Quant Methods: Programming Basics 3 - Reading datafiles, scripting (creating programs or codes)

Reading data into R

First, we need to think about where we are in our directory system and where the data file to read is

getwd() tells you your working directory

setwd() changes your working directory, e.g., setwd("./ProgrammingBasics")

"Files" tab on lower-right shows files in current working directory. The file USGSMissRDischargeAug2021.csv is on Collab under Resources/Datafiles. You should download it and put it in your working directory for R (class folder).

csv files are "comma separated values" files. Excel can read (and write) them as can R.

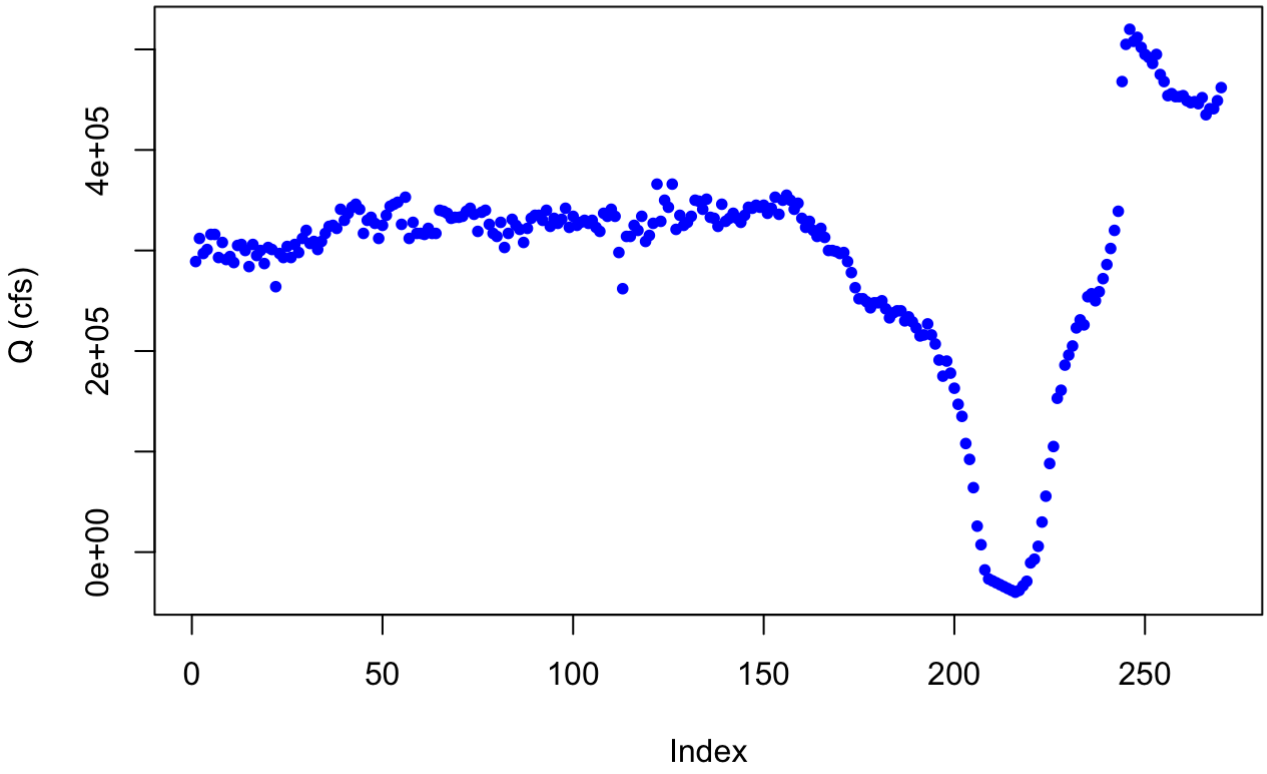
```
Qdf = read.csv(file="USGSMissRDischargeAug2021.csv",sep=",",header=TRUE)
```

When you read data into R from a csv file, it reads into something called a data frame. In this case, Qdf is the name of the data frame. We can refer to each variable (column) in Qdf as, e.g. Qdf$Qcfs (Click blue arrow to left of name in environment tab to see what is in the data frame).

