Lab 2. Introduction to Graphics in R

# Introduction

Today’s lab is all about graphics, and a little bit about R as a language. You will be making histograms and scatterplots of a data set about the rise and fall of metal bands, and using modifications of those graphics to help yourself understand how R works a little bit better. There is also an extra credit portion at the end!

# Learning Outcomes

By the end of this lab you should be able to:

* Use the read.csv() function to import your data
* Make a basic scatterplot using plot()
* Perform simple column math to make a new column
* Modify basic aesthetics of scatterplots and histograms like color, bin size, and point type
* Create, modify, and save scripts

# Part 1: Open Up a Script

Today’s lab is all about modifying code, which means that you will have a lot of code thrown at you. You do not need to type out all of today’s code, or copy-paste it all, because we are giving you a script. You can find this on D2L. To open it, go to **File**, select **Open File** and navigate to your R script.

NOTE: If you didn’t download the file as a .R file, you may have issues. Ask your Section Leader for assistance.

Now that you have your R code open, you can play around with the code more easily! Instead of typing it all out, you can modify it inside the script. Remember to save your changes regularly, and be aware that at the end of class today you will be copy-pasting your script to make sure to keep track of what you are doing!

# Part 2: Importing Data

Use the data import button like we did in Lab 1 to find and import the **metal.csv** file. Now, this is great, but it’s annoying to have to point and click and locate your file every time you open R, so what you’re going to do is learn how to make this action part of your R script.

See in the bottom of your console where the output is? All you have to do is copy and paste the read.csv() code with the file path inside.

metal <- read.csv("C:/Users/Meaghan/Dropbox/Science And School/Courses - Taught/ISTA 116 STATS/Labs/Lab 2 - Introduction to Graphics/metal.csv")

Paste this at the top of your script, and save your script. Now if your computer crashes or something you can open the R script and hit run and it will automatically load your data.

### QUESTION 1: Copy and paste your read.csv() code as Line 1 of your script. You can use the enter key to add space if you need it.

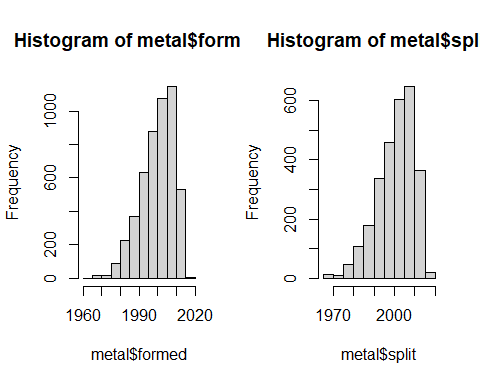
# Part 3: Modifying Histograms

## Part 3.1 Explore the Data

First off, let’s take a look at this data. Use the hist() code as follows to get a better idea of when most of these metal bands formed and when they split up.

hist(metal$formed)

hist(metal$split)



Now these are nice(ish) graphics, but the dollar sign in the labels is distracting and the colors are kind of bland. In today’s lab, you’re going to learn how to change all of those!

## Part 3.2 Colors and Labels

First off, just try running each of the different codes here. They all say “purple” in quotation marks, but each of them does a different thing. These are called **arguments**, which are chunks of code that modify your plot. Take a careful look at each plot to see what each argument changes.

hist(metal$formed, col = "purple")

hist(metal$formed, main = "purple")

hist(metal$formed, xlab = "purple")

hist(metal$formed, ylab = "purple")

hist(metal$formed, border = "purple")

### QUESTION 2: If you wanted to change the x axis label to read “boogers” what code would you run?

A. hist(metal$formed, col = "boogers")  
B. hist(metal$formed, main = "boogers")  
C. hist(metal$formed, xlab = "boogers")  
D. hist(metal$formed, ylab = "boogers")  
E. hist(metal$formed, border = "boogers")

### QUESTION 3: If you wanted to change the bars (not the outline) to be to the COLOR of boogers what code would you run?

