Introduction to hydrographs and chemographs

The goal of this lab is to introduce you to the variability of flow and concentrations in nature, and introduce you to concentration values which can be found in watersheds, and to the concepts of flux calculations.

Hydrological Time Series

Simple hydrograph

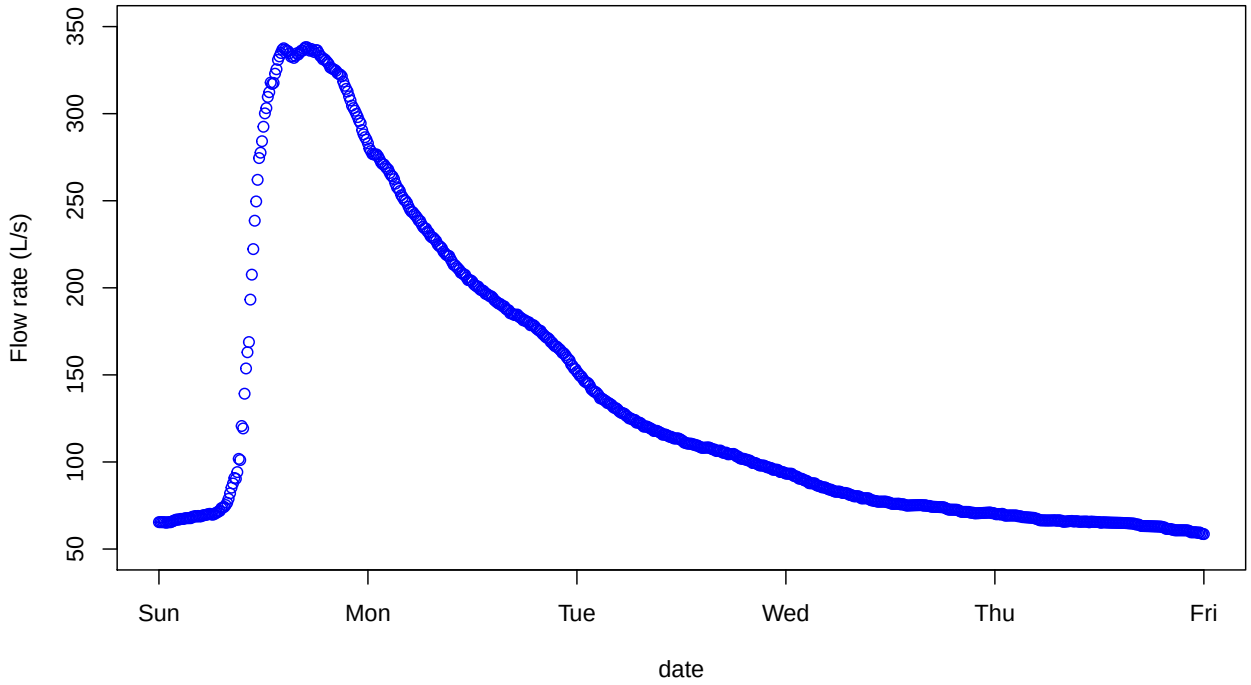
In hydrology, we work with time series of flow rates and concentrations, and many of the conclusions we make are based on the calculations of water and nutrient fluxes. The title of this paragraph is time series. Indeed, in hydrology we measure flow not on a continuous basis, but rather at a given frequency. In hydrology, we have been able to make measurements at a high frequency (hourly or smaller), for over 100 years. From this high frequency data, it is possible to obtain a visual representation of the variation of flow.

The first example below is an example of what we refer to as a simple hydrograph, which follows a rainfall event, with an initial baseflow, a rapidly rising limb, a flow peak, and a more slowly falling limb.

```
hydgph<-read.csv(file = "https://raw.githubusercontent.com/francoisbirgand/discharge_and_flux_calculati
names(hydgph)=c("date","Q","NO3")
date=as.POSIXct(hydgph$date, format = "%Y-%m-%d %H:%M:%S");Q=hydgph$Q*1000
```

```
## Warning in strptime(x, format, tz = tz): unknown timezone 'zone/tz/2018i.
## 1.0/zoneinfo/America/New_York'
```

