OS4118 **API and XML** Fall AY 2020

**Introduction:** In this exercise we will extract a little information from a government-operated API that returns XML. We will then turn the XML into something useful and draw a nice picture.

**The API:** The API is the “Lock Performance Monitoring System” operated by the Army Corps of Engineers. The API documentation itself can be found at this address:

[https://corpslocks.usace.army.mil//lpwb/f?p=121:7:0::NO:::](https://corpslocks.usace.army.mil/lpwb/f?p=121:7:0::NO:::)

I am interested in the tonnage passing through the “Chain of Rocks Lock and Dam 27,” which is located near Granite City, IL, and which is the southernmost lock on the Mississippi River. This data apparently goes back only to January of 2018, so find the tonnage at that lock for all the months from January 2018 through October 2019. Then draw a picture that helps us understand what’s going on – you might compare products to one another, or upbound to downbound traffic, or anything else that interests you.

**Technical Notes:** I recommend the httr package and its GET() function to do the data acquisition. You might write a little loop that, for each month, generates the proper query string and then pass that to GET(). The return from GET()is a special type of object with text inside it. So first we need to extract the text with the content() comment. The result of that will be XML, so you will also want the XML library. Now, in general, XML can be non-rectangular, list-within-list data, but in this case all of the XML entries are of the same sort. So we can use the useful xmlToDataFrame() function. In short, one call to the API might look something like this:

1.) Prepare the query string, by pasting the link, then a "?", then name=value pairs separated by "&".

2.) Call GET() with that query string. Save the result. Let’s call that thing res.

3.) Now xmlToDataFrame (content (thing, "text")) should produce a data frame. Be aware that the columns of that data frame which look numeric will in fact be factors. You convert a factor variable to numeric with a command like

mycolumn <- as.numeric (as.character (mycolumn))

Never use as.numeric() directly on a factor variable unless you understand what you’re doing – it produces the factor’s levels as 1, 2, 3 and so on.