This data is from the USGS waterdata site that provides streamflow data for many sites across the US. This particular dataset is for the Mississippi River at the time that Hurricane Ida approached the coast of Louisiana last week.

Plot Qcfs (discharge in cfs)

```
plot(Qdf$Qcfs, pch=20, col="blue", xlab="Index", ylab="Q (cfs) ")
```



What do you notice?

Q < 0!! Really remarkable

Maybe some gaps in record?

Index is less useful than a time value on x axis

cfs is not an SI unit. Convert to m3/s

There is a DateTime column in the data frame. Can we use that?

Fixes? Create time vector, convert Qcfs to Convert to m3/s, perhaps identify missing values, convert DateTime to a useful form (now character)

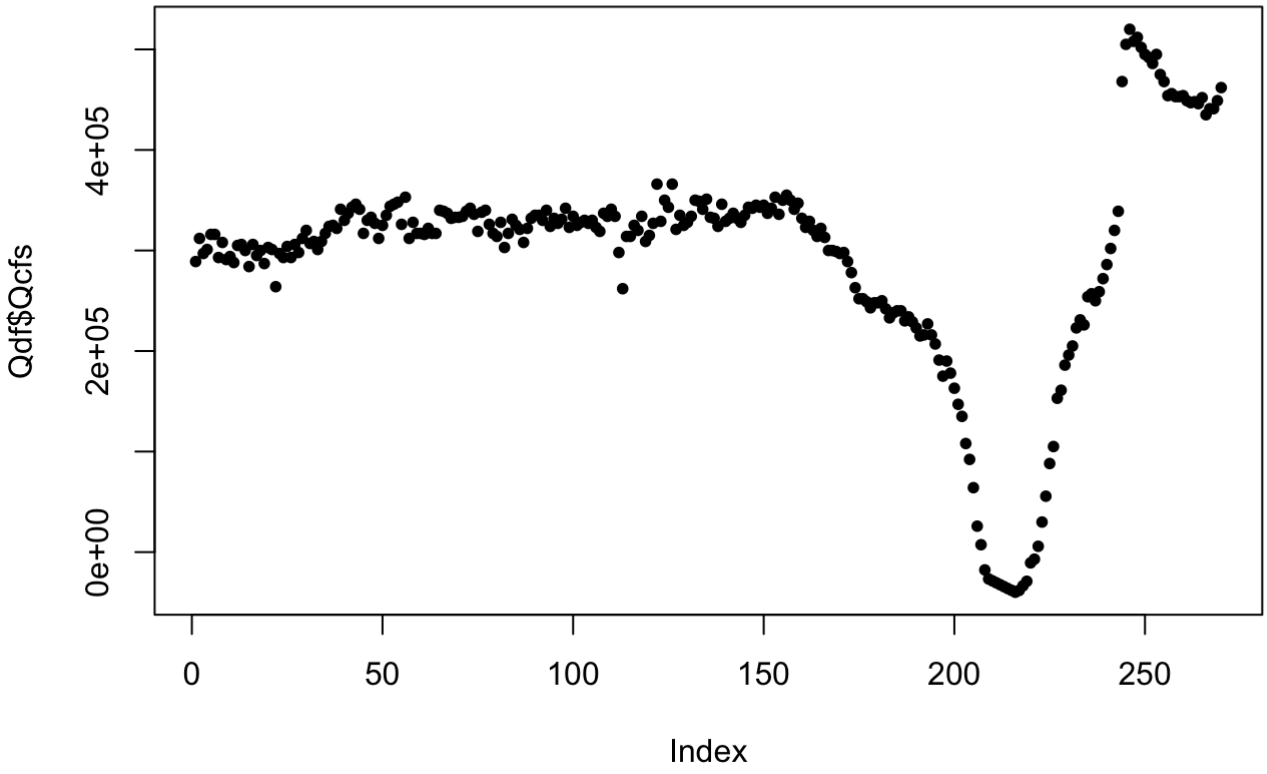
We could continue to work on this in a notebook, but I'd like you to create a code (or script) to do it.