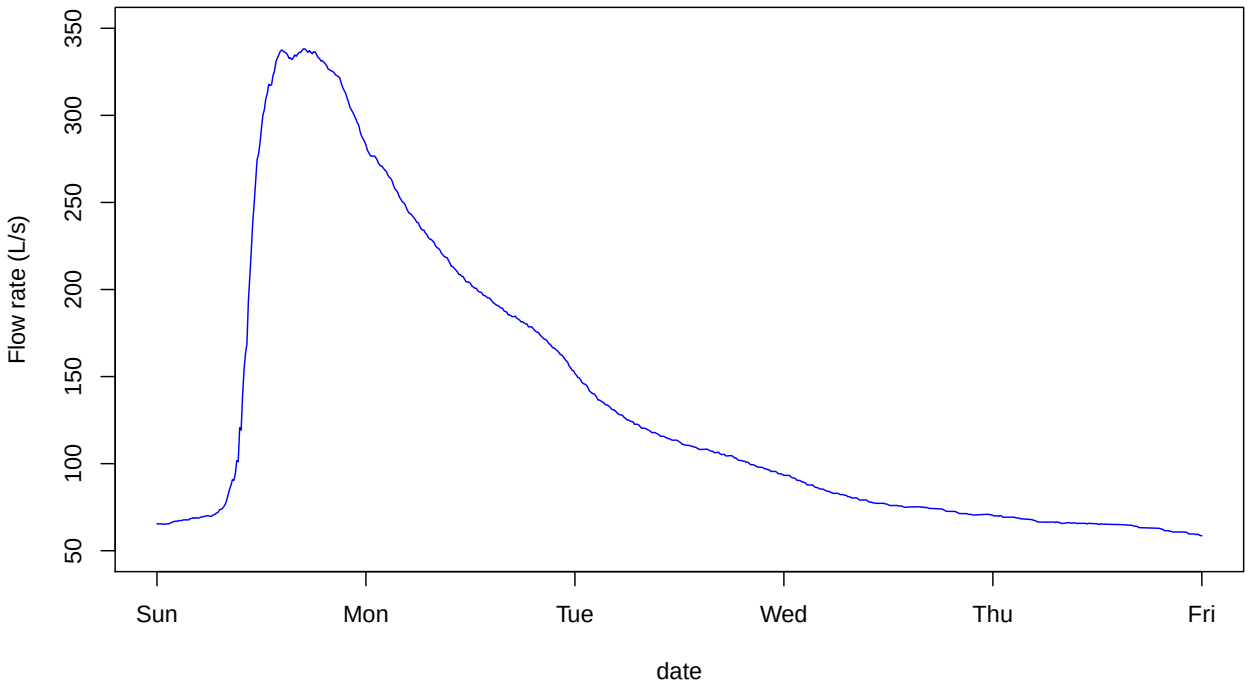
```
par(mar=c(4.5,4.5,0.5,0.5))
xlimHG=as.POSIXct(c("1999-01-03 00:00:00","1999-01-08 00:00:00"));ylimHG=c(50,350)
plot(date,Q,xlab = "date",ylab = "Flow rate (L/s)",type = "p",col="blue",xlim=xlimHG,ylim=ylimHG)
```



Although the points are disjointed, it is very tempting to add a line between them, as our observations and intuitions tell us that there is a pattern of flow up or down, and in this example, the time interval between consecutive values is 600 seconds or 10 min. And indeed, this is exactly what people do, we add lines between points as an approximation of what flow must have looked like during the measurement intervals. The same

data plotted without the measurement points looks like the figure below, appears *continuous*, although it is not!

```
par(mar=c(4.5,4.5,0.5,0.5))
xlimHG=as.POSIXct(c("1999-01-03 00:00:00","1999-01-08 00:00:00"));ylimHG=c(50,350)
plot(date,Q,xlab = "date",ylab = "Flow rate (L/s)",type = "l",col="blue",xlim=xlimHG,ylim=ylimHG)
```



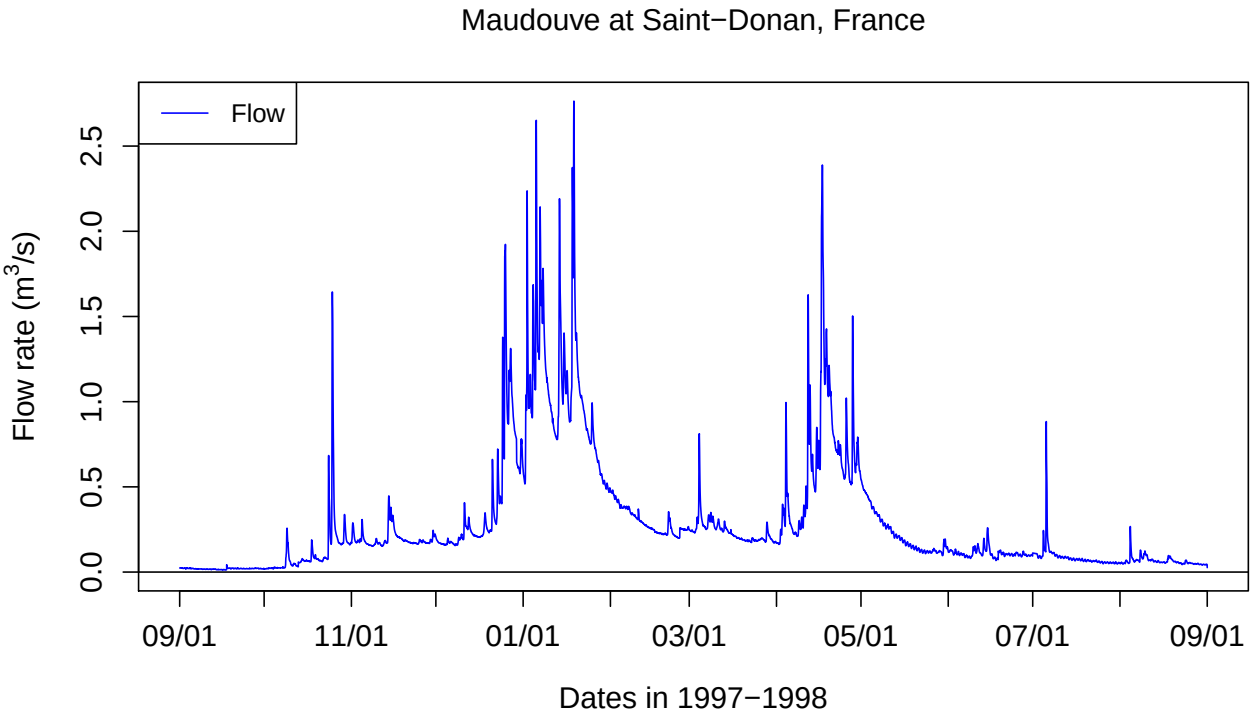
Actual hydrograph over an entire year

The goal of the first part above was to realize that there is no such thing as *continuous data*, but that really all data around the world is an assemblage of discountinuous data points. In the case of hydrological data, all points are autocorrelated, and provided that flow be measured frequently enough, then a linear interpolation between consecutive points is just fine.

