

Introduction to Time-Frequency Analysis and Wavelet Transforms
Prof. Arun K. Tangirala
Department of Chemical Engineering
Indian Institute of Technology, Madras

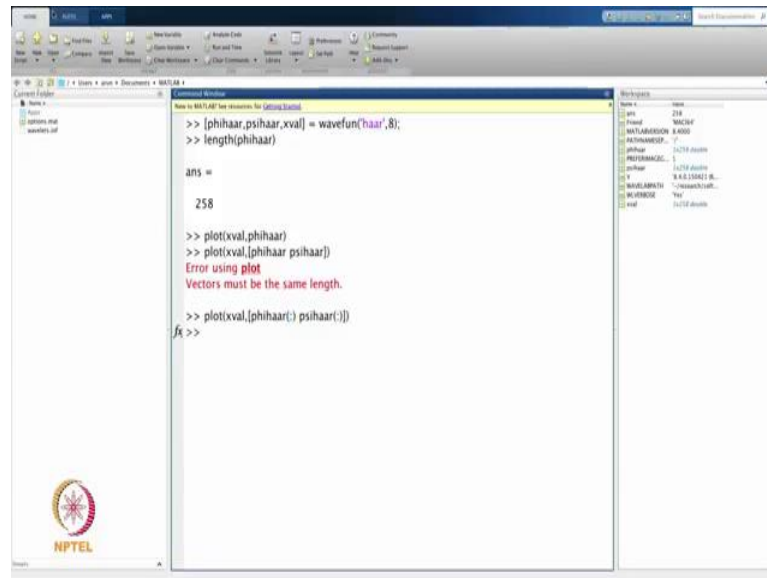
Lecture - 08.5 A

Hello friends, welcome to lecture 8.5 on the in the unit on discrete wavelet transforms. Until now we have looked at the theoretical aspects of DWT, we have studied extensively the orthogonal DWT, and the properties of wavelets, how to compute the DWT. When we say DWT by default it is understood as orthogonal DWT, so how to decompose the signals into different scales or different frequency bands and how to reconstruct or synthesize, and studied certain key relations between the properties of the wavelets such as compact support, and vanishing moments, and so on.

In this lecture what we are going to do is look at the practical aspects, both how to implement, in the sense how to compute the DWT in MATLAB, and study briefly on the boundary effects, remember DWT involves convolution like CWT. And any convolution operation is affected by the border effects; that is how the signal behaves at the borders; that is outside the observation interval. And we are going to primarily discuss 3 popular applications of DWT, namely the discontinuity detection; and more popular one such as signal compression and signal estimation. Again I will show you how to do things in MATLAB with the few illustrative examples.

So, let us begin our journey. The first thing that we would like to do in MATLAB is to be able understand, to be able to generate wavelets which we have seen of course yesterday, but get the different wavelets or different filter coefficients and so on. I have already discussed that routine with you. We have going to primarily use the MATLABs wavelet tool box, and the routine that gets the wavelets for you or wavelet filters or wave fun and w filters. We have seen the usage of wave fun earlier.

(Refer Slide Time: 02:30)



The screenshot shows the MATLAB Command Window with the following code and output:

```
>> [phihaar,psihaar,xval] = wavefun('haar',8);
>> length(phihaar)

ans =

    258

>> plot(xval,phihaar)
>> plot(xval,[phihaar psihaar])
Error using plot
Vectors must be the same length.

>> plot(xval,[phihaar;psihaar(:)])
fx >>
```

