# Intro to Data Science - HW 6

##### Copyright Jeffrey Stanton, Jeffrey Saltz, and Jasmina Tacheva

# Rutwik Ghag

### Attribution statement: (choose only one and delete the rest)

# 1. I did this homework by myself, with help from the book and the professor.

**This module:** **Data visualization** is important because many people can make sense of data more easily when it is presented in graphic form. As a data scientist, you will have to present complex data to decision makers in a form that makes the data interpretable for them. From your experience with Excel and other tools, you know that there are a variety of **common data visualizations** (e.g., pie charts). How many of them can you name?

The most powerful tool for data visualization in R is called **ggplot**. Written by computer/data scientist **Hadley Wickham**, this **“graphics grammar”** tool builds visualizations in layers. This method provides immense flexibility, but takes a bit of practice to master.

## Step 1: Make a copy of the data

1. Read the **who** dataset from this URL: <https://intro-datascience.s3.us-east-2.amazonaws.com/who.csv> into a new dataframe called **tb**.

Your new dataframe, tb, contains a so-called **multivariate time series**: a sequence of measurements on 23 Tuberculosis-related (TB) variables captured repeatedly over time (1980-2013). Familiarize yourself with the nature of the 23 variables by consulting the dataset’s codebook which can be found here: <https://intro-datascience.s3.us-east-2.amazonaws.com/TB_data_dictionary_2021-02-06.csv>.

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──

## ✓ ggplot2 3.3.5 ✓ purrr 0.3.4  
## ✓ tibble 3.1.4 ✓ dplyr 1.0.7  
## ✓ tidyr 1.1.3 ✓ stringr 1.4.0  
## ✓ readr 2.0.1 ✓ forcats 0.5.1

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

tb <- read\_csv("https://intro-datascience.s3.us-east-2.amazonaws.com/who.csv")

## Rows: 5769 Columns: 23

## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (1): iso2  
## dbl (22): year, new\_sp, new\_sp\_m04, new\_sp\_m514, new\_sp\_m014, new\_sp\_m1524, ...

##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

1. How often were these measurements taken (in other words, at what frequency were the variables measured)? Put your answer in a comment.

#The frequency was 1 year, i.e. they were measured yearly from 1980 to 2008

## Step 2: Clean-up the NAs and create a subset

1. Let’s clean up the iso2 attribute in **tb**

Hint: use *is.na()* – well use *! is.na()*

tb <- tb[rowSums(is.na(tb)) + 2 != ncol(tb) , ]

B. Create a subset of **tb** containing **only the records for Canada (“CA” in the iso2 variable)**. Save it in a new dataframe called **tbCan**. Make sure this new df has **29 observations and 23 variables**.

tbCan <- tb[tb$iso2 == "CA",]  
tbCan <- tbCan[rowSums(is.na(tbCan)) != ncol(tbCan) , ]

C. A simple method for dealing with small amounts of **missing data** in a numeric variable is to **substitute the mean of the variable in place of each missing datum**. This expression locates (and reports to the console) all the missing data elements in the variable measuring the **number of positive pulmonary smear tests for male children 0-4 years old** (there are 26 data points missing)

tbCan$new\_sp\_m04[is.na(tbCan$new\_sp\_m04)]

## [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA  
## [26] NA

Error in eval(expr, envir, enclos): object 'tbCan' not found  
Traceback:

1. Write a comment describing how that statement works.

#Boolean values are generated based on the condition and the corresponding values are retrieved

1. Write 4 more statements to check if there is missing data for the number of positive pulmonary smear tests for: **male and female** children 0-14 years old (**new\_sp\_m014** and **new\_sp\_f014**), and **male and female citizens 65 years of age and older**, respectively. What does empty output suggest about the number of missing observations?

tbCan$new\_sp\_m014[is.na(tbCan$new\_sp\_m014)]

## numeric(0)

tbCan$new\_sp\_f014[is.na(tbCan$new\_sp\_f014)]

## numeric(0)

tbCan$new\_sp\_f65[is.na(tbCan$new\_sp\_f65)]

## numeric(0)

tbCan$new\_sp\_m65[is.na(tbCan$new\_sp\_m65)]

## numeric(0)

# There are no missing values as the output has no missing values

There is an R package called **imputeTS** specifically designed to repair missing values in time series data. We will use this instead of mean substitution. The **na\_interpolation()** function in this package takes advantage of a unique characteristic of time series data: **neighboring points in time can be used to “guess” about a missing value in between**.

1. Install the **imputeTS** package (if needed) and use **na\_interpolation( )** on the variable from part C. Don’t forget that you need to save the results back to the **tbCan** dataframe. Also update any attribute discussed in part E (if needed).