Now, let us explore what a typical yearly hydrograph of a watershed in a temperate climate where snow does not play a significant role. The conventional acronym for flow rates is the letter Q.

```
data<-read.csv(file="https://raw.githubusercontent.com/francoisbirgand/francoisbirgand.github.io/master/

WSarea<-24.2 #Area of watershed in km2
WS<-"Maudouve at Saint-Donan, France"
names(data)=c("datetime","Q","C")    # renames the columns in simpler names
data<-as.data.frame(data)
data$datetime<-as.POSIXct(strptime(data$datetime, "%Y-%m-%d %H:%M:%S")) # transforms characters into da
D<-data$datetime
Q<-data$Q   #Defines Q as the flow value (m3/s)

N=nrow(data)   #Sets N to the value equal to the number of total rows in the table

# definition of the x and y axes limits
startdate<-D[1]
enddate<-D[N]
xlim = as.POSIXct(c(startdate,enddate))  # this renders the first and last date understandable for plot
```

```r
ylimQ = c(0,max(Q))            # ylim for flow

ScaleF = 1.2                   # scaling factor for size of fonts and other things

y1lab<-expression("Flow rate (" * m^3 * "/s)")  # defines the label for flow

par(mar=c(4.5,4.5,4,4.5))      # defines the sizes, in number of lines, for the margins (bottom, left, t

    ltyp=c(1,2)

    plot(D,Q,col="blue",type="l",cex=0.1,yaxt="n",
         lty=ltyp[1],xaxt="n",xlab="",ylab="",xlim=xlim,ylim=ylimQ)
    # we are taking all the default addition of axis tick marks and numbers out by using xaxt and yaxt
    # and setting the axis labels at nothing using xlab = "" and ylab = ""
    abline(h=0)
    axis.POSIXct(1, at=seq(startdate, enddate, by="month"), format="%m/%d",cex.axis=ScaleF)
    # this tells R that we want the X axis ticks and values to be displayed as dates, be added on a mon
    # using the month/day format
    axis(2,cex.axis=ScaleF)
    # this tells R that the first Y axis ticks can be displayed  (that function was repressed earlier b
    par(new=TRUE)
    # this tells R that a new plot has already been opened, in other words you are telling R to keep ad
    # on the existing plot

    mtext("Dates in 1997-1998",side=1,line=3,cex=ScaleF) # add in the margin the defined labels and tit
    mtext(y1lab,side=2,line=3,cex=ScaleF)
    mtext(WS,side=3,line=1.5,cex=ScaleF)

    legend("topleft",c("Flow"),lty = c(1), col = c("blue"))
```



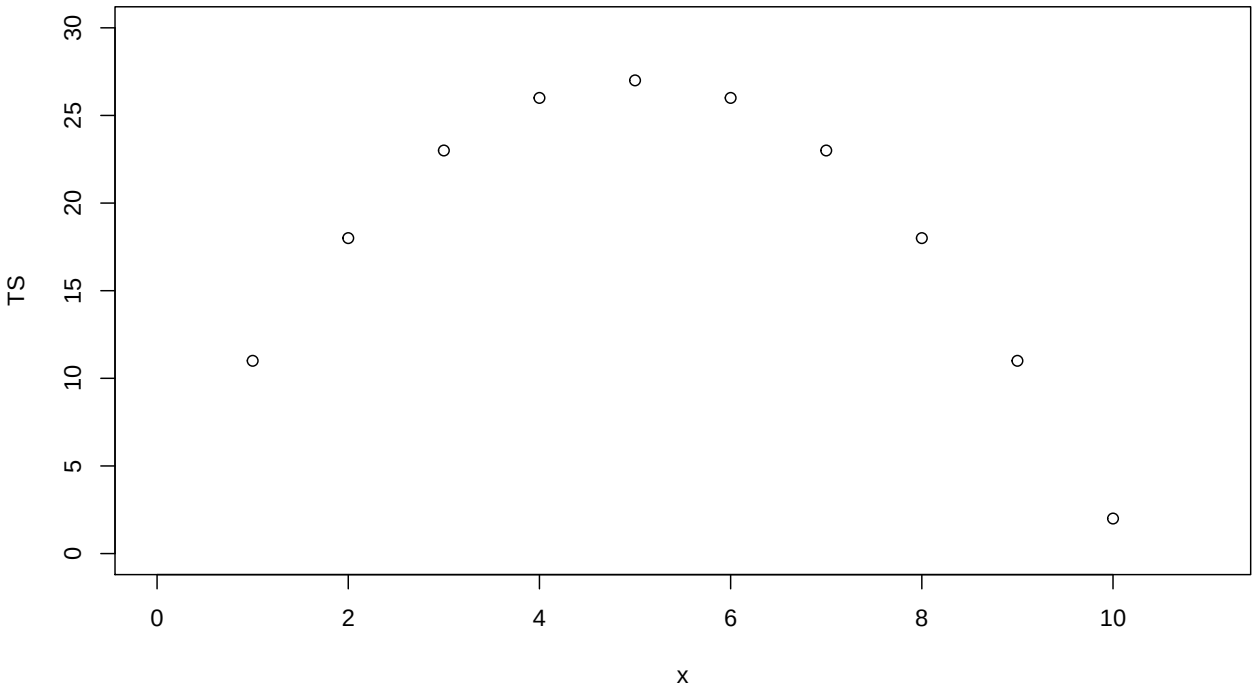
Exercise 1: calculate very simple statistics of flow

