Assignment 1: Introduction

Jack Carpenter

OVERVIEW

This exercise accompanies the lessons in Water Data Analytics on introductory material.

Directions

1. Change "Student Name" on line 3 (above) with your name.
2. Work through the steps, **creating code and output** that fulfill each instruction.
3. Be sure to **answer the questions** in this assignment document (marked with >).
4. When you have completed the assignment, **Knit** the text and code into a single PDF file.
5. After completing your assignment, fill out the assignment completion survey in Sakai.

Having trouble? See the assignment's answer key if you need a hint. Please try to complete the assignment without the key as much as possible - this is where the learning happens!

Target due date: 2022-01-18

Course Setup

1. Post the link to your forked GitHub repository below. Your repo should include one or more commits and an edited README file.

   Link:https://github.com/jbc70/Water_Data_Analytics_2022

Data Visualization Exercises

2. Set up your work session. Check your working directory, load packages `tidyverse`, `dataRetrieval`, and `zoo`. Set your ggplot theme as theme_classic (you may need to look up how to set your theme).

```
#check working directory
getwd()
```

```
## [1] "/Users/Jack/Documents/Duke/Spring 2022/Water Data Analytics/Water_Data_Analytics_2022/Assignmen
```

```
#load packages
library(tidyverse)
library(dataRetrieval)
library(zoo)

#set ggplot theme
theme_set(theme_classic())
#edit: not sure this worked
```

3. Upload discharge data for the Eno River at site 02096500 for the same dates as we studied in class (2012-01-01 through 2021-12-31). Obtain data for discharge. Rename the columns with informative titles, as we did in class.

```
#import data
EnoDischarge2 <- readNWISdv(siteNumbers = "02096500",
                            parameterCd = "00060",
                            startDate = "2012-01-01",
                            endDate = "2021-12-31")
# not sure where parameterCd, startDate, endDate commands come from,
#just copied them from the lesson named "01_Introduction"
#same with the parameter code

#rename columns
names(EnoDischarge2)[4:5] <- c("Discharge_cfs", "Approval.Code")

#this is how to see dataset attributes
#information about what is in the set, where it comes from, etc.
#variable attributes
attr(EnoDischarge2, "variableInfo")
```

```
##   variableCode           variableName              variableDescription
## 1        00060 Streamflow, ft&#179;/s Discharge, cubic feet per second
##       valueType  unit options noDataValue
## 1 Derived Value ft3/s    Mean          NA
```

```
#site attributes
attr(EnoDischarge2, "siteInfo")
```

```
##                   station_nm  site_no agency_cd timeZoneOffset
## 1 HAW RIVER AT HAW RIVER, NC 02096500      USGS         -05:00
##   timeZoneAbbreviation dec_lat_va dec_lon_va       srs siteTypeCd    hucCd
## 1                  EST   36.08722  -79.36611 EPSG:4326         ST 03030002
##   stateCd countyCd network
## 1      37    37001    NWIS
```

4. Build a plot called EnoPlot2. Use the base plot we made in class and make the following changes:

- Add a column to your data frame for discharge in meters cubed per second. hint: package dplyr in tidyverse includes a `mutate` function
- Add a column in your data frame for a 30-day rolling mean of the metric discharge. (hint: package dplyr in tidyverse includes a `mutate` function. hint: package zoo includes a `rollmean` function)
- Create two geom_line aesthetics, one for daily discharge (meters cubed per second) and one for rolling mean of discharge. Color these differently.
- Update your ggplot theme. I suggest "classic." (hint: https://ggplot2.tidyverse.org/reference/ggtheme.html)
- Update axis names
- Change the y axis from a linear to a log10 axis (hint: google "ggplot logged axis")
- Add a legend. (hint: Google "add legend two geom layers ggplot")

```
#got stuck here and had to look at the key - syntax is the hard part
#add column in m3/s
EnoDischarge2 <- EnoDischarge2 %>%
mutate(Discharge_metric = Discharge_cfs/35.3147,
       Discharge_rollmean = rollmean(Discharge_metric, 30, fill = NA))
#could not get R to recognize Discharge_cfs
#I guess the %>% makes the difference although I'm not sure why
#let's take a look if %>% worked
glimpse(EnoDischarge2)
```