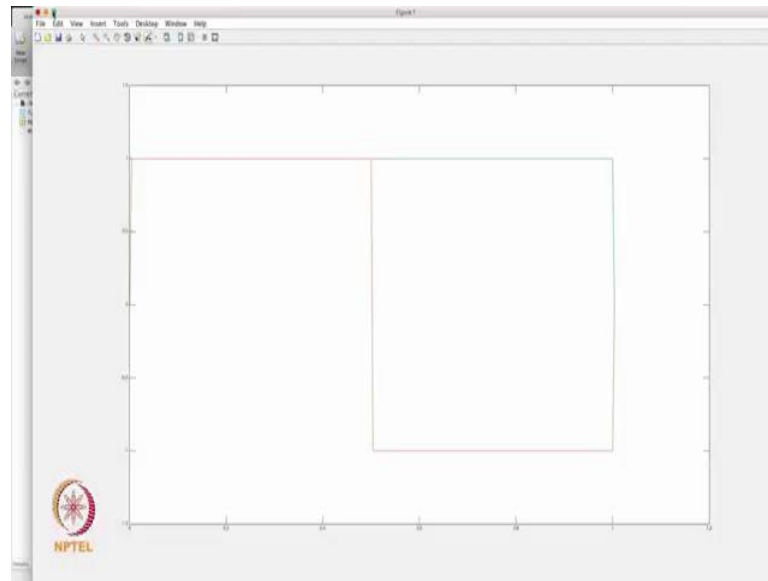
The right-hand pane shows the Workspace with variables: ans (258), phihaar (1x258 double), psihaar (1x258 double), xval (1x258 double), and a list of other workspace variables.

So, suppose I want to generate a Haar wavelet, then again I use the wave fun and I want to have the scaling function, the wavelet and the values at which these are being computed. In the Haar case, remember that we have an explicit expression. We do not have to compute this scaling function or the wavelet from the filter coefficients. We know already that the scaling function is a box function, and the wavelet function is box with a reversal of the sign, half way through.

So, the second argument here, as we have seen earlier, represents the number of iterations it takes to compute the wavelet from the filters for the Haar wavelet alone and a few other wavelet for which close form expression exists. This second argument is simply going to tell the wave fun as to how many points at which this scaling functions or the wavelets have to be generated.

So, here, if I look at the length of 5 Haar, it is roughly, 2 power 8 plus 2. It does not give, normally for wavelets which have closed form expressions, the second argument in wave fun will generate the scaling function or the wavelet at 2 power that second argument. But here, the additional thing that is done is 2 power 8 plus the length of the filter. So, let us plot the scaling function for the Haar wavelet; it is a familiar figure for us.

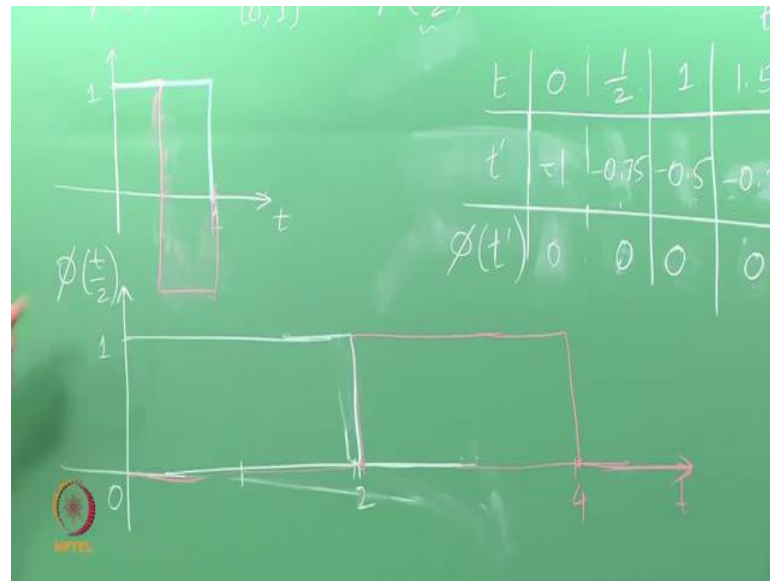
(Refer Slide Time: 04:33)



So, this is how it look like; it is a box function. And we can also plot on the same figure the wavelet as well, because the scaling functions and the wavelets are returned as row vectors, we should force them as column vectors. And therefore, you can see now, the scaling function and the wavelet; scaling function shown in blue and the wavelet is shown in red. We can actually ask for a thicker line width to improve the visibility.

