**Some notes about detecting peaks in fast-sampling CO2 concentration signals**

M. Favaron

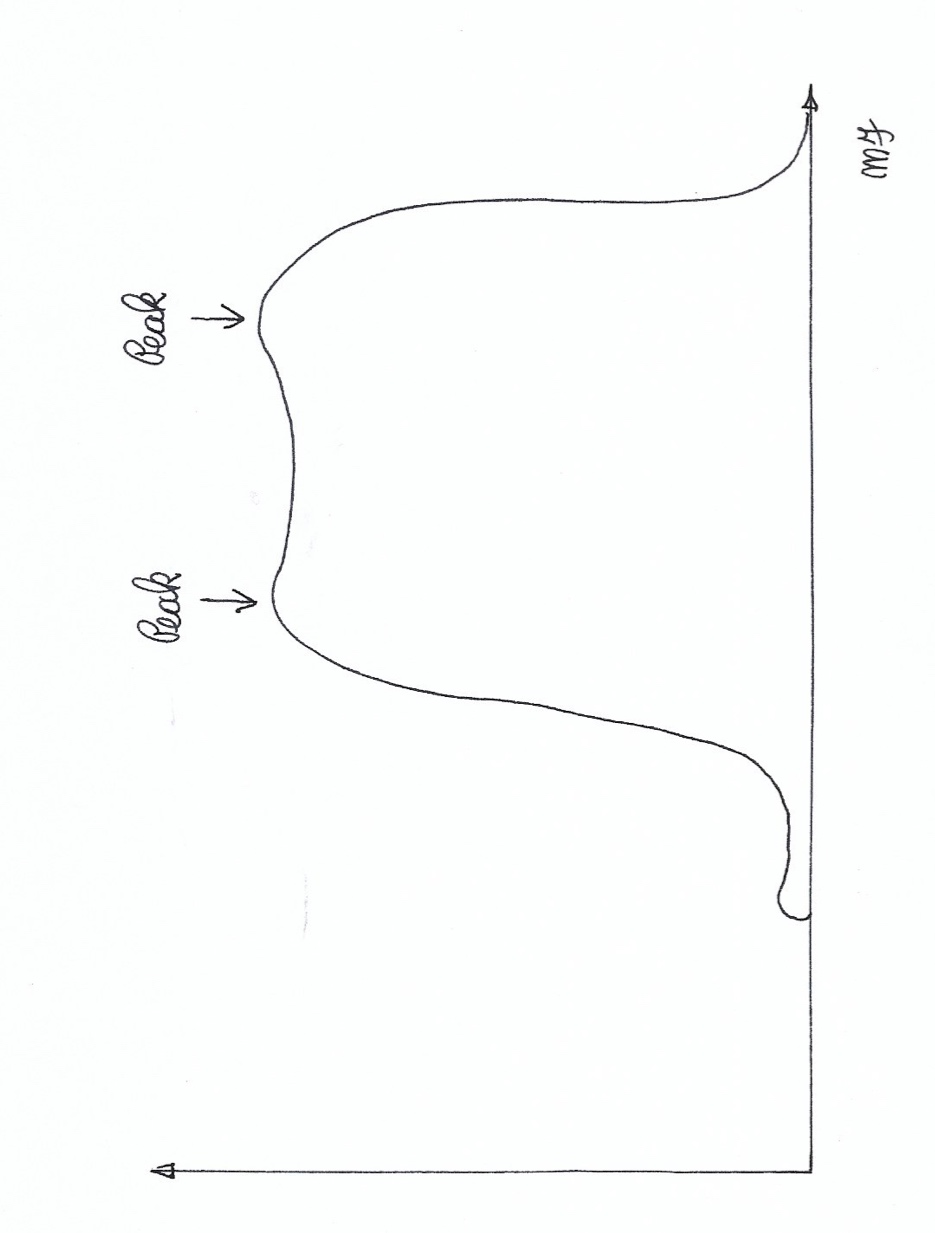
**1.Introduction**

What a *peak* is?

The question looks quite obvious, and the answer trivial. Of course, a peak is a peak, and very few of us would found this tautology offensive.

After all, the concept of a peak is so ingrained in our perceptual systems, that we all possess our personal “definition”, maybe just in terms of some neural circuit in our visual cortex. So deeply buried in the immense haystack of all others basic neural circuits, that defining a “peak” in words may even feel difficult to some people.

