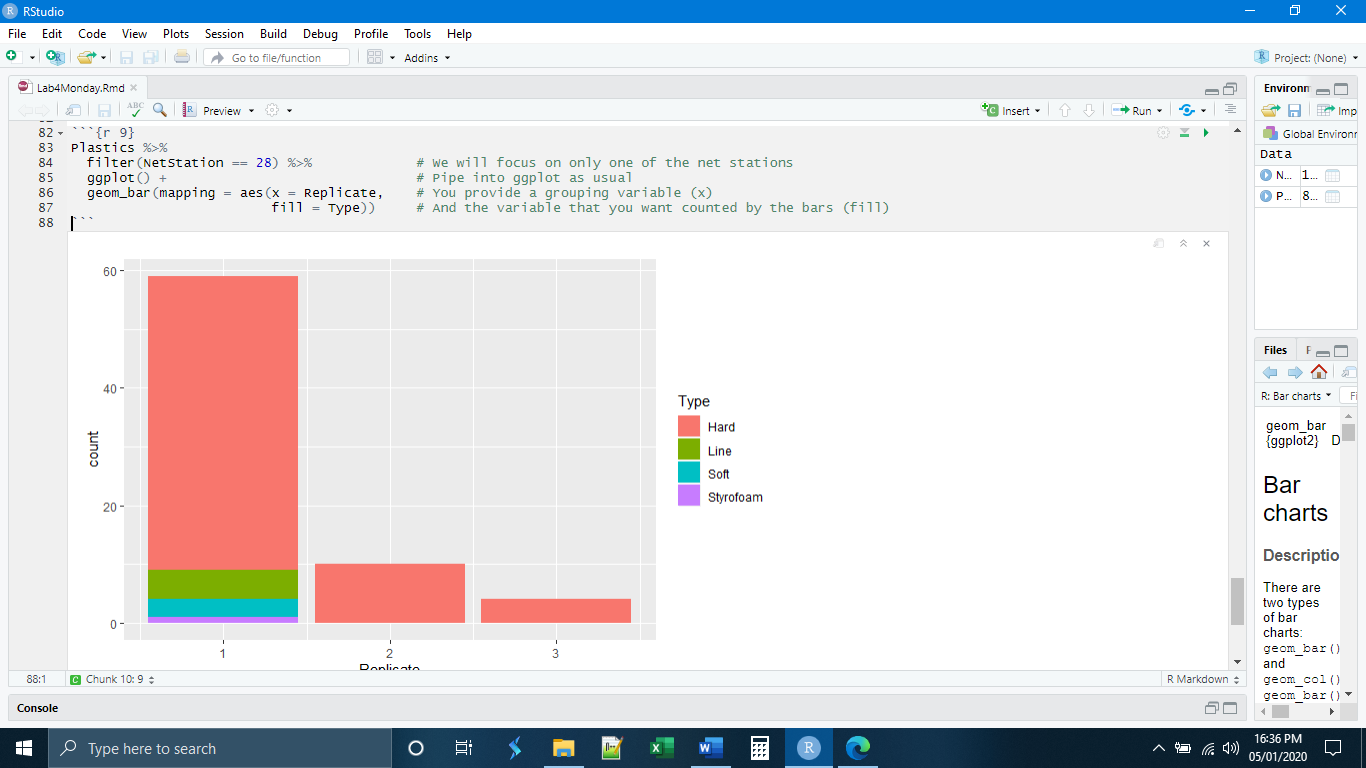
*geom\_jitter*

The jitter plot does not completely solve this problem but it helps to ameliorate it to some extent. Instead of the points being plotted in a line, they are randomly plotted to the left and right of a center line such that more of them are visible.