#install.packages("imputeTS")  
library(imputeTS)

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

tbCan$new\_sp\_m04 <- na\_interpolation(tbCan$new\_sp\_m04)

1. Rerun the code from C and E above to check that all missing data have been fixed.

tbCan$new\_sp\_m04[is.na(tbCan$new\_sp\_m04)]

## numeric(0)

tbCan$new\_sp\_m014[is.na(tbCan$new\_sp\_m014)]

## numeric(0)

tbCan$new\_sp\_f014[is.na(tbCan$new\_sp\_f014)]

## numeric(0)

tbCan$new\_sp\_f65[is.na(tbCan$new\_sp\_f65)]

## numeric(0)

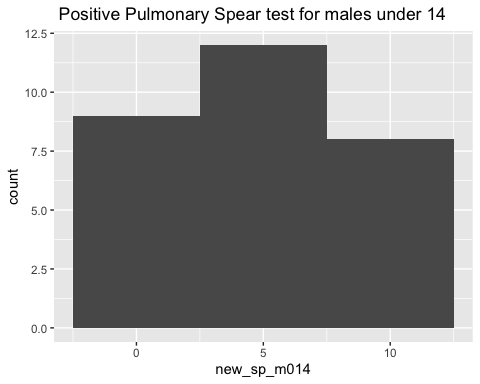
tbCan$new\_sp\_m65[is.na(tbCan$new\_sp\_m65)]

## numeric(0)

## Step 3: Use ggplot to explore the distribution of each variable

**Don’t forget to install and library the ggplot2 package.** Then: H. Create a histogram for **new\_sp\_m014**. Be sure to add a title and briefly describe what the histogram means in a comment.

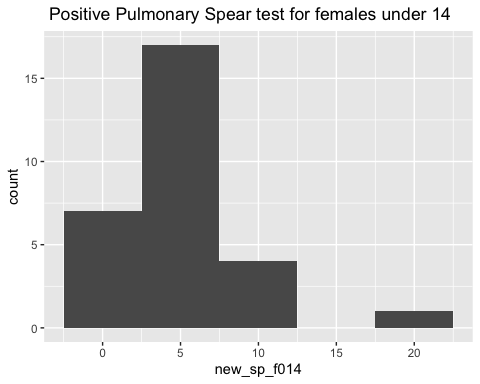
library(ggplot2)  
myplot <- ggplot(data = tbCan, aes(new\_sp\_m014)) + geom\_histogram(binwidth = 5)  
myplot <- myplot + ggtitle(" Positive Pulmonary Spear test for males under 14")  
myplot



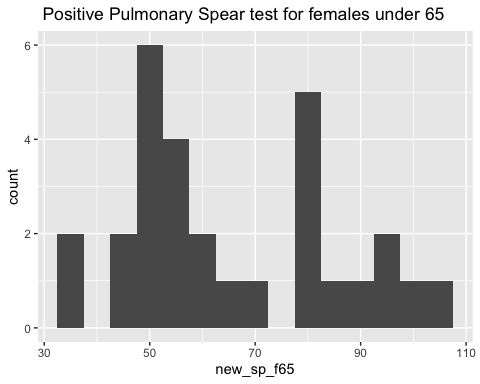
# the histogram gives us a frequency plot of the no.of +ve pulmonary spear tests for males under 14 in Canada between the time frame mentioned

1. Create histograms (using ggplot) of each of the other three variables from E with ggplot( ). Which parameter do you need to adjust to make the other histograms look right?

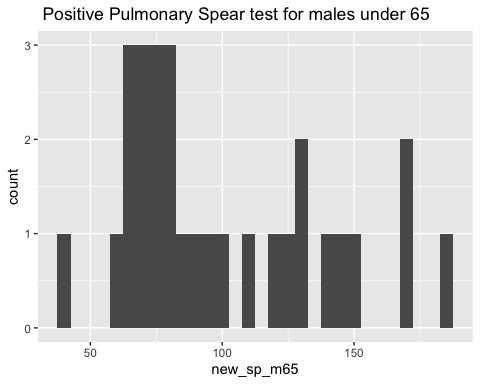
myplot1 <- ggplot(data = tbCan, aes(new\_sp\_f014)) + geom\_histogram(binwidth = 5)  
myplot1 <- myplot1 + ggtitle(" Positive Pulmonary Spear test for females under 14")  
myplot1



myplot2 <- ggplot(data = tbCan, aes(new\_sp\_f65)) + geom\_histogram(binwidth = 5)  
myplot2 <- myplot2 + ggtitle(" Positive Pulmonary Spear test for females under 65")  
myplot2



myplot3 <- ggplot(data = tbCan, aes(new\_sp\_m65)) + geom\_histogram(binwidth = 5)  
myplot3 <- myplot3 + ggtitle(" Positive Pulmonary Spear test for males under 65")  
myplot3