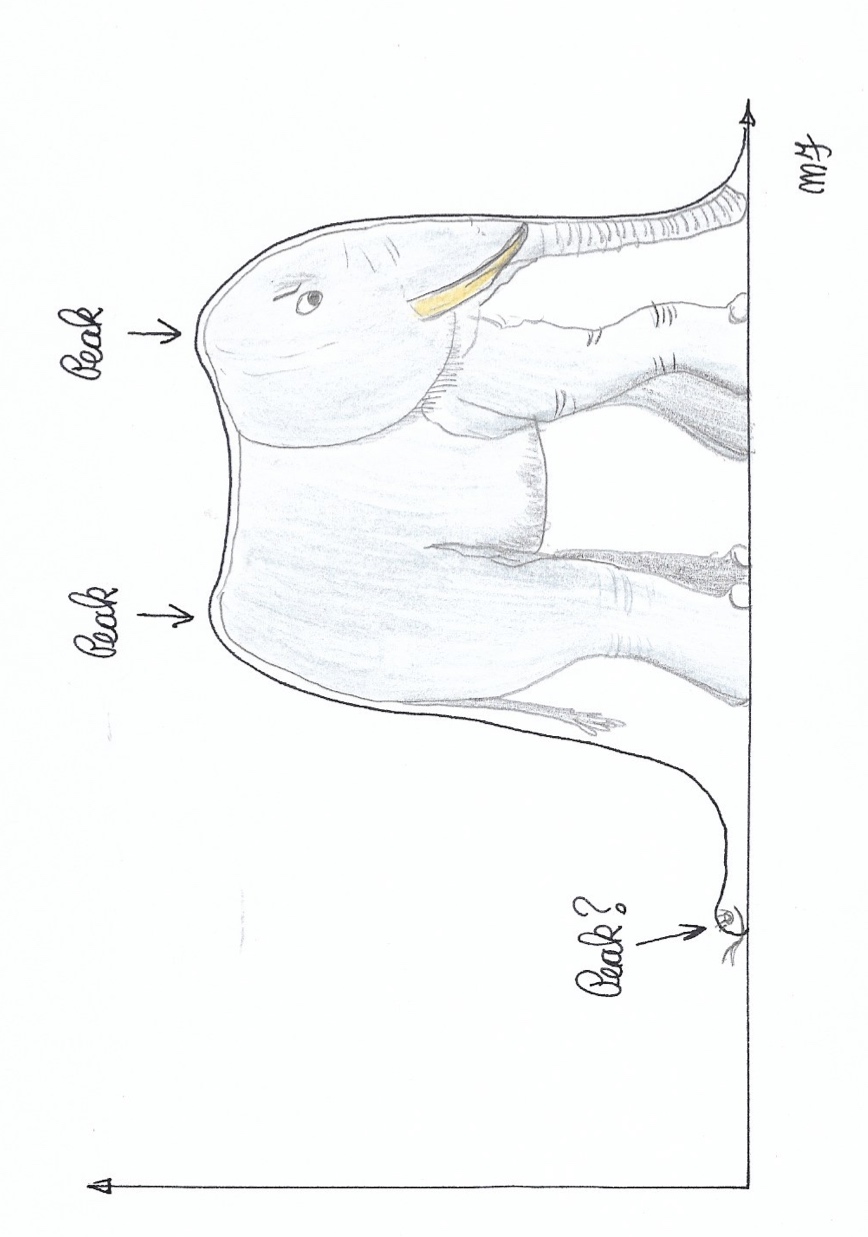
Maybe, in cases like this an inductive approach is easier to get. So, let’s start with a famous example:



Little question two peaks exist in this case (I’ve marked them). Although more a curve than the graph of a real-valued function, the plot exhibit changes above a smooth appearance, and due to these changes up and downs appear. Two ups look upper than others, and stand visually, to the extent to be very good peak candidates.

At least, as far as *I* am involved: having not yet defined exactly what a peak is, a large margin exists to subjectivity and taste.

