NAHSL CE: You say tomato, I say tomato: Data Visualization in Excel, R, and Tableau (R portion)

Tess Grynoch

October 4, 2019

Brief Introduction to R and RStudio

The term “R” is used to refer to both the programming language and the software that interprets the scripts written using it.

RStudio is currently a very popular way to not only write your R scripts but also to interact with the R software. To function correctly, RStudio needs R and therefore both need to be installed on your computer.

To make it easier to interact with R, we will use RStudio. RStudio is the most popular IDE (Integrated Development Interface) for R. An IDE is a piece of software that provides tools to make programming easier.

Reasons to use R:

- Easy to reproduce
- Visualization possibilities are limitless
- Able to create a visualization without altering the data we input into our script regardless of the analysis we run
- Open source
- Large and welcoming community

Create a new project

- Under the File menu > New Project > Existing directory > *Browse for the folder where you saved the data* > Create project
- Create a new file where we will type our scripts. Go to File > New File > R script OR click the plus sign hovering over the white rectangle in the toolbar. Click the save icon on your tooolbar, or, in the menu, File > Save As and save your script as “NAHSLCEvizs.R”

It’s good practice to keep all related data, analysis, and text in the same folder in order to use relative paths as opposed to the full path to the file’s location.

Start by giving our document a title using a comment which are denoted by a #

```
#Title as a comment
#by Name

#Comments allow you to add notes as you write your code
```

Notice how after we start making changes the file name in the tab turns red and has an * beside it. It's a reminder that we have not saved the new changes to the document yet. As we go along, remember to hit the save icon or control + s (On PC) or command + s (On Mac).

R and Rstudio have a number of basic built-in functions and arithmetic. To run a line or block of code, move your cursor to anywhere on the line or within the block and press control + enter (On PC) or command + return (On Mac).

```
2+2
```

You can also assign a name to a number, variable, table, function, or plot. To do this, use a <- to point to the name you want to use. This allows you use the name in place of the object.

```
4+2
y <- 4
y+2
```

R is also case sensitive so make sure to spell correctly.

```
y+2
Y+2 #You will get an error saying object 'Y' not found (Note you can also add
a comment to the end of a line of code)
```

Install packages and add libraries

To expand on what is already available through RStudio, we need to install packages. Today we'll be using tidyverse

```
install.packages("tidyverse")
```

Once we install the package, we need to tell it which libraries we will be using

```
library(tidyverse)
library(readxl)
```

If you need help or want to find out more about a library or function, add a ? to the beginning of the library or function and run the code.

```
?readxl
```

Load data

The first dataset we'll be using today is the Rotten Tomatoes Top Movies by Genre as of January 15, 2019. To load the data we're going to use the read_excel function and we're going to name the data as "data"

```
data <- read_excel("./ROTTEN_TOMATOES_TOP_MOVIES_20190115.xlsx",
                   sheet="AverageRevPerGenre")
```

I've used the relative path to the file above but if I were to use the full destination name it would look something like "C://Users/grynochc/Desktop/ROTTEN_TOMATOES_TOP_MOVIES_20190115.xlsx"

We can view a the column names and first 6 entries in our table by using the head function.

```
head(data)
```

Let's find out more about the head function.

```
?head
```

Guess what function is used to see the last six entries

```
tail(data)
```

You can also run a function on a particular column of a dataframe by indicating the column preceded by a $

```
max(data$AVGNumRev)
```

Bar Charts

Note: Since we are working with a dataframe, to reference a particular cell or area, row first, column second (opposite of excel who use column-row notation, ex. A2). In R, counting also starts at 1 (not 0 for any Python users in the crowd).

Question: What is the average number of reviews per genre? Who has the highest average of reviews? What order do the genres fall in?

With any ggplot chart, you start with what data you will be using.

```
ggplot(data = data)
```

Next, we'll define our plot area (mapping) using the aesthetic (aes) function. This generates a blank plot.

```
ggplot(data=data, mapping=aes(x=Genre, y=AVGNumRev))
```

We add the various elements to the chart as geoms with a + symbol on the previous line which will automatically indent the next line to keep all our code in a block to run together.

```
ggplot(data = data, mapping=aes(x=Genre, y=AVGNumRev)) +
  geom_col()
```

We have our bar chart but it is not in the same order as the table. Any guesses as to what has happened?