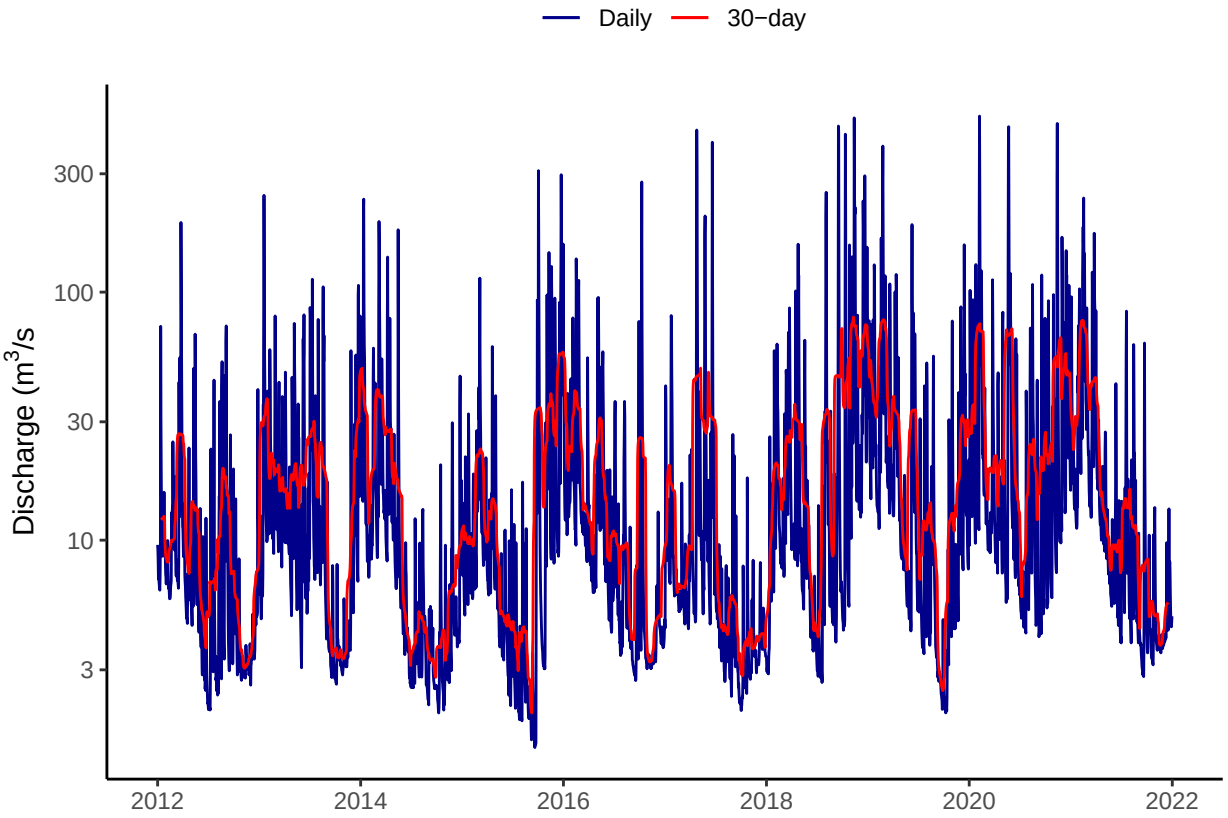
```
## Rows: 3,648
## Columns: 7
## $ agency_cd          <chr> "USGS", "USGS", "USGS", "USGS", "USGS", "USGS", "US~
## $ site_no            <chr> "02096500", "02096500", "02096500", "02096500", "02~
## $ Date               <date> 2012-01-01, 2012-01-02, 2012-01-03, 2012-01-04, 20~
## $ Discharge_cfs      <dbl> 338, 320, 288, 246, 249, 241, 232, 224, 222, 256, 3~
## $ Approval.Code      <chr> "A", "A", "A", "A", "A", "A", "A", "A", "A", "A", "~
## $ Discharge_metric   <dbl> 9.571085, 9.061382, 8.155244, 6.965938, 7.050888, 6~
## $ Discharge_rollmean <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
```

```r
#looks like it worked I think

#lets make a ggplot
EnoPlot2 <-
  ggplot(EnoDischarge2, aes(x = Date)) +
  geom_line(aes(y = Discharge_metric, color = "Daily")) +
  geom_line(aes(y = Discharge_rollmean, color = "30-day")) +
  scale_color_manual(values = c("Daily" = "darkblue", "30-day" = "red"))+
  scale_y_log10(name = expression("Discharge (m"^3*"/s")) +
  theme_classic() +
  theme(axis.title.x = element_blank(), legend.title = element_blank(),
        legend.position = "top")

#Let's see if it worked
EnoPlot2
```

```
## Warning: Removed 29 row(s) containing missing values (geom_path).
```



5. In what ways was the second plot a more effective visualization than the first?

ANSWER: The second plot has both the data and an average plotted, allowing for comparison. Color-coding also allows for the parts of the plot to be easily distinguished and thus much more useful to display information quickly and easily. The fact that the second plot is in metric arguably makes it easier to read for the majority of scientists, although since this is a US audience that point is somewhat negated.

6. What portions of the coding were challenging for you?

   ANSWER: Section 4 was really challenging. I couldn't figure out how to get the mutate function to work and ended up needing the key to figure out the syntax. The syntax on a lot of this is really the hardest part - unlike excel R doesn't helpfully tell you when you're missing an input, it just doesn't work so figuring out what I'm missing to make things work is the hard part. Some of the smaller parts I also don't quite understand, like the "fill" for the rolling mean and why "color" for the geom_lines are named and then specified as colors later. What does "fill" do and why not just name the color there and then without the extra step? The "element_blank" is also a little confusing as well - what does that mean?

7. Interpret the graph you made. What are the things you notice about within- and across-year variability, as well as the differences between daily values and 30-day rolling mean?

   ANSWER: Within each year, there is clear seasonal variability in both the dailies and rolling mean. 2017 is an erratic year - there is a large low-flow period in the middle of where the runoff spike should be. As time goes on, there appears to be a slight positive increase in both max and min discharges each year, indicating a wetter trend. There isn't a ton of interannual variability, relative to within a given year, the majority of data points appear to be within the same bounds as each other with that wetting trend. Clearly late 2015, late 2019, and late 2021 are exceptionally dry points.