From time ti time, it happens we can “see through things” and, for example, build a model for the phenomenon at stake, as A. de Saint Exupery surely made with his beautiful explain of our curve:

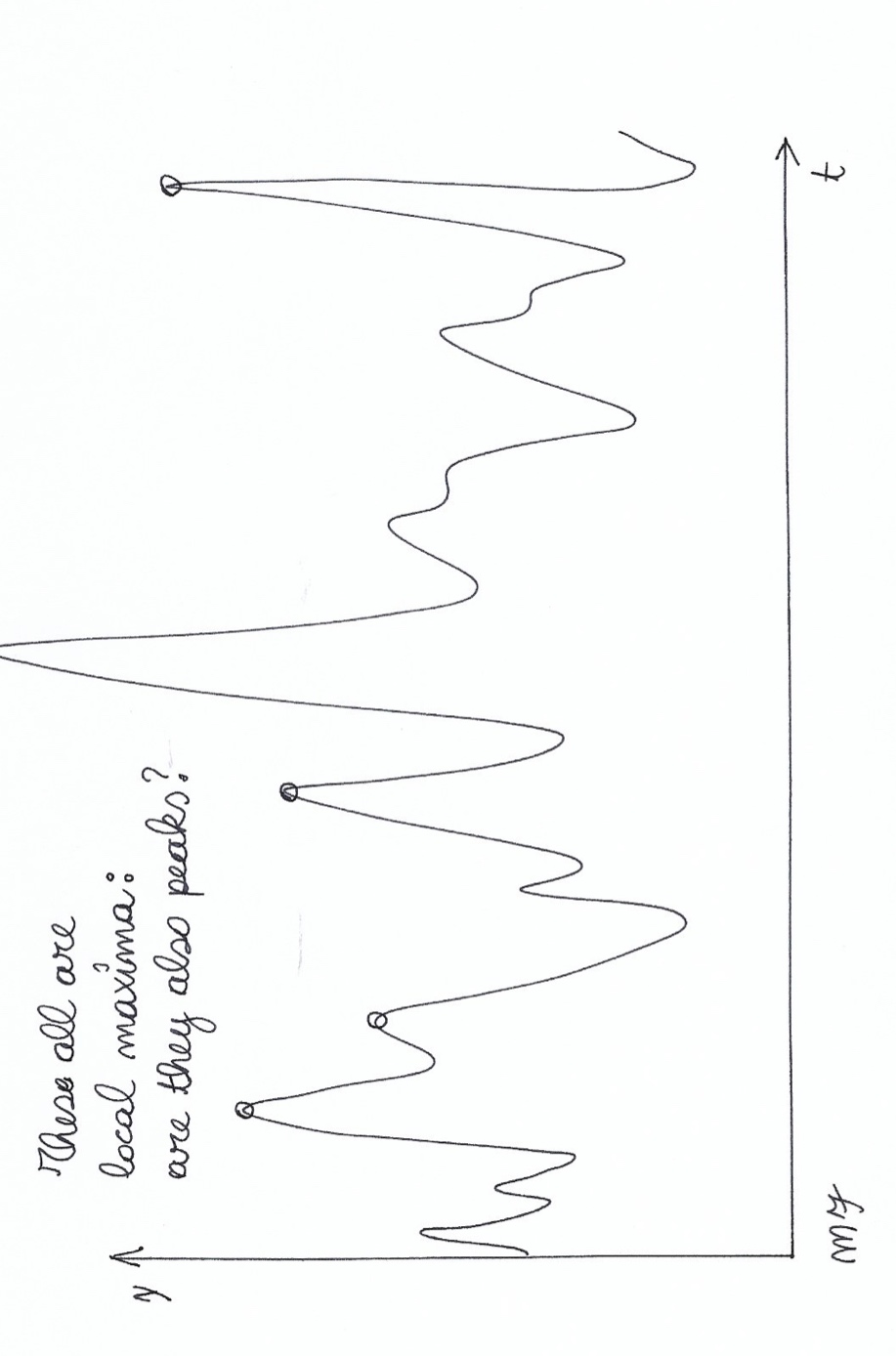


We all can understand the elephant’s disappointment – and the snake’s (surely a python given the mostly-mathematical contents of these notes) happiness. But our added knowledge carries also with it a dose of complication, and we may decide some minor features of our curve, in particular the snake’s head top, are also peaks. This is quite perceptual, in my feeling: would we have known for sure the python trying to digest the elephant was in reality a hat, I guess I’d been content with two, not three peaks.

But in fact, what characterizes a peak? Not really its absolute magnitude, otherwise we’d have to label only one peak, the rightmost, just above the elephant’s head.

Maybe, a better something could be a peak being somewhat above the “usual” level of the curve, being well aware this level may (as in our case) change quite importantly.

This looks intuitive, but, it’s also quite a dire requirement: to apply it, we should make sure in advance the curve has one, or many, “usual” levels. Of course, not all natural signals look this way, as we may see in the following figure.



