**R Workshop 31st January 2024 – Worksheet 2**

**Exe River Flow Analysis**

Introduction and Reading in Data

Now you know some of the basics of R, we will now start analysing some data. To begin with, we will look at plotting up and analysing some flow data from the river Exe, specifically gauged daily flow at Trews Wier.

The code is shown in text boxes here which can be copied and pasted into R or RStudio. Alternatively, a notebook version of this activity exists as will have been explained in the workshop. Suggested changes that you may want to do to the code are shown in **bold**. However please do also play around with your own ideas too!

The data is provided as a .csv file at: [https://nrfa.ceh.ac.uk/data/station/info/45007](https://colab.research.google.com/corgiredirector?site=https%3A%2F%2Fnrfa.ceh.ac.uk%2Fdata%2Fstation%2Finfo%2F45007). Information on the dataset can be found here, with a link to download the .csv file on the 'Daily flow data' tab, and then by clicking 'Download flow data'. It will be called '45007\_gdf.csv'.

Opening this .csv file in Excel, you can see it has a lot of information at the top, and then two columns for the data, showing the date and the flow. We will need to put this in our current directory to use it, which can be found using the *getwd()* function.

To read in the file, we will use the *read.csv()* function, skipping the first 20 lines which has all the information in it at the top (skip=20) and we will tell R that there are no headers (header=F, for 'FALSE'), so it will not look to put names at the top of the columns as there are none in the .csv file. Finally we will use the *head()* function to check the top of the data that we have loaded in, to check that everything looks good and as expected!

riv\_table <- read.csv('45007\_gdf.csv', skip=20, header=F)

head(riv\_table)

Everything should look good. We can make a data frame of this table, which allows R to understand the dates in the left column as actual dates (currently they are 'strings'), and we can give the columns names to help with understanding data quickly.

For this we use the *data.frame()* function, creating columns called 'date' and 'flow' and assigning them the first column of the table with the *as.Date()* function to turn those strings into dates, and the second column for the flow rates. Checking how it looks with the *head()* function again you can see that we now have headings for the columns and the 'class' of the first column has changed to 'date'. Now we can plot...

riv\_data <- data.frame('date'=as.Date(riv\_table[,1]), 'flow'=riv\_table[,2])

head(riv\_data)

Plotting

We can plot the data in the most simple way now using *plot()* like below:

plot(riv\_data)

You can see that the dates have worked correctly, and the axis have been labelled to the names of the columns. However, we can make this tidier, and make a line plot rather than using points as is the default in R.

plot(riv\_data, type='l', main='Daily flow of Exe', xlab='Year', ylab='Flow (m^3/s)')

As we have learned from the basics of R, using type='l' in the *plot()* function makes it plot a line, and we have added a main, x axis label and y axis label that looks better.

**Try and change the colour and/or style of the line by updating the code in the block above and rerunning it.**

Monthly Averages

Above we have looked at the daily time series, which is quite noisy or jumpy. It could be clearer to see any trends in the data by looking at the mean daily flow values over each month instead. To do this we will use a *for loop* that loops over each individual year and month string from the date column (e.g. 1996-11 or 2008-07) and takes an average of the flow values that have dates that start with year and month combination.

To begin with, we need to determine the different months to loop over. For this, is it probably easier to go back to the string from the original table rather than the dates objects as these strings are easier to work with.

We use the *substr()* function which takes in a string or vector of strings and two numbers and returns the characters in the strings that are between the positions. We want the characters in positions 1 to 7 (XXXX-XX, remember the dash is a character too). Then, for our *for loop* we do not want repeats, we only want to unique values that are returned:

months <- unique(substr(riv\_table[,1], 1, 7))

print(months)

You will see what the combination of *unique()* and *substr()* has done from the output above. However, notice that there is an 'M' in the 6th position. Looking back at the original .csv file or in the riv\_table object here, you will see that appears to be linked to where there is missing data. We will remove this for our *for loop*, otherwise we will get missing data in the middle of our monthly time series when we actually want what is in April 1997 after March 1997. We can remove this like we have learned in the basics:

months\_fixed <- months[-6]

print(months\_fixed)

Now we have this set of months, we can make a *for loop* to create these averages. There are 3 steps to this:

1) Create a vector to store the monthly values in that is the same length as the number of months we have.

