Signal Processing Algorithms & Architecture

Dr. Anirban Dasgupta
Department of Electronics & Electrical Engineering
Indian Institute of Technology Guwahati

Lec 5: Envelope Detection and Smoothing

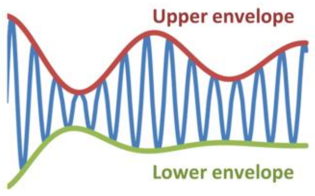
Hello everyone, welcome to a new lecture on the topic of envelope detection and smoothing. This is the course on signal processing algorithms and architectures. I am Dr. Anirban Dasgupta and let us get started.

So, to recap what we learned in time domain signal processing so far. So, we learned about time shifting and scaling, amplitude shifting, and scaling modulation, correlation, convolution and zero-crossing detection. So, we also studied peak detection. So, today we are going to study about envelopes detection and smoothing with the pattern.

Matching is the remaining topic. So, what is a signal envelope? So, an envelope is typically a smooth curve. that outlines the extreme values of the signal. Like, if you see this figure, this blue curve is the signal while the curve, which is pretty smooth, and it is obtained by the extremas of the peaks of the signal and the troughs of the signal, these are called envelopes of the signal.

So typically, in amplitude-modulated signals, so this envelope represents the amplitude of the message signal as it varies over time. So, this red one is the upper envelope formed by the maximum of the signal. Whereas the green one is depicted by the Smooth curve combining the minima of the signal is our lower envelope. So, we have understood what an envelope is. Let us try to understand what it is the problem of envelope detection. So, given a discrete-time sequence which the task is depicted by these blue lines.

The purpose of envelope detection is to find the two envelopes $E_{max}(n) and\ E_{min}(n)$ which correspond to the upper and lower envelopes respectively like in this figure,



If you see, this is depicted by this red curve and the green curve. Now, why should we do envelope detection? So, there are many applications. I would like to discuss three of them. One aspect is demodulation in communication systems as I already said, the envelope is typically a representation of the message signal in amplitude-modulated signals.

So, if we want to do demodulation, the envelope detection is a very good use case. In speech and audio processing, this envelope can give a lot of information about the speech signal, with respect to the pitch of the signal, or some prosodic features of the speech. In biomedical signals, envelope detection is also important and plays a vital role in many applications, such as removal of the baseline drift or baseline wander in biomedical signals, like ECGs. Here you can see an example. So now let us try to understand.

What are the algorithms for envelope detection? So, three of the most popular algorithms Include rectification with smoothing, peak. Detection with smoothing and the Hilbert transform. So, in this lecture, I would like to discuss. About the first two topics, which are rectification. and peak detection and also since both of these Methods use smoothing, so we will discuss.

The smoothing techniques as well. So, first algorithm is rectification and smoothing. So, in this, what is rectification? So typically, you might have rectification studied in terms of the conversion of AC to DC. So, in this context, rectification typically means taking the absolute value of the signal. So, the first step of the algorithm is to take the absolute value of the signal $x[n]$ and then perform the low-pass filtering operation on this absolute value signal and you keep doing this until you find a suitable e_{max} or your upper envelope.

Now, what about the lower envelope? So, this is also done in a similar way, but not the same way that you first get the inverted signal that is $-x[n]$ take the modulus and find the negative of that value and that signal let us say this is y_n[n]. So, perform low pass filtering on this signal y_n[n] and this will give you the lower envelope. So, since this is an algorithm, we will understand or analyze the computation complexity of this algorithm. So, the first is taking absolute value. So, if the signal is of length n, you I have to take the absolute value for each. which is typically an $O(n)$ operation. And for filtering, consider that we are Using a convolution where m is the Dot m will be the operation of the filtering operation. And similarly, the remaining two are exactly kind of the same operations. So, it will also be