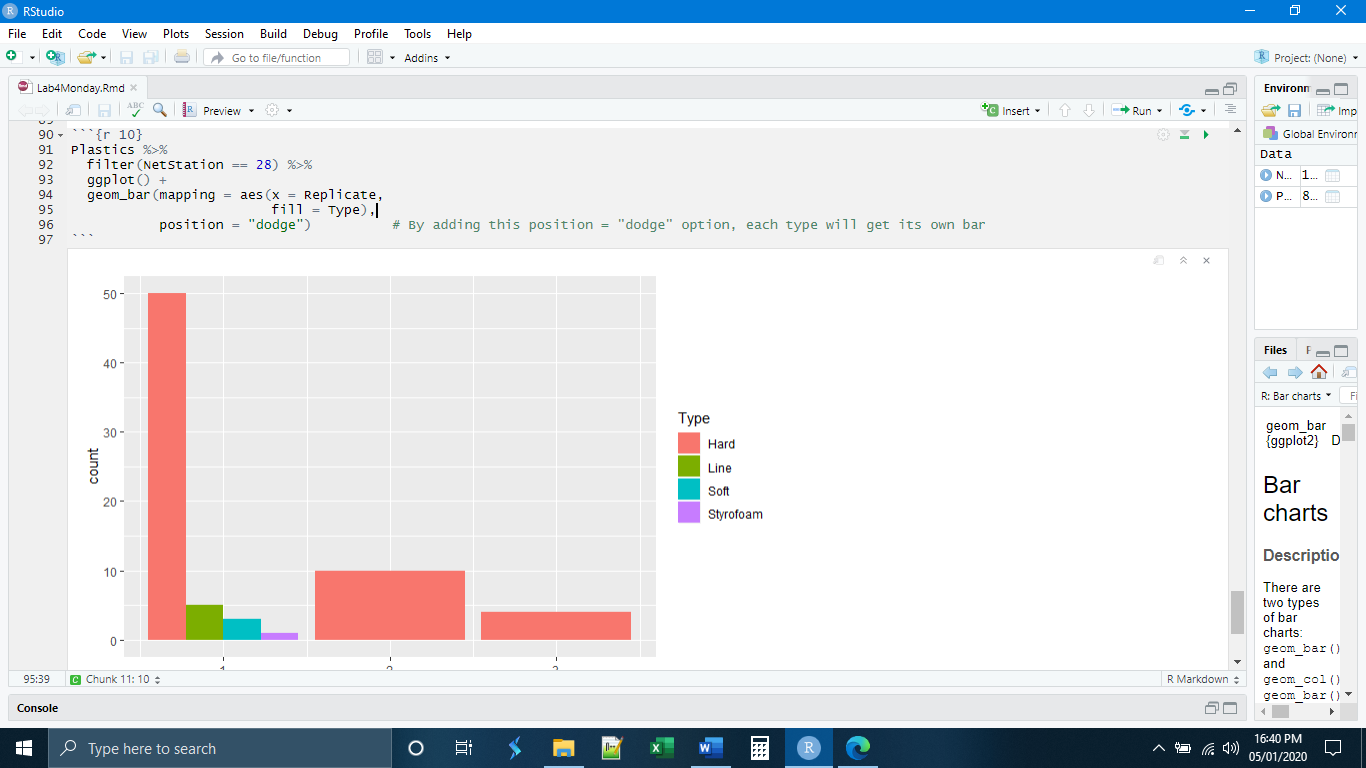


*geom\_bar()*

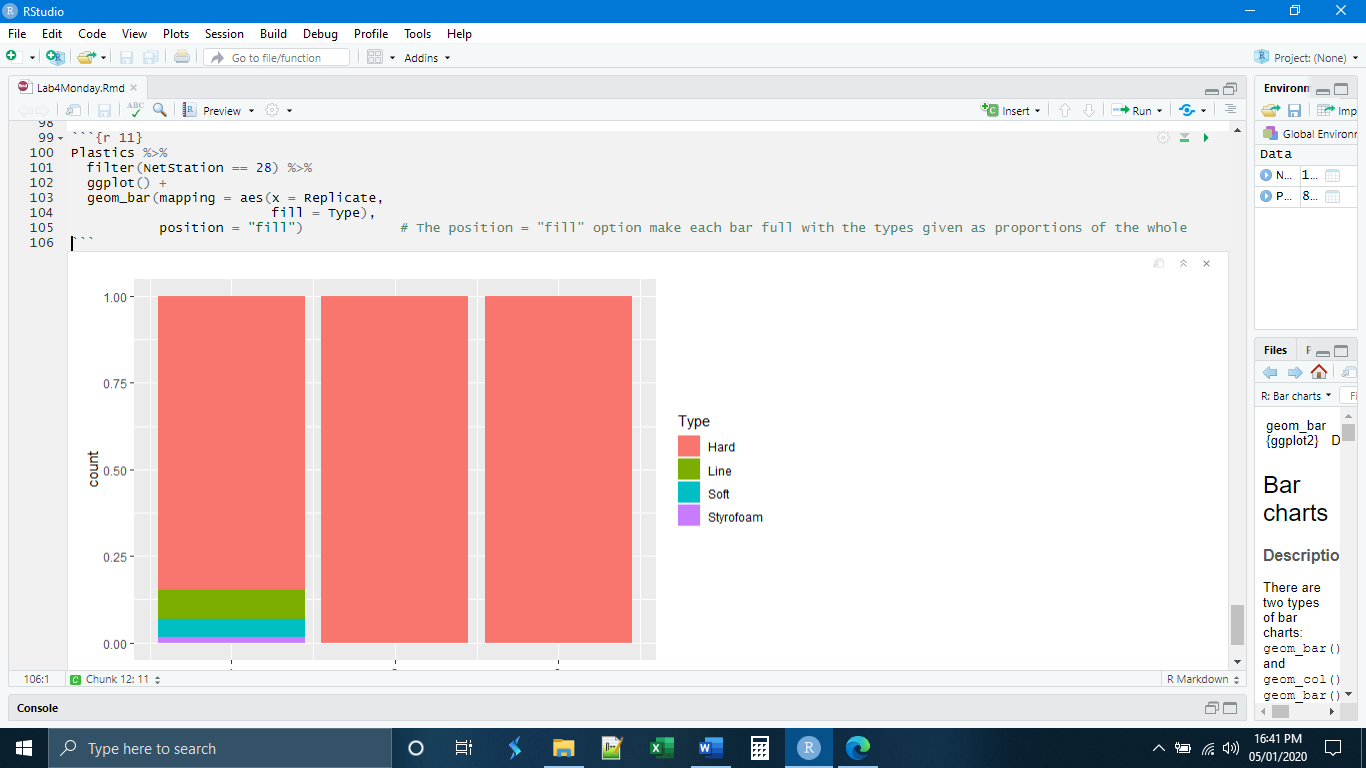
The last plot we will be going over is the bar plot. The bar plot is used for plotting the number of occurrences of one variable according the grouping of a second variable. The syntax is a little different than what we are accustomed to.



The basic bar graph simply shows the counts stacked on top of one another. This way, visually, the whole bar represents all of the plastics from each replicate with the relative contributions of the different types highlighted. By using the position = “dodge” option, we can give each type its own bar.