Scripting in R

Enable script window in R Studio (upper left).

We'll type in our code there [repeated below]

```
# Read and plot USGS discharge data
Qdf = read.csv("USGSMissRDischargeAug2021.csv")
plot(Qdf$Qcfs, pch=20) #plot values by index number
```



Save, giving it a name, e.g. ReadUSGS.R

To run, click Source button on upper right of source editor or highlight all text and click run or press Ctrl+Shift+Enter

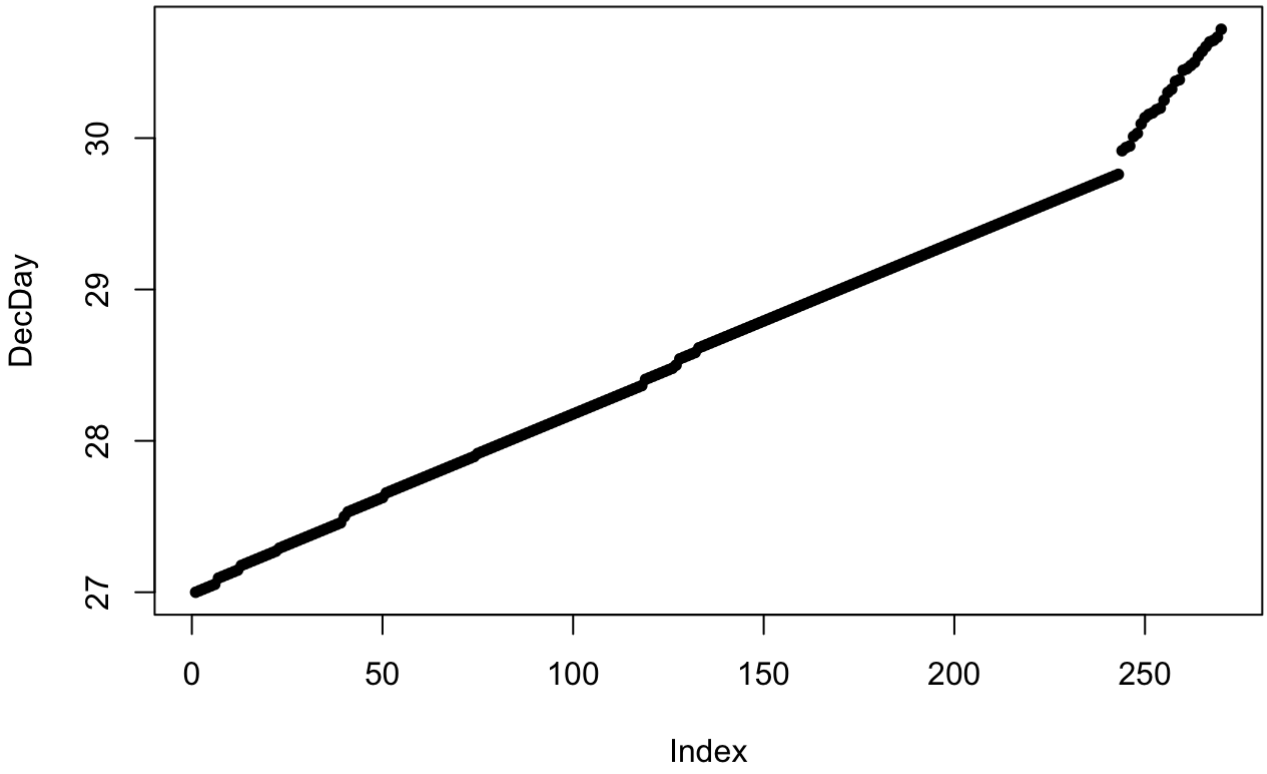
If you look at the list of files in your working directory you should see ReadUSGS.R. Double clicking on it in your directory should cause it to open in RStudio.