Now, hopefully, you are able to see the scaling function and the wavelet. Very often one has difficulty in visualizing how these functions whether $\phi(t)$ or $\psi(t)$ look like when they are scaled and translated, either dilated or compressed. Normally, we are interested in the dilated versions because, as we have to learnt earlier, the signal is available at the finest time resolution, and we always end up constructing only approximations; an approximations are generated by the dilated versions.

(Refer Slide Time: 06:25)



So, assume that this scaling function or the wavelet that we have plot is available here at the same resolution as the signal, and we want to generate $\phi(t)$ by 2 or even $\phi(t)$ by 2 minus 1. Then, let us first understand by hand how to generate these values. If I am given the $\phi(t)$ which here we know is the box function in the interval 0 to 1, which means it is understood to be 0 outside this interval, then I would like to construct $\phi(t)$ by 2. It is fairly easy.

The first step is to have a dummy variable which is equal to t by 2. So, let us call this dummy variable as t' . And now all I have to do is evaluate $\phi(t')$, given this information. Note that this t' is a dummy variable. It tells me essentially what is the values of ϕ at any value that t takes here. So, that is the, that is how you interpret the information. So, let us understand how to do this. I would like to evaluate $\phi(t)$ by 2. Let us call that as $\phi(t)$ prime.

And my objective is to calculate what is $\phi(t)$ prime as a function of t given this information. So, what we shall do is we shall take values of t here, but instead track $\phi(t)$ prime, right. Because, eventually, I want to plot $\phi(t)$ by 2 versus t ; that goal does not go away. So, when t is 0, of course t' is 0. So, what we shall do is we shall have another row for t' . And then a third row for $\phi(t')$ which is nothing but $\phi(t)$ by 2.

So, when t is 0, t' is 0. And let us pick another value here. When t is, let us say, half

then t prime is $1/4$. So, I know, ϕ of t prime therefore, when I am evaluating $\phi(t)$ prime at t prime equals to 0 I am asking what is ϕ at 0; ϕ at 0 by definition is 1. And likewise, here ϕ at $1/4$ is 1, because at any value in this interval 0 to 1, ϕ assumes the value of 1, right. So, I have here; and then I pick 1 here for t , therefore, I get half; likewise, I get $\phi(t)$ prime to be 1, and so on.

So, if I pick here 1.5, I have 0.75 for t prime, once again the value is 1. And when t is 2, t prime is 1, right. And then I have ϕ to be 1 at this point. As you can see, already $\phi(t)$ prime is a dilated version, why? Because, $\phi(t)$ is 1 in the interval 0 to 1, but $\phi(t)$ by 2 is 1 in the interval 0 to 2. So, it is been already stretched. The moment I step out of this interval, 2.5, let us I say, 1.25 it goes to 0, and so on. So, even if I look at 3, therefore, remains 0.

So, if I were to draw $\phi(t)$ prime this is how it would look like, this is if $\phi(t)$ looks like. And let us say, this is 1, and here is 1. So, this is how $\phi(t)$ looks like. Here the x axis is t ; y axis is $\phi(t)$. Now I am going to draw $\phi(t)$ prime or $\phi(t)$ by 2. So, the x axis is still t , right; x axis is still t , but y axis is $\phi(t)$ by 2; and I have 0, 1, and then here 2; this table here tells me that $\phi(t)$ by 2 is now a dilated version and goes to 0 after this; you see that.

Now, I need to compute; remember, not only do I need to compute $\phi(t)$ by 2, I need to be able to compute integer translates of this. At any scale the approximations coefficients in theory are the inner products of the signal with the scaling functions at that scale and its translates. So, how does the $\phi(t)$ by 2 minus 1 look like? Obviously, we do not use this to compute the coefficients, we use the fast algorithm. But, you should be really comfortable in visualizing what ϕ , how $\phi(t)$ by 2 looks like, and how its integer translates look like.

So, let us ask, what is $\phi(t)$ by 2 minus 1, the first translate, alright. So, now, let us call this as t prime. So, what we are going to do is we are going to erase these values here, and ask how $\phi(t)$ prime, now my t prime is no longer t by 2, but it is t by 2 minus 1, right. So, when t is 0, what is t prime? T prime is minus 1, alright. So, what is ϕ at minus 1? This information is the key; ϕ at minus 1 by definition here is 0 for the Haar scaling function, is 0.