To get our chart to order the bars to resemble the order of our table we need to explicitly order the factor levels of the x axis, Genre.

```
data$Genre <- factor(data$Genre, levels = data$Genre[order(data$AVGNumRev)])

data$Genre #check

#run chart again
ggplot(data = data, mapping=aes(x=Genre, y=AVGNumRev)) +
  geom_col() #now ordered!
```

We now have a basic bar chart but based on the best practices we learned there are a number of improvements we can make to this chart to better communicate our story.

Since R has such infinite variety of features for charts and there is usually more than one way to accomplish a task, I find it useful to create a plot wish list for all the improvements I want to accomplish with a chart. This is also useful if you are new to R and need to look up how to make the changes.

Plot wish list

- Add title
- Change axes titles
- Flip axes
- Change the color of the bars
- Change the theme of the chart (background, axes, etc.)
- Add data labels
- Highlight the bar of my favorite genre

Addings title and axes using labs and formatting using theme(axis.text.x, plot.title)

```
ggplot(data = data, mapping=aes(x=Genre, y=AVGNumRev)) +
  geom_col() +
  labs(x="Genre", y= "Number of Reviews", title ="Average number of reviews pe
r genre for the top 100 movies from Rotten Tomatoes") +
  theme(axis.text.x = element_text(angle=45, hjust = 1),
        plot.title = element_text(size=24))
```

Check out what else you can do with element_text

```
?element_text
```

To flip the x and y axes to create a horizontal bar chart, we can use coord_flip

```
ggplot(data = data, mapping=aes(x=Genre, y=AVGNumRev)) +
  geom_col() +
  labs(x="Genre", y= "Number of Reviews", title ="Average number of reviews pe
r genre for the top 100 movies from Rotten Tomatoes") +
  coord_flip()
```

To change the color of our bars we are going to use the fill aesthetic within geom_col. There is also a color aesthetic and it dictates the outline of the bars.

The fill aesthetic will accept a variable name if you want the colors to be based on a category (ex. Genre), there is also a limited number of color names that can be used (“red”, “green”, “blue”, etc.), but if there is a specific color that you want to use, I recommend using hexidecimal color codes. My preferred site for finding hexideximal color codes is color-hex.com (https://color-hex.com) but there are many other options.

```
ggplot(data = data, mapping=aes(x=Genre, y=AVGNumRev)) +
  geom_col(fill = "#aba8a8") +
  labs(x="Genre", y= "Number of Reviews", title ="Average number of reviews pe
r genre for the top 100 movies from Rotten Tomatoes") +
  coord_flip() +
  theme(plot.title = element_text(size=20, hjust=.5))
```

Check out what other aesthetics are available in geom_col

```
?geom_col
```

The theme of your chart dictates the overall look of your chart and are preset combinations of backgrounds, axes formats, and gridline formats. A shortlist of formats is located on the second page of the ggplot2 cheatsheet. I’m a fan of the classic theme so that’s what I’m going to use.

```
ggplot(data = data, mapping=aes(x=Genre, y=AVGNumRev)) +
  geom_col(fill = "#aba8a8") +
  labs(x="Genre", y= "Number of Reviews", title ="Average number of reviews pe
r genre for the top 100 movies from Rotten Tomatoes") +
  coord_flip() +
  theme(plot.title = element_text(size=20, hjust=.5)) +
  theme_classic()
```

To add data labels to our bar chart we’ll use the geom_text which we’ll adjust horizontally to sit outside the bars.

```
ggplot(data = data, mapping=aes(x=Genre, y=AVGNumRev)) +
  geom_col(fill = "#aba8a8") +
  labs(x="Genre", y= "Number of Reviews", title ="Average number of reviews pe
r genre for the top 100 movies from Rotten Tomatoes") +
  coord_flip() +
  theme(plot.title = element_text(size=20, hjust=.5)) +
  theme_classic() +
  geom_text(aes(label=AVGNumRev), hjust=0)
```