A. hist(metal$formed, col = "olivedrab2")  
B. hist(metal$formed, main = "olivedrab2")  
C. hist(metal$formed, xlab = "olivedrab2")  
D. hist(metal$formed, ylab = "olivedrab2")  
E. hist(metal$formed, border = "olivedrab2")

NOTE: There are lots of colors available in R. Try this list for some ideas: <http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf>

## Part 3.3 Common Mistakes

You’ll notice for all of these examples, you have to put the color or label in quotation marks. What happens if you don’t? Run this code and look at the error message to find out:

hist(metal$formed, col = violetred2) #This Code Gives You An Error!

R is telling you that the object violetred2 is not found. Here’s the difference: if you put something in quotations, R knows it is running as a **character string**, where as if you don’t, it thinks it is looking for an **object** - something already named and in the environment window. Now, you CAN name a character (something in quotations) as an object and use that. For example, the following code runs just fine:

oohsopretty <- "violetred2"  
hist(metal$formed, col = oohsopretty)

### QUESTION 4: If you try to run the code: pretty <- violetred2 without quotation marks around the second bit ("violetred2"), R breaks. Why?

1. It hates this color
2. It is looking for an object called violetred2 that doesn’t exist
3. R just hates you and wishes for you to suffer.

Missing quotation marks can be an even more subtle problem - at least in this example, you’ve got a solid red warning in the console. There is a phenomena I like to refer to as the **Blue Plus Sign of Death** that can happen in R. It is a vortex of madness, but it is escapable - click in the console, and hit escape.

To replicate this mistake is really easy: only use half of a quotation set, or half of a parentheses. For example, run the code below:

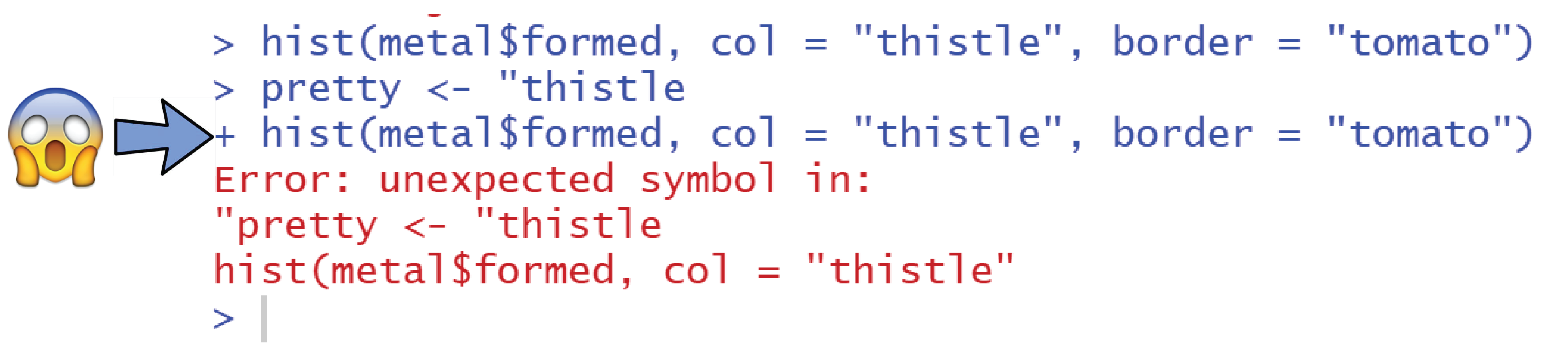
#Watch out, this code doesn't work!  
pretty <- "thistle

It looks like it ran, didn’t it? But beware - *you have been trapped in the vortex!!* If you run any other code, it won’t work now. Try running:

hist(metal$formed, col = "thistle", border = "tomato")

You’ll notice that this pretty pastel graphic does not show up! Instead, it gives you an “unexpected symbol” error. By not running your code correctly, it basically smooshed two sentences together (pretty <- “thistle hist(metal$formed, col =”)) until it completed the quotation mark.

Look carefully at the console. You see that plus sign?



It is as the ancient adage says:

If thine quotes are not in pairs  
Of the plus sign you must beware(s)

This also is the case for parentheses or, later on in class, square brackets. The troublesome thing about the plus sign is that it doesn’t always go away immediately. For example, try running the following code line by line:

pretty <- "thistle  
hist(metal$formed)  
2+2  
metal2 <- metal

You’ll see the plus signs begin to breed and take over in your console, forming an unstoppable army of irritation. Well, okay, not unstoppable - remember the cure is to click in the console and then press escape. Then you will be returned to the helpful blue carrot sign, which allows you to run code again.