Calculate the lowest, highest, mean, and median flow rates, as well as the 10^{th} percentile, and the 90^{th} percentile. For this, there is the `quantile` function, which works as `quantile(Q,percentile value)`.

Calculate water fluxes

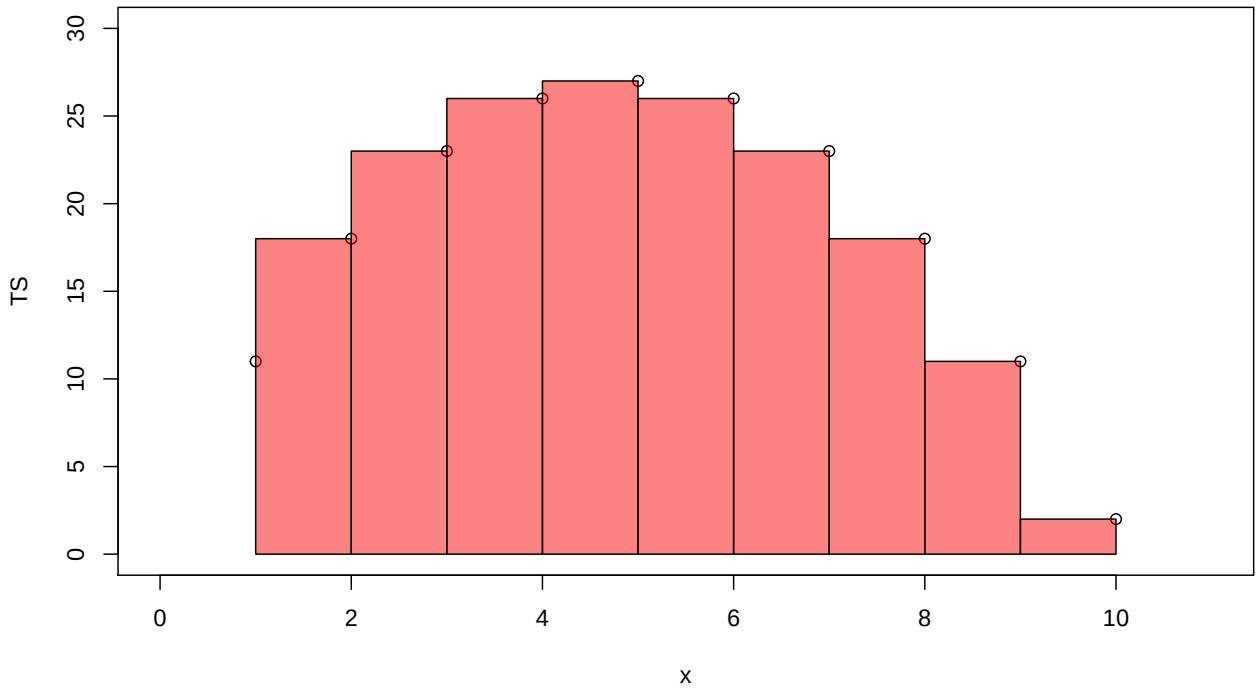
To make this more understandable, let us take a mock simple hydrograph, as plotted below.

```
##  [1] "cumTS: 11"  "cumTS: 29"  "cumTS: 52"  "cumTS: 78"  "cumTS: 105"
##  [6] "cumTS: 131" "cumTS: 154" "cumTS: 172" "cumTS: 183" "cumTS: 185"
```