When t is half, I have $1/4$ minus 1. So, I have minus 0.75. And once again this is 0. When t is 1, I have minus, t prime is minus 0.5. So, $\phi(t)$ prime remains 0. When t is

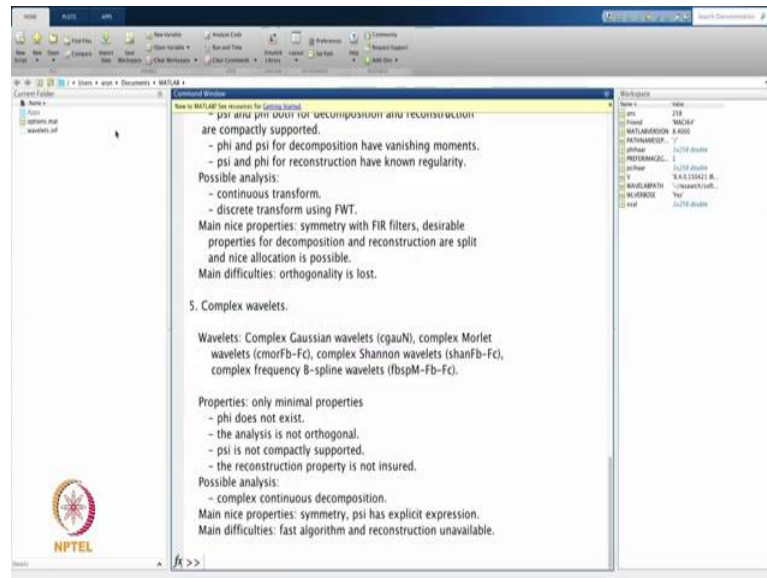
1.5, t prime is minus 0.25 and ϕ remains 0. When t is 2 then t prime is 1, and that the, t prime is 0 and that is when the $\phi(t)$ prime switches on, right. And when t is 2.5, I have here 0.25 and ϕ remains 1, and so on. So, the last value here at which it remains 1 is t equals 4 at which point t prime is 1. And beyond 4, t prime is greater than 1, as a result $\phi(t)$ prime is 0.

So, how does the plot of $\phi(t)$ by 2 minus 1 look like? Just for convenient sake we shall draw here on the same plot; now we will use a different color to plot $\phi(t)$ by 2 minus 1. It says that, this table tells me that $\phi(t)$ by 2 minus 1 remains 0 until t takes on a value of 2, right. So, it is going to remain 0 here. And really, what is going to happen is, at this point $\phi(t)$ by 2 minus 1 takes a value of 1 and remains non 0 until 4; remains 1 until t takes a value of 4, and then it goes to 0 once again.

So, now it is very clear that at any scale the integer translates of this scaling function are non over lapping. In fact, if you take the inner product of these 2, it will turn out to be 0, right. And you can repeat this exercise for yourself for the wavelet as well; the only difference is, the wavelet has the shape, as you can see on the board; so this is blue; we have the scaling function, and the wavelet look like this.

In fact, different, the values are different as you can see. So, it actually goes this way, right. That is how the wavelet looks like. Once again you can write a table for $\psi(t)$ by 2, and ψ of, and all it is integer translates, and get an idea. Once you have done this for 1 or 2 integer translates you get an idea of how they look like. But, the key point is that we can see the dilation clearly. In fact, $\phi(t)$ by 4 would extent until 4, and so on. So, this is the useful exercise to understand how these wavelets or scaling functions and the dilations and translations look like.

(Refer Slide Time: 15:49)



And now get back; and of course, you can use this wave fun to generate a number of wavelets. And if you want to know what wavelets are available, type wave info at the command prompt; and it will give you all the different classes of wavelets that have been coded in this tool box. Namely, they are something called crude wavelet; is actually nothing crude about this; they are essentially the complex or the real valued wavelets which do not have necessarily an admissibility constant or they do not have a scaling function in the sense of DWT.