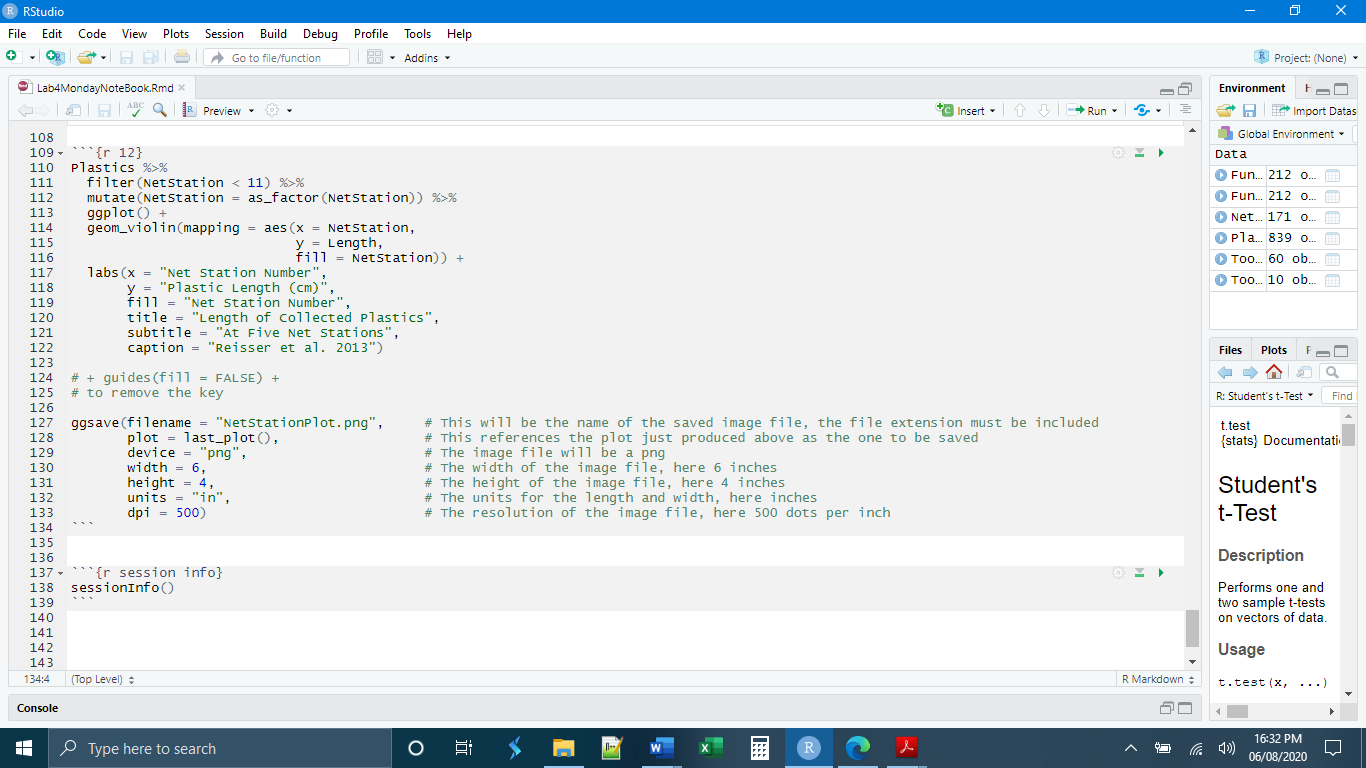
Another option is position = “fill”, allows you to turn each of the bars into a measure of proportion. Each bar will have a color in proportion to the number of observations of that type.



*ggsave()*

The final element of our introduction to ggplot2 is the ggsave() function, which saves plots produced in ggplot as image files. Like the labs() function, it is easy to copy and paste these lines of code at the end of a ggplot and change the parts that need to be changed. Unlike labs() the ggsave() function will not be part of the piped expression that produced the plot. ggsave() will be a new expression that references back to the plot just produced. The new file will be saved in the working directory, unless specified otherwise.

Let’s remake the plot from chunk 5, add some labels, and save it.



*Independent Exercises*

**1. (5 Points)** Make a new chunk called “chunk 1” and write code to

- Read in the nettows\_info.csv data (On Blackboard)

- Make a violin plot with the grouping variable "# VesselTripName" and the response (y) variable “TotalPlastics”

(Hint: You will need to rename "# VesselTripName" so that it is only one word)

- Use both the color and fill options to color the violins based on "# VesselTripName"

(Hint: Do the color and fill go inside or outside aes()?)

- Flip the coordinates of the plot to make the axes readable

**2. (5 Points)** Make a new chunk called “chunk 2” and write code to

- Make a bar plot of the nettows data with the grouping variable “CollectedBy” and the response (y) variable “Windspeed”.

(Hint: you will need to convert WindSpeed to a factor variable first)

- Use the position = option to make the bars represent proportions of trips with a given wind speed

- Flip the coordinates of the plot to make the axes readable