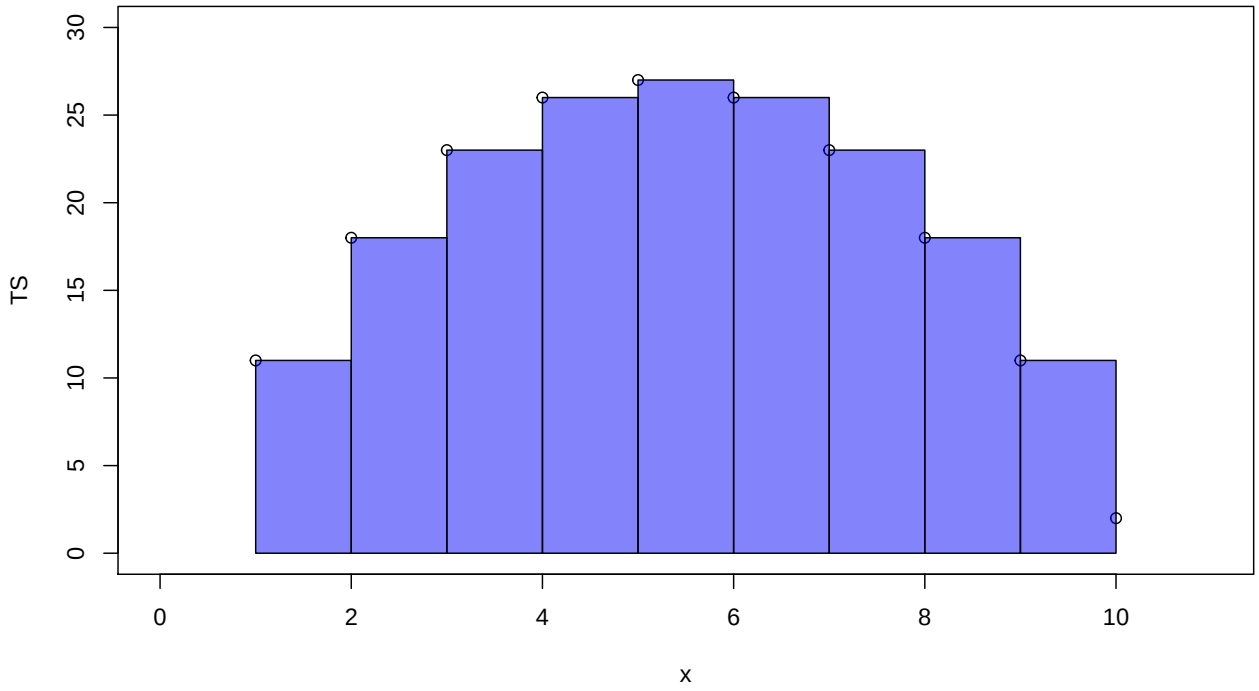
Now, let us imagine that *TS* (for Time Series) values actually correspond to 1-sec instantaneous flow values in *L/s*, with the first and last values corresponding to the initial and final time over which we wish to calculate the cumulative volume. The flow volume corresponds to the the area under the curve. There are several ways of calculating the area or the flow. The first simple method is to rectangles like in the figure below.

```
suprectxy=cbind(1:9,rep(0,9),2:10,TS[2:10])
infrectxy=cbind(1:9,rep(0,9),2:10,TS[1:9])
par(mar=c(4.5,4.5,0.5,0.5))
plot(x,TS,xlim=xlim,ylim=ylim)
transpred <- rgb(250, 0, 0, max = 255, alpha = 125)
for (i in 1:9){rect(suprectxy[i,1],suprectxy[i,2],suprectxy[i,3],suprectxy[i,4],col = transpred)}
```



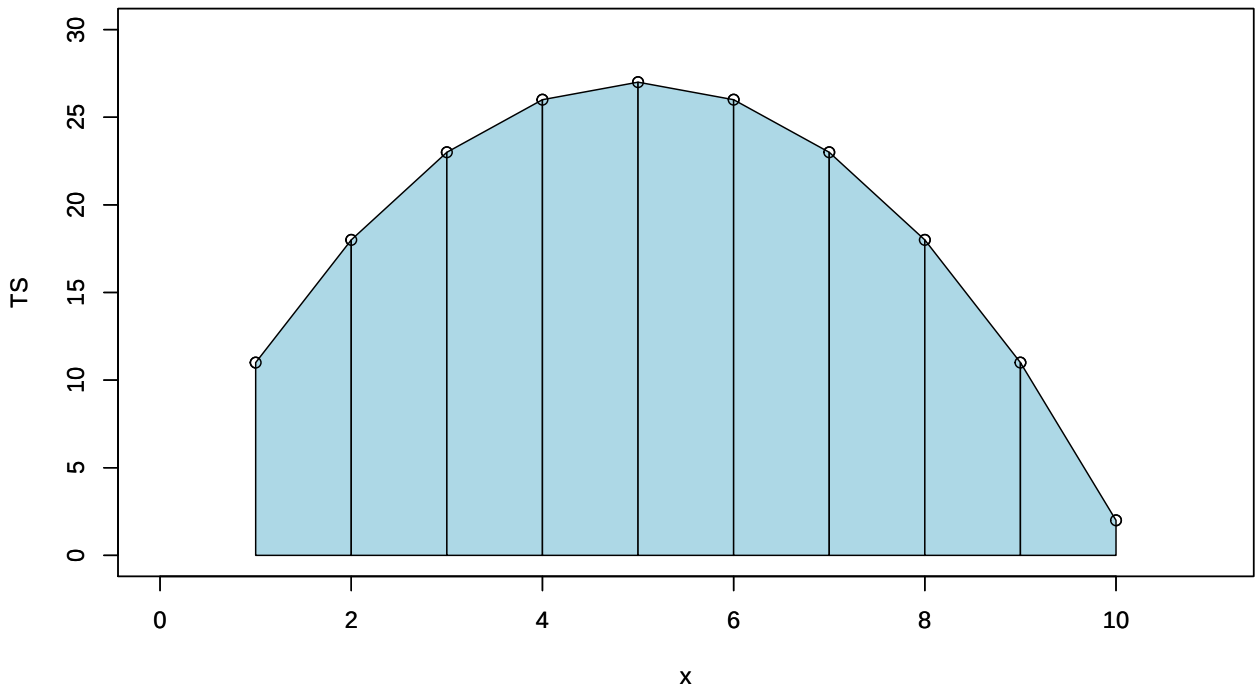
In this case, we essentially make the hypothesis that flow remains constant over the interval, and in this case, we make the hypothesis that the correct flow value to be used is the one at the end of the interval. Obviously, another way to calculate would be to use the initial value of the interval as the value to be used over the interval like in the figure below:

```
par(mar=c(4.5,4.5,0.5,0.5))
plot(x,TS,xlim=xlim,ylim=ylim)
transpblue <- rgb(0, 0, 250, max = 255, alpha = 125)
for (i in 1:9){rect(infrectxy[i,1],infrectxy[i,2],infrectxy[i,3],infrectxy[i,4],col = transpblue)}
```