$$O(N) + O(N \cdot M) = O(N \cdot M), M \text{ being the filter size}$$

Now, out of these two the costliest operation is the low pass filtering operation. So, we can conclude that the complexity is order of n dot m. So, here are two examples of signals where rectification is performed which will be followed by smoothing. So, if you see this, blue is the original signal, and the red will give you the inverted signal. So, the upper half will denote the upper envelope. It is not the upper envelope. So, the low pass version of this positive signal will be the upper envelope. So, if you perform a low-pass filter, it will be something like a smoother version like this. So, this gives you the upper envelope of the signal whereas this will give you the lower envelope. Now, if the signal is a bit more complicated, so this is a beautiful sign like a wave, but if the signal is complex, not simple in the sense complex values but complicated in nature, so still you can use this method. So, you do this rectification first, and then on these values, you can perform smoothing operation to get the upper envelope and similarly, you can get the lower envelope.

So, this technique is very nice when performing operations like demodulation but in complex problems, it may not be so straightforward. So, the second method often solves this because an envelope, by definition, is the interpolation of the peaks, we already have learned a lot of peak detection algorithms.

So let us apply peak detection first then we will interpolate the peaks. So, this is the second algorithm which is peak detection and smoothing, so in this. What we will do is algorithm. Identify the local maxima in the signal. Instead of rectification, and the same. I will follow for the lower envelope as well. I am just talking about the upper envelope. So, this can simply be extended.

For the lower envelope, then we have to identify the peaks, so identification. Of the peaks we have already discussed. The various algorithms are typically. All of them or most of them have $O(n)$ complexity. So first step is peak detection. The second step is to interpolate between the peaks or the local maxima. So, how do we interpolate? We can generate.

A smooth curve, and we can use either linear, quadratic, or cubic spline interpolation to interpolate the peaks and that will typically have $O(p)$ complexity for p peaks is typically less than n and not equal to.

So, the next step or the last step is the smoothing operation and in this smoothing operation we will try to reduce some presence of noise in the signal in the envelope. Typically, again, if we consider smoothing by using the convolution operation, this will be $O(n \cdot n)$. Now, if we compare the complexities, so this smoothing operation is the most time-consuming operation. So, the complexity of this algorithm will also be $O(n.m)$. So, this is a typical example where we have a signal that is represented by this blue curve and using this, we find the local maxima and minima. So, the red dots are the peaks that are the local maxima, whereas the green dots are the local minima, and then the interpolation is done, and this interpolation is pretty smooth. So. there is no need for any low pass filtering operation.

But you can see that this is the method of finding the envelopes work well. Now if the filter structure were a bit more complex than the low pass filtering will make the envelopes smoother. So now we have talked about the term.

Smoothing, and this smoothing also appears. As a time-domain signal processing problem. So let us try to learn some smoothing algorithms too. Typically, in the normal signal systems or signals processing course, this low pass filtering is termed smoothing. But smoothing is not just a linear filtering operation, smoothing can be done non-linearly as well. So, here I will study three popular linear filters. Means these have a kernel, and you have to convolve these filters to find the smoothed result version of the signal then order statistic filters that are typically non-linear filters they rely on order statistics, such as ascending order or in descending order of the signal window of the signal and then another popular filter which is Savitsky-Goulet filter. Now those who are very familiar with Python based on signal processing, they will see, or they will use a lot of Savitsky-Goulet filters in their smoothing operations.

So let us understand these things. So, moving average filter. As the name suggests, we take the average, and then that central value. of the signal is replaced by its averaged value. So typically, if I have a kernel, let's say I have Say some signal value; say I want to do an average for a window of length 3, say L equals 3. So in this case, my formula will be-

$$y[n] = \frac{1}{L}\sum_{k=0}^{L-1} x[n-k];$$

L is the length of the filter. So typically, If I put this, then $y[n]$ is nothing but 1 by 3. It gives me $\frac{1}{3}\{x[n] + x[n-1] + x[n-2]\}$.