Now let's add to it to make it more informative. First, create a time vector. How might we do that?

We have the day, hour and minute. Can use those to create a decimal day.

```
#Create time vector
DecHr = Qdf$Hour + Qdf$Min/60  #decimal hour
DecDay = Qdf$Day + DecHr/24   #decimal day

plot(DecDay, pch=20)
```



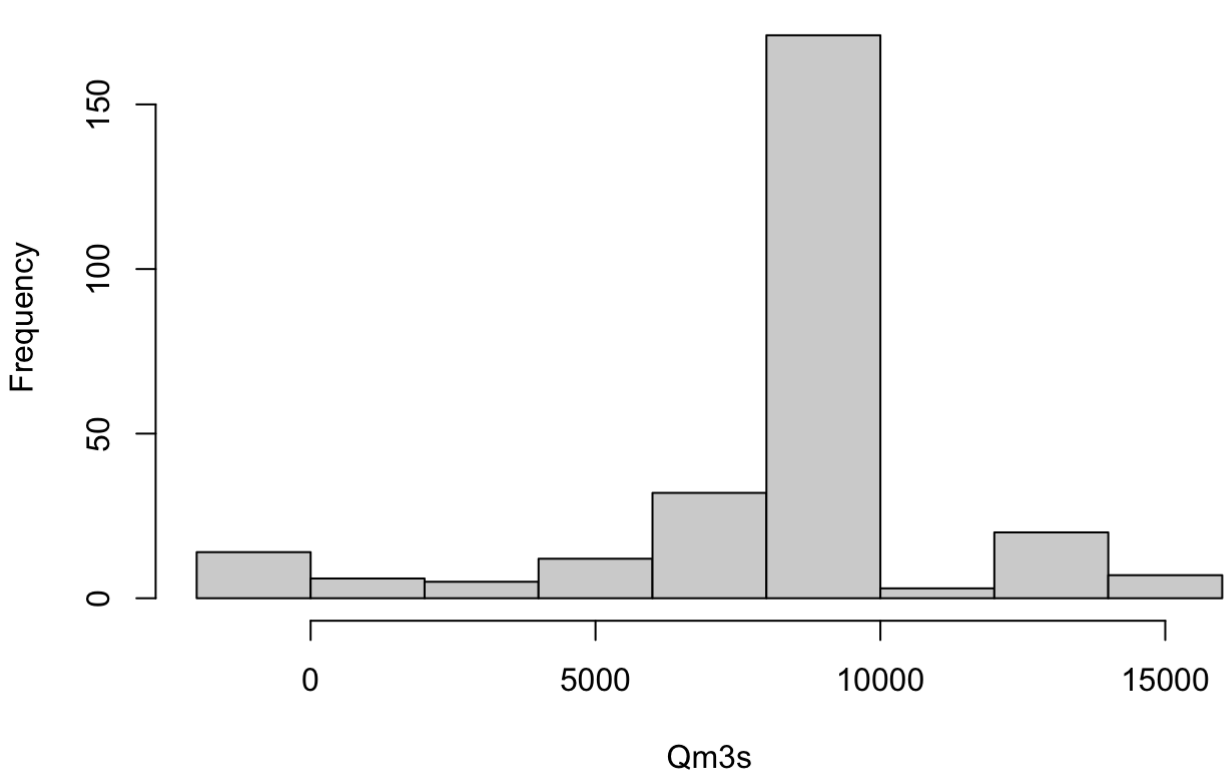
Changes in slope and wiggles are indicative of missing times. Click the box on the right of Qdf in the environment tab. It should bring up the file in the top-left window.

Scroll down a bit. Values should consistently increment by 15 mins, but some are clearly missing. Does it matter? How could we fix it?