Lastly we'll highlight one row by adding another geom_col and filter for our favorite genre.

```
ggplot(data = data, mapping=aes(x=Genre, y=AVGNumRev)) +
  geom_col(fill = "#aba8a8") +
  labs(x="Genre", y= "Number of Reviews", title ="Average number of reviews pe
r genre for the top 100 movies from Rotten Tomatoes") +
  coord_flip() +
  theme(plot.title = element_text(size=20, hjust=.5)) +
  theme_classic() +
  geom_text(aes(label=AVGNumRev), hjust=0) +
  geom_col(data = filter(data, Genre == "Animation"), fill="#c90000")
```

Congratulations on completing the plot wish list!

To export our final chart and share it with the world, we'll first assign our chart a name then use ggsave to export

```
RevPerGenreChart <- ggplot(data = data, mapping=aes(x=Genre, y=AVGNumRev)) +
  geom_col(fill = "#aba8a8") +
  labs(x="Genre", y= "Number of Reviews", title ="Average number of reviews pe
r genre for the top 100 movies from Rotten Tomatoes") +
  coord_flip() +
  theme(plot.title = element_text(size=20, hjust=.5)) +
  theme_classic() +
  geom_text(aes(label=AVGNumRev), hjust=0) +
  geom_col(data = filter(data, Genre == "Animation"), fill="#c90000")

RevPerGenreChart

#Export chart
ggsave("./Charts/ReviewsPerGenre.png",RevPerGenreChart)
```

With ggsave you can also specify width, height, and units.

Histograms and stacked bar charts

Question: What is the distribution of genres in each rating group (1-10, 11-20, etc.)?

For this question, we're going to create a histogram of the rating using geom_histogram and stack the

genres inside using the fill aesthetic.

We're also going to use the full Rotten Tomatoes dataset so we need to import the sheet with the full dataset.

```
fulldata <- read_excel("./ROTTEN_TOMATOES_TOP_MOVIES_20190115.xlsx", sheet="rot
ten_tomatoes_top_movies_2019")

ggplot(data=fulldata, mapping=aes(x=Rating))+
  geom_histogram(aes(fill=Genres), binwidth = 10)
```

Find out more about geom_histogram

```
?geom_histogram
```

Scatter plot

Question: Is there a correlation between number of reviews and rating? Are the genres grouped together?

Scatter plots employ geom_point

```
ggplot(data=fulldata, mapping=aes(x=NumReviews, y=Rating))+
  geom_point(aes(color=Genres), shape=16, size=2)
```

With geom_point you also have the option to designate the shape of the points. The shapes are numbered 1-25 and are listed on the ggplot 2 cheatsheet. Some of the points have a fill and a color aesthetic and some only have color.

Horizontal dumbbell dot plot (Australia data)

Question: What is the difference between the average minimum and maximum temperature in 4 Australian cities in August 2019?

Import the Australia temperature dataset

```
#import data
austemp <- read_excel("./DAILY_WEATHER_AUSTRALIA.xlsx", sheet="USE_FAHRENHEIT")

head(austemp)
```

A horizontal dumbbell dot plot, which I will refer to as the dot plot from now on, is composed on lines and points.
Let's start with the lines

```
ggplot(austemp)+
  geom_segment(aes(x=AvgMaxTemp, xend=AvgMinTemp, y=City, yend=City), size=2)
```

Next, we'll add our points

```
ggplot(austemp)+
  geom_segment(aes(x=AvgMaxTemp, xend=AvgMinTemp, y=City, yend=City), size=2) +
  geom_point(aes(x=AvgMinTemp, y=City), size=8) +
  geom_point(aes(x=AvgMaxTemp, y=City), size=8)
```