2) Within the *for loop*, check which substring of the riv\_table date strings have the year and month combination we want which a which statement

3) Take the mean of the flows (column 2 in the riv\_table object) on those rows we have just calculated and store it in our object we made above. Note that *na.rm=T* (TRUE) will ignore any missing data, which would mess up our average.

mon\_flow <- rep(NA, length(months\_fixed))

for (i in 1:length(months\_fixed)) {

month\_inds <- which(substr(riv\_table[,1], 1, 7) == months\_fixed[i])

mon\_flow[i] <- mean(riv\_table[month\_inds,2], na.rm=T)

}

Now we have this object, we can create a data frame like before, of these monthly means:

month4plot <- as.Date(paste(months\_fixed,'-15',sep=''), format='%Y-%m-%d') #you can set your date up like this using the as.Date() function for the 15th of each month, pasting '-15' to the end of each of the unique month strings

riv\_mon\_data <- data.frame('date'=month4plot, 'flow'=mon\_flow)

And plot as before:

plot(riv\_mon\_data, type='l', main='Monthly mean daily flow of Exe', xlab='Year', ylab='Flow (m^3/s)')

**How would you add the original daily data on in a different colour on top of this? (Consider the *lines()* function and the ylim of the plot! ylim is actually used further down so you may want to come back to this afterwards)**

**Creating a Mean Annual Cycle**

In some cases we may be interested in what the average year looks like, or the mean annual cycle. This is achieved by taking the mean across the whole dataset for all Januaries, Februaries, Marches etc. We already know what the unique month values look like, it is easier than the year-month combinations from above:

month\_labs <- c('01','02','03','04','05','06','07','08','09',10:12)

print(month\_labs)

Note we couldn't just do *month\_labs <- c(1:12)* as we are looking specifically for '01' and '02', etc. Now we do a *for loop* very similar to last time, except we are looking for the *substr* values for 6 to 7 in the dates this time, rather than 1 to 7. We will also calculate the standard deviation alongside the mean, otherwise this for loop should look very familiar:

month\_avg <- rep(NA, 12)

month\_sd <- rep(NA, 12)

for (i in 1:12) {

month\_inds <- which(substr(riv\_table[,1], 6, 7) == month\_labs[i])

month\_avg[i] <- mean(riv\_table[month\_inds,2], na.rm=T)

month\_sd[i] <- sd(riv\_table[month\_inds,2], na.rm=T)

}

print(month\_avg)

print(month\_sd)

The printed outputs at the end show the 12 mean values and then the 12 standard deviations. It would be clearer to plot these of course, which we will do with specific labels on the x-axis:

monthlabs4plot <- c('Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec') #plotting labels for x axis

plot(month\_avg, type='b', main='Mean annual cycle of daily flow of Exe', xlab='Month', ylab='Flow (m^3/s)', xaxt='n', ylim=range(c(month\_avg+month\_sd,month\_avg-month\_sd))) #plot sets the y axis range to fit in the mean +/- sd

lines(month\_avg+month\_sd, type='b', lty=3) #dotted lines (lty=3) are added for the +/- sd

lines(month\_avg-month\_sd, type='b', lty=3)

axis(1, at=c(1:12), labels=monthlabs4plot) #make a nice axis

Note that we changed the y-axis range so it fits the +/- sd lines in. The argument *type='b'* plots both lines and points. **Try and change the style of the points using the pch argument and add a legend with the *legend()* function. You may have to google these for help!**

**Box and Whisker Plots**

You may want to present this data in a different way, with a box and whisker plot for example. To do this, we need to append the month onto our riv\_data data frame using the dollar sign notation. The below creates a new column called 'month' to the data frame and assigns one of the labels we used for plotting above, based on the '01', '02' etc part of the date. The *head()* line shows the top of this (which all coincides with November).

Then, mainly because R needs explicitly telling these things, we need to turn these labels into 'factors' which in reality just tells R to plot the box plot in the correct year order (else it would plot the months alphabetically). For the second *head()* output you can see that all that has changed is now the third column has 'fct' for factor above it.

Finally, we create the box plot with the *boxplot()* function which works similarly to the normal *plot()* function. Here we are telling R to make boxes based on the flow, separated by month.

riv\_data$month <- monthlabs4plot[as.numeric(substr(riv\_data$date, 6, 7))] #add a new columnn to the original data frame that uses the labels from above and looks at which month in the in the date string in each line

head(riv\_data)

riv\_data$month <- factor(riv\_data$month, levels=c('Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec')) #this needs to be done to make sure it plots in the correct order

head(riv\_data)

boxplot(flow ~ month, data=riv\_data, main='Distribution of daily flow of Exe per month', xlab='Month', ylab='Flow (m^3/s)') #this makes the boxplot of flow values based on month in the dataset

You will notice a number of points above the whiskers which are considered by R as outliers (they lie outside a certain range from the mean values to be considered as part of the whiskers). At the start, it was noted that the daily data was very noisy which is what causes this. We could fix this by using the monthly data frame we created earlier. **Alter the monthly data frame we made and make another box plot which will not be as noisy. Your code should look similar to the one above.**

**Other Statistics**

There are other statistics that can be calculated on the data. Using some of the skills from the R basics we learned earlier:

riv\_data$date[which.min(riv\_data$flow)] #finds the date at the index where the minimum flow is

**Adapt the above code block to determine:**

**1) The date of the maximum flow**

**2) The month with the minimum average flow**

**3) The month with the maximum average flow**