Convert cfs to m3/s. Then look at the distribution of Qm3s

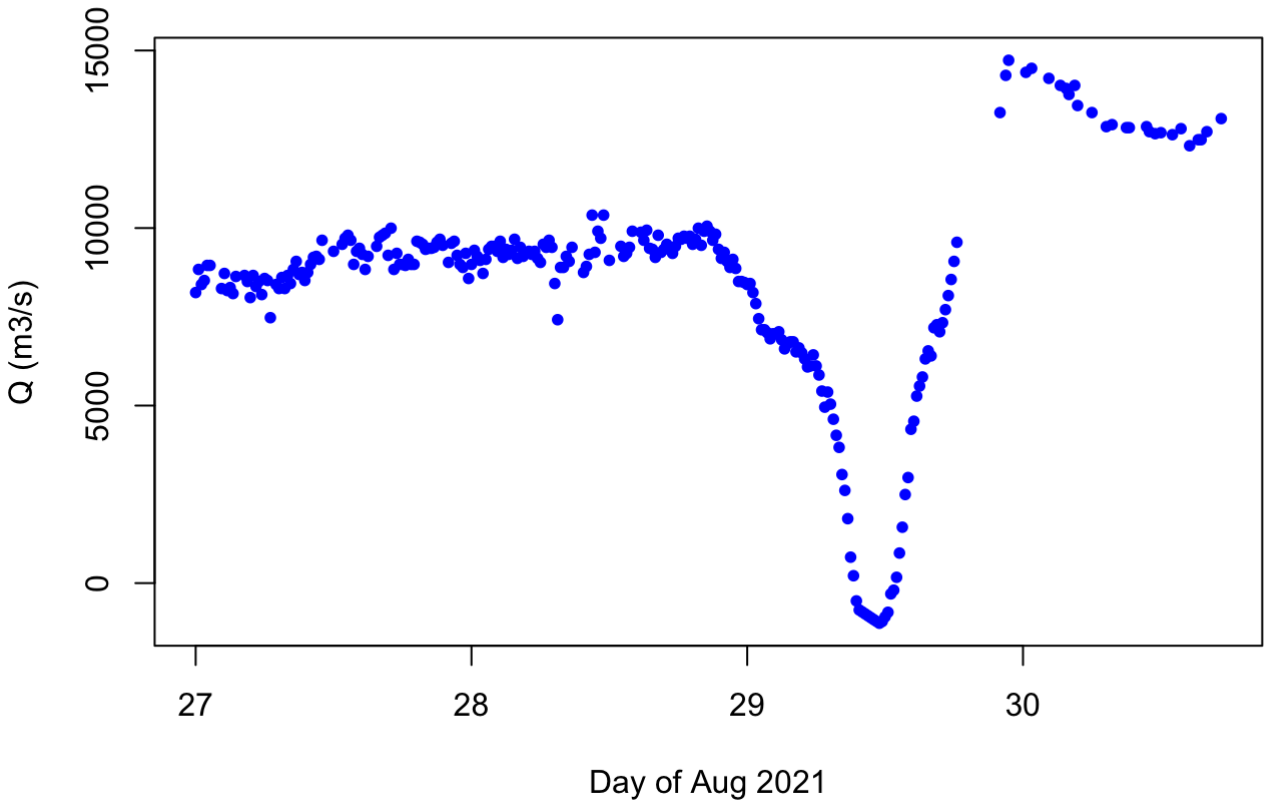
```
Qm3s = Qdf$Qcfs * (0.3048^3)   #convert units to SI

hist(Qm3s)
```



Now let's plot Q vs time

```
plot(DecDay, Qm3s, pch=20, col="blue", xlab="Day of Aug 2021", ylab="Q (m3/s)")
```



Creating a date-time vector

The easiest way to do this (which we will explore more deeply in a few weeks) is using the package lubridate. Install it from the packages tab.

We can use a command of the form mdy_hm() to convert Qdf$DateTime to a date-time class vector.