Still this is not very satisfying for the mind because constant flow over an inteval is just not realistic. What we would rather have is something that would use the trapeze method like in the example below.

```
par(mar=c(4.5,4.5,0.5,0.5))       # this to split the plot layout into two columns
plot(x,TS,xlim=xlim,ylim=ylim)
date<-as.character(date)                  # need to transfer dates back into character to make some oper
polygx<-cbind(head(x,-1),head(x,-1),x[-1],x[-1],head(x,-1))         # x values of the polygons
polygy<-cbind(rep(0,length(TS)-1),head(TS,-1),TS[-1],rep(0,length(TS)-1),rep(0,length(TS)-1))   # y val
for (j in 1:(length(TS)-1)){polygon((as.vector(polygx[j,])),as.vector(polygy[j,]),col="lightblue")}
date<-as.POSIXct(date)
par(new=TRUE)
plot(x,TS,xlim=xlim,ylim=ylim)
```



One could calculate the cumulative flow by adding over time the area under each trapeze at in the example below.

```
cumTS_inst=matrix(0,10,1)   # 10 values, the first one being 0 to associate to initial time
for (i in 1:9){QQ=(TS[i]+TS[i+1])/2;cumTS_inst[i+1]=cumTS_inst[i]+QQ}
as.vector(cumTS_inst)
```

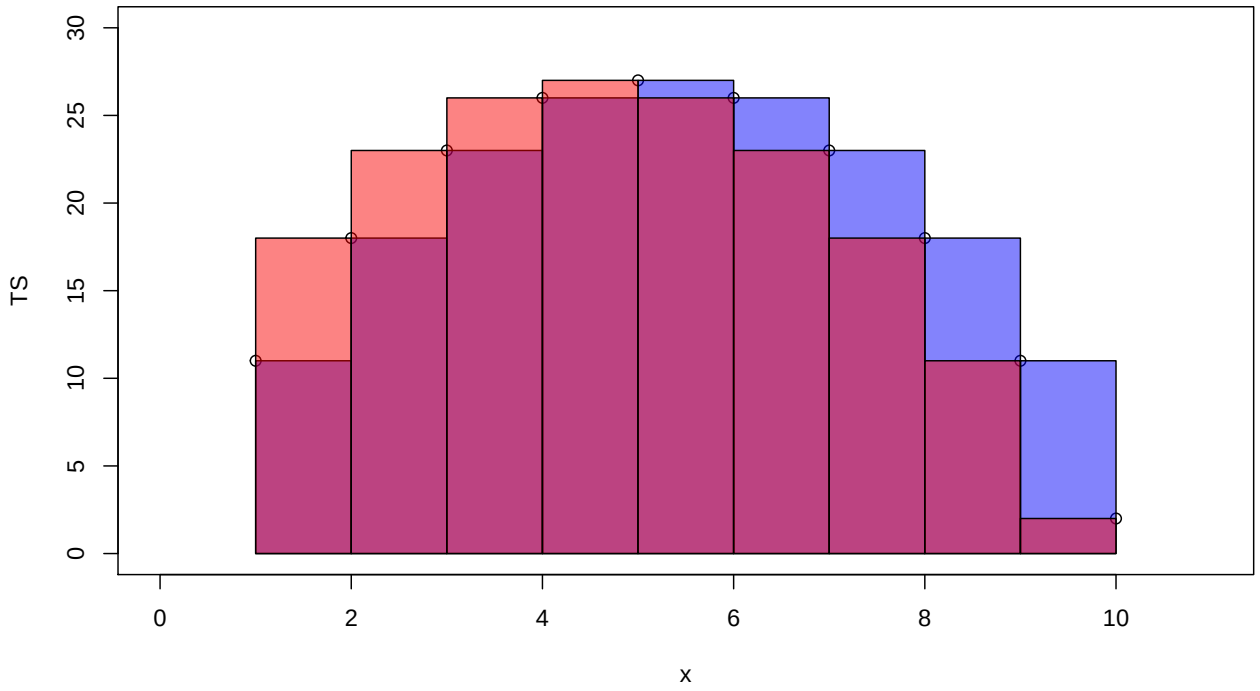
```
##  [1]   0.0  14.5  35.0  59.5  86.0 112.5 137.0 157.5 172.0 178.5
```

This works quite well actually. However, when one is dealing with thousands of points, it takes a while to compute. There is, however, a fantastic function in R, which is called `cumsum` for cumulative sum. For example `cumsum(1:5)`yields 1, 3, 6, 10, 15. The last value, thus yields the cumulative sum. One can use the `tail` function to automatically retrieve the last value such as in `tail(cumsum(1:5),1)`.