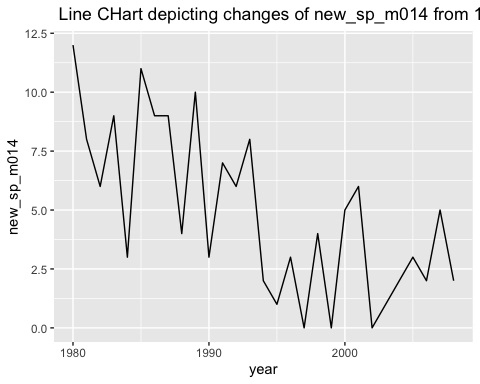


# The bin\_width parameter needs to be adjusted

## Step 4: Explore how the data changes over time

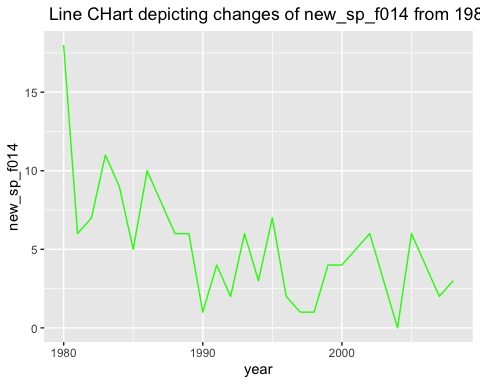
1. These data were collected in a period of several decades (1980-2013). You can thus observe changes over time with the help of a line chart. Create a **line chart**, with **year** on the X-axis and **new\_sp\_m014** on the Y-axis.

ggplot(tbCan, aes(x= year, y=new\_sp\_m014)) +  
 geom\_line() + ggtitle(" Line CHart depicting changes of new\_sp\_m014 from 1980 - 2013")

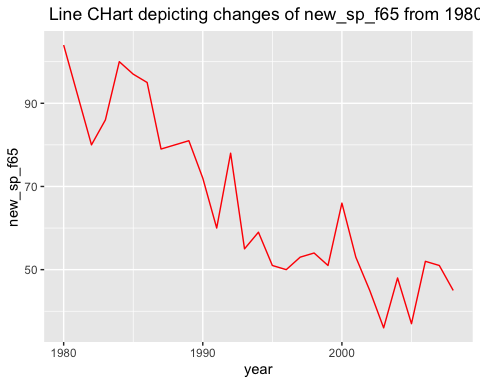


1. Next, create similar graphs for each of the other three variables. Change the **color** of the line plots (any color you want).

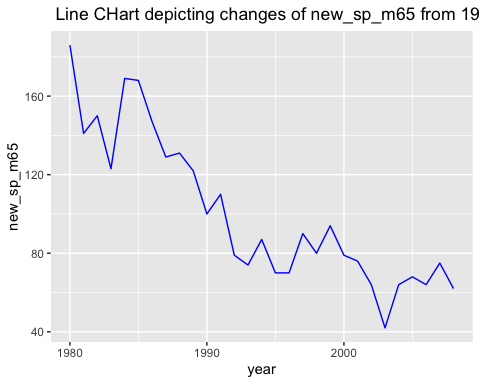
ggplot(tbCan, aes(x= year, y=new\_sp\_f014)) +  
 geom\_line(color = 'green') + ggtitle(" Line CHart depicting changes of new\_sp\_f014 from 1980 - 2013")



ggplot(tbCan, aes(x= year, y=new\_sp\_f65)) +  
 geom\_line(color = 'red') + ggtitle(" Line CHart depicting changes of new\_sp\_f65 from 1980 - 2013")



ggplot(tbCan, aes(x= year, y=new\_sp\_m65)) +  
 geom\_line(color = 'blue') + ggtitle(" Line CHart depicting changes of new\_sp\_m65 from 1980 - 2013")

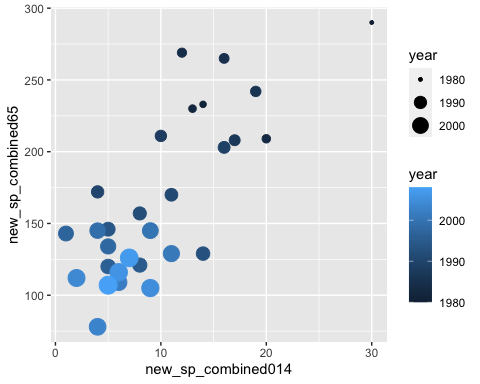


1. Using vector math, create a new variable by combining the numbers from **new\_sp\_m014** and **new\_sp\_f014**. Save the resulting vector as a new variable in the **tbCan** df called **new\_sp\_combined014**. This new variable represents the number of positive pulmonary smear tests for male AND female children between the ages of 0 and 14 years of age. Do the same for SP **tests among citizens 65 years of age and older** and save the resulting vector in the tbCan variable called **new\_sp\_combined65**.

tbCan$new\_sp\_combined014 <- tbCan$new\_sp\_m014 + tbCan$new\_sp\_f014  
tbCan$new\_sp\_combined65 <- tbCan$new\_sp\_f65 + tbCan$new\_sp\_m65

1. Finally, create a **scatter plot**, showing **new\_sp\_combined014** on the x axis, **new\_sp\_combined65** on the y axis, and having the **color and size** of the point represent **year**.

ggplot(tbCan, aes(x=new\_sp\_combined014, y=new\_sp\_combined65, color=year, size=year)) + geom\_point()



1. Interpret this visualization – what insight does it provide?

# As time passes, the number of positive cases keeps decreasing. On average, the people older than 14 and less than 65 test positive more frequently