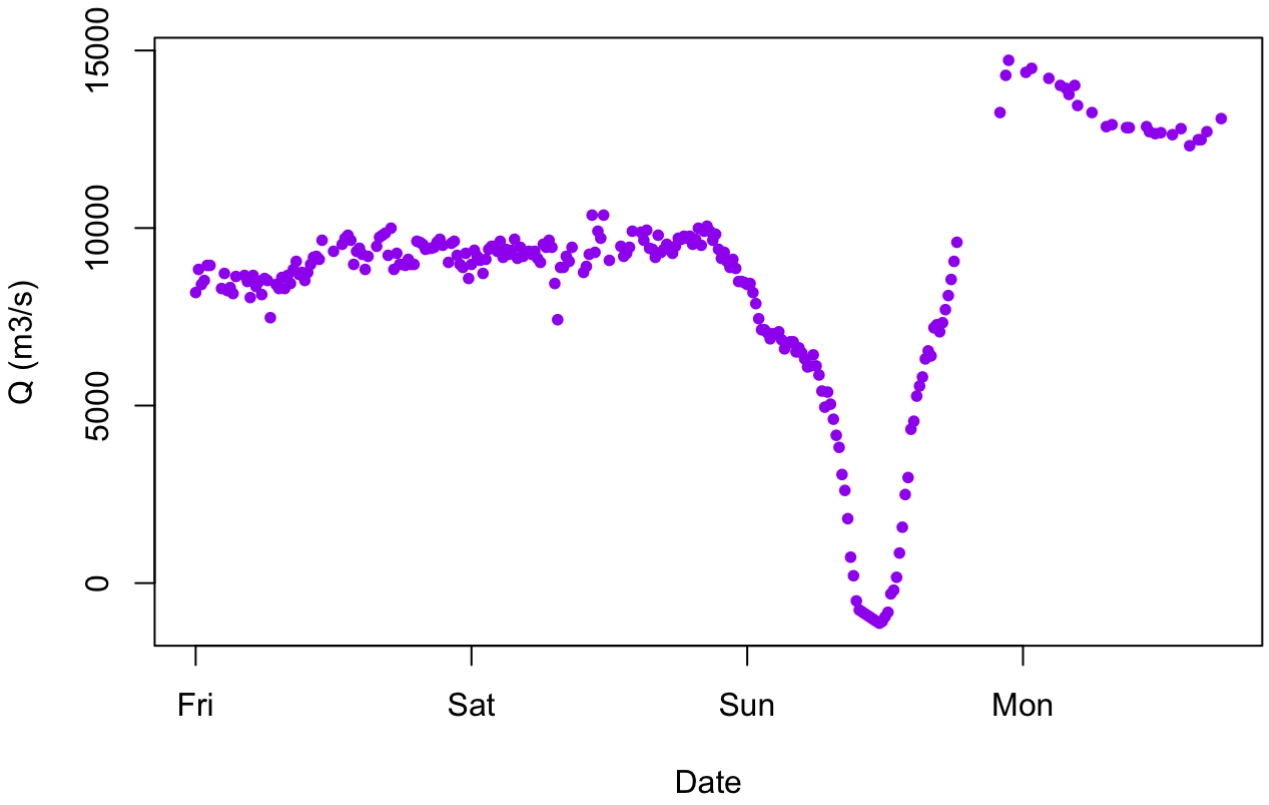
```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
DateTime = mdy_hm(Qdf$DateTime)
```

```
plot(DateTime, Qm3s, pch=20, col="purple", xlab="Date", ylab="Q (m3/s)")
```



Once we have dates in this form, we can create a sequence of them using seq.POSIXt()

```
tbeg = DateTime[1]
tend = DateTime[length(DateTime)]
tseq = seq.POSIXt(tbeg, tend, by = "15 min")
length(tseq)
```

```
## [1] 358
```

```
length(DateTime)
```

```
## [1] 270
```

We can see from the difference in lengths that a number of times are missing in the USGS dataset. Does it matter? Depends on what you want to do with the data. We'll talk about this more in a few weeks.

Assignment 1 The first assignment (available in Collab under Resources/Assignments) is due a week from Friday.