In my view, what distinguishes a “true peak” from the local maxima in figure above is, in the former, the existence of a signal’s “before” and “after”, just around the peak. Ideally, these “before” and “after” parts are outcomes from appropriate-weak-stationary processes (maybe different). Indeed this is quite a personal and subjective feeling, but, also, it seems to me to capture quite well the kind of peaks one might find in concentration measurements as we made at Fontanella.

May then a “peak” be imagined as a local maximum which is also an outlier from the empirical density of the “before” part? Of course in my heart this definition looks quite sensible, and possibly correct.

Thinking to our cases, some additional points come to my mind:

* The “before” distribution is likely to be well approximated by a log-normal, with (then) well defined mean and standard deviation. But definitely it “must” not fit this expectation: often means just often, not always.
* The actual distribution may be fat-tailed, especially to the right (as a log-normal happens to be). This too is intuitive, the right density tail coming from the swerves experienced by the plume because of natural phenomena like wind meandering: something definitely “not rare”.
* I’m quite sure the distribution of the “before” part will *not* be normal. A bit of healthy pessimism may help here…
* Distributions of the “before” and “after” part are likely to match (according to my gut feeling: exactly the same 😊; my motivation might seem unscientific, but I guess the naturally existing CO2 will “react” as it would do usually, in lack of our artificial emission: and its statistical properties will change with the usual calm, as ground temperature and plants activity slowly changes with time).

I’d like to hear from you whether something more can be said.

**2.An easy detector**

Surfing the web (for other reasons) I enjoyed a sudden attack of serendipity, and found (on Stack Overflow) a simple detection algorithm.

The original is written in Matlab (which I don’t have), but translating it to modern Fortran proved not that great deal, and here is the result (sorry so sloppy):

! Find peaks in a data vector, assuming a normal distribution. This program is the Fortran

! translation of "Smoothed z-score algo (very robust threshold algorithm)"; see

!

! https://stackoverflow.com/questions/22583391/peak-signal-detection-in-realtime-timeseries-data

!

function FindPeaks\_Simple( &

rvX, &

lag, &

threshold, &

beta, &

signals, &

avgFilter, &

stdFilter &

) result(iRetCode)

! Routine arguments

real, dimension(:), intent(in) :: rvX

integer, intent(in) :: lag

real, intent(in) :: threshold

real, intent(in) :: beta

integer, dimension(:), allocatable, intent(out) :: signals

real, dimension(:), allocatable, intent(out) :: avgFilter

real, dimension(:), allocatable, intent(out) :: stdFilter

integer :: iRetCode

! Locals

integer :: n

integer :: i

real :: rAvg

real :: rStd

real :: rSumX

real :: rSumX2

real, dimension(:), allocatable :: filteredY

! Assume success (will falsify on failure)

iRetCode = 0

! Check something can be made

n = size(rvX)

if(n < 2) then

iRetCode = 1

return

end if

! Reserve workspace

if(allocated(signals)) deallocate(signals)

if(allocated(avgFilter)) deallocate(avgFilter)

if(allocated(stdFilter)) deallocate(stdFilter)

allocate(signals(n))

allocate(avgFilter(n))

allocate(stdFilter(n))

allocate(filteredY(n))

! Initialize data

signals = 0

filteredY(1:lag+1) = rvX(1:lag+1)

avgFilter(1:lag) = 0. ! Not used, really

stdFilter(1:lag) = 0. ! Not used, really

! Compute mean and standard deviation of signal beginning

rSumX = sum(rvX(1:lag+1))

rSumX2 = sum(rvX(1:lag+1)\*\*2)

rAvg = rSumX / (lag+1)

rStd = sqrt(rSumX2/(lag+1) - rAvg\*\*2)

avgFilter(lag+1) = rAvg

stdFilter(lag+1) = rStd

! Main loop: process all remaining time

do i = lag+2, n

! Locate peak

if(abs(rvX(i)-avgFilter(i-1)) > threshold\*stdFilter(i-1)) then

if(rvX(i) > avgFilter(i-1)) then

signals(i) = 1

else

signals(i) = -1

end if

filteredY(i) = beta\*rvX(i)+(1.-beta)\*filteredY(i-1)

else

signals(i) = 0

filteredY(i) = rvX(i)

end if

! Update comparison values

rSumX = sum(filteredY(i-lag:i))

rSumX2 = sum(filteredY(i-lag:i)\*\*2)

rAvg = rSumX / (lag+1)

rStd = sqrt(rSumX2/(lag+1) - rAvg\*\*2)

avgFilter(i) = rAvg

stdFilter(i) = rStd

end do

end function FindPeaks\_Simple