### QUESTION 5: Which of these codes will summon the blue plus sign (select all that apply):

A. hist(metal$)  
B. hist((metal$formed)  
C. hist(metal$formed, col = "thistle"")  
D. hist(metal$formed, col = "thistle)  
E. hist(metal$formed, col = thistle)

## Part 3.4 Breaks and Limits

One other important thing about histograms to note is that the bars are known as **bins** and they can be manipulated - by changing the number, changing the placement, and changing how zoomed in or out the graphic is. Run the following code and pay close attention to each of the histograms you make to see how they are different.

hist(metal$formed, breaks = 5)

hist(metal$formed, breaks = c(1950, 1970, 1971, 1972, 2010,2020))

hist(metal$formed, xlim = c(2000, 2010))

hist(metal$formed, xlim = c(2000, 2010), breaks = 45)

hist(metal$formed, xlim = c(2000, 2010), breaks = c(1950, 2000, 2000.5, 2001, 2001.5, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2020))

hist(metal$formed, ylim = c(3, 45))

### QUESTION 6: If you wanted to make a histogram with 35 bins, which of the following code chunks would do that (select all that apply)?

A. hist(metal$formed, breaks = 35)  
B. hist(metal$formed, breaks = c(35))  
C. hist(metal$formed, xlim = 35)  
D. hist(metal$formed, xlim = c(35))

### QUESTION 7: One of the easiest ways to lie with your graphics for a histogram is to manipulate the bin dimensions. If you wanted folks to think that metal bands only started forming in the late 1990’s, which of these histograms would best sell that story?

A. hist(metal$formed, breaks = 5)  
B. hist(metal$formed, breaks = c(1920, 1990, 2000, 2005, 2010, 2015, 2020))  
C. hist(metal$formed, ylim = c(0, 100))  
D. hist(metal$formed, xlim = c(1900, 2020))

## Part 3.5 More Help

For more things you can do to modify your histogram or for help figuring out the code in the first place, you need to look into the R help code. Now, to be fair, you can usually google all of this. But sometimes it’s helpful to look at the manual. You access this by using *?*, i.e.:

?hist

# Part 3: Column Math

OK, so we have been looking a lot at the formation of metal bands. But what about their dissolution and sad decline? Well, there is the split column… but what if we want to know exactly how long a band has been around for? That requires doing a little math. Fortunately, that’s pretty easy in R. You just subtract them, like so:

metal$split - metal$formed

Of course, that makes a horrible mess in your console. So you really want to save it in your environment as an object, so that you can look at it later.You can do this the same way we did it in Lab 1:

totaltime <- metal$split - metal$formed

But we’ve been using full column names this whole time for plots. To stay consistent, it is easiest to probably just name this r object as a column, rather than as a value string. You can do this pretty easily - just make up a new column name, and use the dollar sign to indicate that this new column is part of metal:

metal$totaltime <- metal$split - metal$formed

### QUESTION 8: Which of the following codes would save the number of days a band was in existence as a column in the metal dataframe?

A. metal$days <- metal$totaltime \* 365  
B. days <- metal$totaltime \* 365  
C. metal$totaltime \* 365

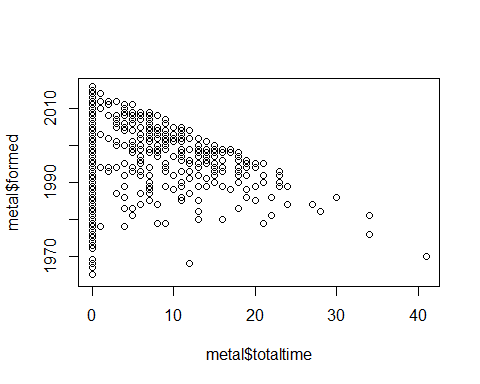
# Part 4: Scatter Plots

## Part 4.1 Basics

Now that we know the total time that a band was around, we can look to see if that is decreasing. Or, put another way - are metal bands made recently more likely to survive for a long time than metal bands that formed earlier on in metal band history?