Now, we'll add labels for our dots

```
ggplot(austemp)+
  geom_segment(aes(x=AvgMaxTemp, xend=AvgMinTemp, y=City, yend=City), size=2) +
  geom_point(aes(x=AvgMinTemp, y=City), size=8) +
  geom_point(aes(x=AvgMaxTemp, y=City), size=8) +
  geom_text(aes(x=AvgMinTemp, y=City, label = AvgMinTemp), size=5, color="whit
e") +
  geom_text(aes(x=AvgMaxTemp, y=City, label = AvgMaxTemp), size=5, color="whit
e")
```

Lastly, we'll add color to the dumbbells, add our title, and edit the axes labels

```
ggplot(austemp)+
  geom_segment(aes(x=AvgMaxTemp, xend=AvgMinTemp, y=City, yend=City, color=Cit
y), size=2) +
  geom_point(aes(x=AvgMinTemp, y=City, color=City), size=8) +
  geom_point(aes(x=AvgMaxTemp, y=City, color=City), size=8) +
  geom_text(aes(x=AvgMinTemp, y=City, label = AvgMinTemp), size=5, color="whit
e") +
  geom_text(aes(x=AvgMaxTemp, y=City, label = AvgMaxTemp), size=5, color="whit
e") +
  labs(x="Temperature (F)", y= "City", title ="Average temperature, minimum an
d maximum in F, for 4 cities in Australia")
```

Bonus: Slope graph (Australia data)

A slope graph is also composed of lines and points and we'll start with the lines again.
Note the difference in the x and y axes of the dot plot and slope graph.

```
ggplot(austemp)+
  geom_segment(aes(x=0, y=AvgMinTemp, xend=1, yend=AvgMaxTemp), size=2)
```

Then add the points

```
ggplot(austemp)+
  geom_segment(aes(x=0, y=AvgMinTemp, xend=1, yend=AvgMaxTemp), size=2)+
  geom_point(aes(x=0, y=AvgMinTemp), size=4)+
  geom_point(aes(x=1, y=AvgMaxTemp), size=4)
```

We're going to add a little extra room for our labels by adjusting the scale on the x axis. We can also adjust the limits of our y axis to center our chart.

```
ggplot(austemp)+
  geom_segment(aes(x=0, y=AvgMinTemp, xend=1, yend=AvgMaxTemp), size=2)+
  geom_point(aes(x=0, y=AvgMinTemp), size=4)+
  geom_point(aes(x=1, y=AvgMaxTemp), size=4)+
  scale_x_continuous(limits = c(-0.5,1.5), breaks = c(0,1), labels = c("Averag
e Min Temp","Average Max Temp"))+
  ylim(30,70)
```

Lastly we'll add our data labels using annotate, a different function to perform the same task as the geom_text function, and title

```
ggplot(austemp)+
  geom_segment(aes(x=0, y=AvgMinTemp, xend=1, yend=AvgMaxTemp), size=2)+
  geom_point(aes(x=0, y=AvgMinTemp), size=4)+
  geom_point(aes(x=1, y=AvgMaxTemp), size=4)+
  scale_x_continuous(limits = c(-0.5,1.5), breaks = c(0,1), labels = c("Averag
e Min Temp","Average Max Temp"))+
  ylim(30,70)+
  annotate("text", x=-.08, y=austemp$AvgMinTemp, label=austemp$AvgMinTemp)+
  annotate("text", x=1.08, y=austemp$AvgMaxTemp, label=austemp$AvgMaxTemp)+
  ggtitle("Average temperature, minimum and maximum in F, for 4 cities in Austr
alia")
```

Adding a Benchmark Line to a Bar Chart

For this example we're going to add a median line to the Rotten Tomatoes bar chart we created earlier.

Let's start by determining what our median is:

```
median(data$AVGNumRev)
half <- median(data$AVGNumRev) #Giving our result, 107, a name
```

Then we'll revisit our previous bar chart and add the median line using geom_hline

```
RevPerGenreChart +
  geom_hline(yintercept = half)
```

Materials from The Carpentries' (previously Data Carpentry's) R for Social Scientists were incorporated into this class and used under the CC-BY 4.0 license.