Why mentioning this? Well, it turns out that the rectangles in the figures above are essentially representing the application of the `cumsum` function. In fact one can obtain the area under the trapezes by combining the rectangle approaches just like below:

```
par(mar=c(4.5,4.5,0.5,0.5))
plot(x,TS,xlim=xlim,ylim=ylim)
for (i in 1:9){rect(infrectxy[i,1],infrectxy[i,2],infrectxy[i,3],infrectxy[i,4],col = transpblue)}
```

```
for (i in 1:9){rect(suprectxy[i,1],suprectxy[i,2],suprectxy[i,3],suprectxy[i,4],col = transpred)}
```



Each trapeze is the average of the orange and purple rectangles!! So, one can obtain the same time series of cumulative flow by using this very simple code line:

```
c(0,(cumsum(TS[-1])+cumsum(head(TS,-1)))/2)
```

```
##  [1]   0.0  14.5  35.0  59.5  86.0 112.5 137.0 157.5 172.0 178.5
```

Exercise 2

Calculate the cumulative volume in mm at the end of the year in the Maudouve at Saint-Donan.

Evaluating the importance of rare high flow events: flow duration curves

Sorting flow and load values

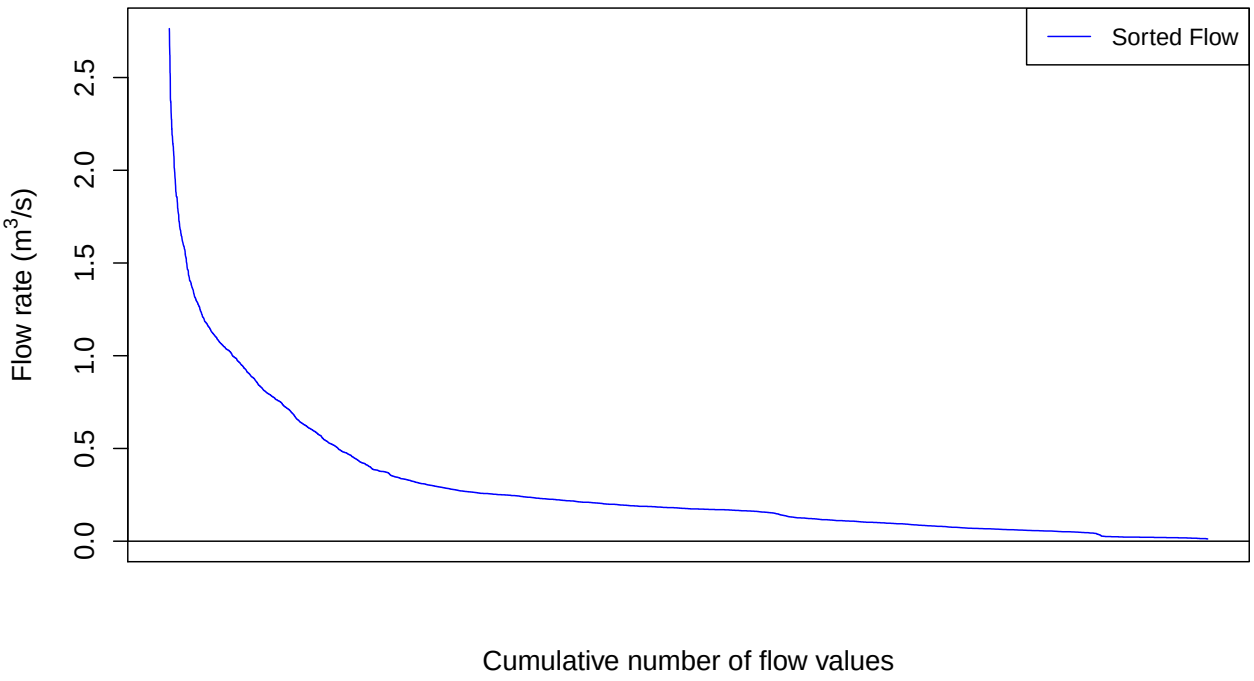
Flow duration curves represent the percentage of the total flow that occurred in x% of the time corresponding to the highest flows. The same applies for loads. This might sound a bit merky, but hopefully it will not with the further explanations below. To get there, one first needs to order flow and loads in descending order.

```
QSort=sort(Q,decreasing = TRUE)    #Sorts instantaneous flow rates in descending order
```

The blue hydrograph from above now becomes:

```
par(mar=c(4.5,4.5,1,1))
plot(QSort,col="blue",type="l",cex=0.1,yaxt="n",
         lty=ltyp[1],xaxt="n",xlab="",ylab="",ylim=ylimQ)
abline(h=0)
axis(2,cex.axis=ScaleF)
mtext(y1lab,side=2,line=3,cex=ScaleF)
```

```
mtext("Cumulative number of flow values",side=1,line=3,cex=ScaleF)
legend("topright",c("Sorted Flow"),lty = c(1), col = c("blue"))
```



The next thing to do is to integrate under the sorted curves to obtain the cumulative flow as a function for each represented flow value.