To evaluate this question, we’re going to use the plot() function to make a scatterplot diagram. Scatterplots are **bivariate**, meaning they have two variables (histograms are **univariate** and only have one). So we have to use two columns, and specify to R that these two columns are two different objects. You can do that in R using a comma:

plot(metal$totaltime, metal$formed)



### QUESTION 9: Look at the plot. Which goes first, the x or the y axis when using a comma?

1. x is first
2. y is first

You can also use a **tilde**, or ~ symbol with this particular function. While the comma tells R “these two objects”, the tilde is more specific - it says “this object, according to this object.” Later on in term, we will be using functions with tildes more often.

plot(metal$totaltime~metal$formed)

### QUESTION 10: Run the code and look at the plot. Which goes first, the x or the y axis when using a tilde?

1. x is first
2. y is first

## Part 4.2 Modifications

Many of the arguments for plot() are the same as for hist() but there’s one argument below that won’t work in a histogram. Run the following code to see what each modification does.

plot(metal$totaltime~metal$formed, col = "purple")

plot(metal$totaltime~metal$formed, pch = 16)

plot(metal$totaltime~metal$formed, pch = 17)

plot(metal$totaltime~metal$formed, xlim = c(1990, 2010), ylim = c(0,10))

### QUESTION 11: If you wanted to make the points into triangles that were red, what code would work:

A. plot(metal$totaltime~metal$formed, col = "red", pch = 0)  
B. plot(metal$totaltime~metal$formed, col = "red", pch = 17)  
C. plot(metal$totaltime~metal$formed, fill = "red", pch = 17)  
D. plot(metal$totaltime~metal$formed, col = "red", xlim = 15)

NOTE: There are many different point shapes. Try this list for more examples: <http://www.sthda.com/english/wiki/r-plot-pch-symbols-the-different-point-shapes-available-in-r>

## Part 4.3 Analysis

Take a close look at your totaltime column and your graphic. Use a histogram if you need to. There seems to be a trend, where bands that formed earlier lasted much longer. However, there’s a problem with interpreting the trend that way.

### QUESTION 12: What is the problem with interpreting metal bands as now lasting shorter and shorter durations? There is more than one answer to this question, but the NA values in your dataset may help you find out one.

# Part 5: EXTRA CREDIT PLOT COMPETITIONS

It has all come down to this. Your final challenge for this lab - should you choose to accept it - is to make a histogram or a scatterplot of any data in the metal dataset. You may choose whether you want this to be an *ugly* plot competition (make such a hideous, deformed plot that no person could ever read it or understand it) or a *beautiful* plot competition (the opposite of that). Winners will be accepted for both categories though you can only win one.

The rules are:

1. it must demonstrate data from this lab
2. there must actually be data in it (no blank submissions)
3. it must be made in R, and in R alone (though you can use code not from this lab)
4. you must post your code to your D2L discussion board
5. your code must have: any column math you did, the plot code, and nothing else (don’t copy over the whole lab)

You may also include a copy of the plot if you like, but your entry will only be considered if you copy over the code.

The top 5 ugliest and top 5 most beautiful plots will get 1 point of extra credit. The section leaders and your instructor will be judging your plots based on their hideous/gorgeous nature, and also their technical difficulty in coding.

You are allowed to modify your code beyond what we’ve learned in lab. Here are some other things you can change if you like. Be aware that the abline() and text() functions only work if you highlight them and the plot() or hist() code and run them at the same time.

hist(metal$split)  
abline(h = 300, lty = 2)  
abline(v = 2000, lwd = 5, col = "blue")

plot(metal$totaltime~metal$split)  
abline(h = 10, lty = 2)  
abline(v = 2000, lwd = 5, col = "blue")  
abline(a = -200, b = .1, lwd = 2, col = "red")

plot(metal$totaltime~metal$split)  
text(x = 1975, y = 20, "Sad times for metal fans")  
text(x = 1999, y = 40, "yay metal", col = "red")

Best of luck to you all! May the best/worst plot win!