Now this is a causal moving average filter. Why causal? Because it is the output. There is no dependence on any future value of my input. But in some implementations, which are not real-time; they can use non-causal. implementations also where instead of taking $x[n-2]$, they take $x[n+1]$.

And typically, if you see this, it is the kernel 1 by 3, 1, 1. So, which is doing the average of the past three values of the signal and now, since the length of the the signal is L, and it is a convolution operation and the convolution are done in the naive way, so we will say the order is n · L. Another important filter is the exponential moving average filter, and this is typically implemented in a recursive manner, or it is also an IIR structure where my output is dependent on some fraction of the present input and some fraction minus that of the just immediate past output.

And this alpha will determine what fraction of my present input, it is to be taken and the previous output, which is of course dependent on the previous inputs as well and since this is implemented in a recursive manner, so this is kind of a dynamic programming structure and your exponential moving average filter will have a complexity of O(n). So, another similar filter is the gaussian low-pass filter. So, what is the difference with Gaussian? A moving average is when I take the average; I am giving each sample an equal weight, which is 1 by L, where L is the length of the filter. But in Gaussian, as you know, the bell-Shaped curve, so it has the highest weight to that central pixel and as it the goes away from the central pixel weight is decreasing, which means that it will give the highest weight to the present input and as I go away from the present input, the weights. Will decay exponentially like this bell curve. So, this is a typical Gaussian filter.

You can take some value of sigma's and that will give you your specific $h[n]$. So these are some comparisons, say I have a noisy signal like this one. So, with our moving average filter of size 5, with our exponentially moving average filter with an alpha value of 0.7 and a Gaussian filter of size 5 and sigma of 0.2 you can see that the smoothing operations are varying significantly. So, here in the moving average, we can see that it is smoothing very nicely. So, if this is an envelope detection problem, then we can see that the moving average is performing. Nice because it is creating an envelope-like structure.

But, on the other hand, it is removing some detailed information in this region where the Gaussian on the other hand is retaining that, the exponential moving average. Is also retaining that, but my moving average filter is distorting or smoothing out that detail which may not be desirable in many signals processing application.

But for the context of envelope detection, this is preferable or may be desirable. Now coming to a non-linear filter which is the order statistic filter. So, in this order statistic filter, it is very useful when we are dealing with outliers or impulse noise and the three main filters in this context are the. Median filter, max filter, and mean filter and the complexity will be

$$O(log\ L \text{ and } N\ log\ N).$$

Why? Because there is a sorting operation required. Now for example Say if I have some values like 3, 2, 7, 6, and 5. and this value needs to be replaced. So first, I need to sort them out. So, it will be 2, 3, 5, 6, and 7 and now, if I am using Median, so this middle value will be replaced by 5, which is the median. If I use a max filter, it will be 7, which is the same value, so it will have no effect and if I use a min filter, it will take 2 which is the minimum of the window. So, these are the comparison and contrast technically, we see that the median is doing well. So, it is retaining the structure kind of to a great extent, but the min and max filters in this case, they are not performing very nicely. But maybe again for envelope detection.

The problem is that these min and max filters may be useful. Now coming to the last filter, which is the Savitsky-Goulet filter. So, in this what we do is polynomial fitting. So, we have some data points, and a low-degree polynomial fits the data points typically of a small the window and the polynomial fit can be a quadratic or a cubic spline, and then a least squares fit is made to the polynomial. So, the points in that window will be replaced by the least squares fit and this will typically smooth the signal, and the window needs to be moved from one place to another like a sliding window. So, this is the typical algorithm. Savitsky-Goulet filter where first I have to select the window size and then the polynomial order is typically like this. An order of 2 or 3 is sufficient for most signals and then we have to fit the polynomial. So, once this polynomial is fit, now the central value of the window is replaced by the windows polynomial and one more important thing is that since we have to pick the central value of the window is typically an odd sized window, even for the order statistics filter also. And then we have to slide the window and repeat the same. So, thank you very much. We will meet again in the next lecture.