Exercise 3

Plot the cumulative sorted flow volume, expressed in percentage of the total

So now the cumulative flow volume curve corresponding to the highest flow rates as a function of the cumulative number of flow values looks like this:

Notice that there is still no unit added for x axis because I decided that the cumulative number of flow value does not really add a lot to the analysis. However, it becomes very interesting to transform these values in probability of occurence. Each value has 1/N the probability to occur. We can also calculate the cumulative probability of occurence of flow values. Flow and load duration curves are thus derived this way.

In more details, the cumulative discharge calculated at each instantaneous flow rate can be calculated as a percentage of the total discharge yielding Wk% values corresponding to the kth cumulative probability and the time elapsed at each point can be calculated as a percentage of the total time. This works because even though flow rates are rearranged, the same amount of data points exist within the dataset with the same time increment occurring between each value. Wk% values can then be plotted as a function of the percentage of the total time. This is what is referred to as Flow Duration Curves. This provides a way of demonstrating of how relatively flashy the watershed may be, either relatively to other watersheds or to previous years.

The flashiness of a watershed refers to how rapidly flow is altered as a result of storm events/varying conditions. More frequent spikes in flow in response to precipitation events, in which flow increases and decreases more greatly and rapidly, are typically indicative of watersheds with predominant portions of streamflow being influenced by surface runoff, a quicker responding contributor of water to streamflow.

```
Wk=quantile(cumQSort,probs=seq(0.01,1,0.01))/tail(cumQSort,1) #Calculates and assigns Wk% values to a p
```

Flow and Load duration curves

This method allows us to see the percentage of the total discharge that occurs in a fraction of the total time with the lowest probabilities of occurrence corresponding with the highest flow rates associated with event flow. Therefore, if one watershed produces a majority of the total discharge in 50% of the time versus a watershed that produces a majority of the total discharge in 80% of the time, that watershed may be considered relatively flashier because a greater portion of the total discharge occurs in association with higher flow rates. In other words, streamflow would be considered more reactive to event water because the event hydrograph rises and recedes more quickly than the other watershed. This quick rise and recession allows for most flow to occur in a smaller percentage of the time versus the watershed that has a much wider event hydrograph spanning across a greater range of instantaneous flow values over a greater period of time. Visually, this method can provide a relative comparison of the flashiness of multiple watersheds. In the example above, the shape of the curve in the first watershed would have a greater slope towards the lower percentages/probabilities of occurrence and the curve for the second watershed would be somewhat flatter.

```
    xlim=c(0,100);ylim=c(0,100);
    plot(1:100,Wk*100,xlab="Prob of Occurrence %",ylab="Mk% & Wk%",xlim=xlim,ylim=ylim,pch=21,col="blac
    par(new=TRUE)
    plot(1:100,Mk*100,xlab="Prob of Occurrence %",ylab="Mk% & Wk%",xlim=xlim,ylim=ylim,pch=22,col="blac
    par(new=TRUE)
    abline(1,1,xlab="Prob of Occurrence %",ylab="Mk% & Wk",col="black",lty="dashed",xlim=xlim,ylim=ylim)
    par(new=TRUE)
    legend("bottomright",c("Cumul Q","Cumul NO3 load"),
           pch=c(19,19),
           col=c("blue",ColElmt),
           bg="white")
    title(main="Nitrate Load and flow duration curves") # the \n in the text allows for line break in t

    #Code for plotting the double cumulative plot using the normal distribution probabilities to zoom i
    v<-c(1,2,3,5,10,25,50,75,90,95,97,98,99,100) #v is a set of user defined values of percentages of t
    x<-v/100  #Sets x as values of v in percentages
        y<-Mk[v]  #Sets y as the set of Mk% values corresponding with percentages of the total time def
    xx<-qnorm(x) #qnorm takes a given probability (x values in this case) and returns the corresponding
    yy<-qnorm(y) #Here, qnorm is taking the percentages of the load that occur associated with percenta
    yV<-Wk[v] #Sets yV as the set of Wk% values corresponding with percentages of the total time define
    yyV<-qnorm(yV) #qnorm takes percentages of the flow occurring associated with percentages of the to
    yy<-cbind(yy,yyV) #Binds both sets of Z-score values corresponding with Mk% (yy) and Wk% (yV)
    xx<-cbind(xx,xx) #Binds both sets of Z-scores correspdonding with defined percentages of the total
    color<-c(ColElmt,"blue") #Defines the plot colors for yy and yV

    for (i in 1:2){
    plot(xx[,i],yy[,i],xlab="Probability of Occurrence (%)",ylab="Mk% & Wk%",type="o",xaxt="n",yaxt="n"
    par(new=TRUE)
    }

    axis(1,at=xx[,1],labels=x*100)
    axis(2,at=xx[,1],labels=x*100)
    title(main="Double cumulative probability plot for nitrate load \nand for flow volume") # the \n in
    abline(h=xx,lty=3,col="grey") #Plots gridlines
    abline(v=xx,lty=3,col="grey")
    legend("bottomright",c("Cumul NO3 load","Cumul Q"),
```

```
          pch=1:2,
          col=color,
          bg="white")
```