So, the family of wavelets that fall into this category are the Morlet wavelets, the Mexican hat. Actually Mexican hat wavelet has a scaling function. So, in some sense this is misleading; it tells you that there is no scaling function, but there is a scaling function that is useful in CWT but not useful in DWT. The scaling function that comes out of a Mexican hat does not have the orthonormal property; that is the scaling function and its integer translates do not have the orthonormal property that are necessary for DWT. That is why it is said here that there is no scaling function, but that is not entirely true.

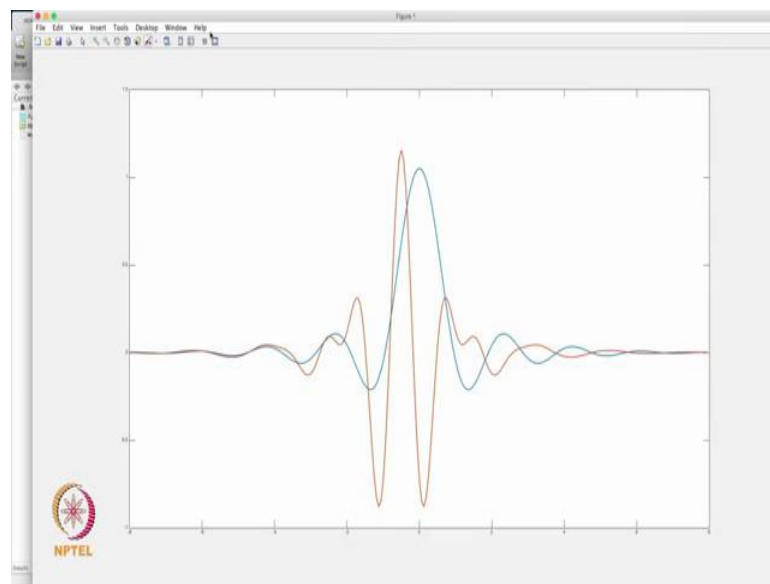
And then you have infinitely regular wavelets; what this means is that there are a number of the vanishing moments are very high. And the properties of each of this family is given here. So, if you look at the meyer wavelets, these are the only regular wavelets which do not have a compact support necessarily. In fact, they do not have a compact support, but the scaling function that comes out of the meyer wavelet can be used for

DWT.

So, if you look at the properties here, ϕ exists and the analysis is orthogonal which means that the scaling function can be used for DWT. And the ψ and ϕ are indefinitely derivable which means differentiable, and it should be, should understand that. It says that the ψ and ϕ are not compactly supported. In fact, you can ask for the meyer function and understand why they do not have compact support. We have seen the meyer wavelet before. But, just for own confirmation let us actually ask for the meyer wavelet.

Unfortunately, the string that you are supposed to pass on to wave fun is not consistent to the name of the wavelet. So, the correct string that has to be that is acceptable is m e y r for whatever reason. So, let us plot the meyer wavelet.

(Refer Slide Time: 19:02)



So, this is how the scaling function; the one that is shown in blue is a scaling function, the one that is shown in the red is the wavelet function, both belonging to the meyer family. It is clear now why the scaling functions or the wavelet functions from the meyer family do not have a compact support because they only died on exponentially. Consequently, the filter coefficients associated with this are not going to be of finite length; that means, they are not fir filters.

Now, what does it mean? Well, it means that in DWT you may not get all the nice computational efficiency and so on, and that the $\phi(t)$, of course has a large number of

vanishing moments and so on. So, if you go back to the properties of wavelets, here it is says, for Meyer wavelets the filters, the discrete transforms exist, but you have to use non FIR filters. However, they are symmetric as you have seen from the scaling function. You can see here that the scaling function is symmetric, unlike the Daubechies wavelet or unlike the Daubechi scaling function.

Then come the family of the orthogonal compactly supported wavelets. So, you see, as I said in the previous lecture, until Daubechi came up with her result it was somehow believed that there exists no compactly supported wavelet that could be orthogonal, except for the Haar wavelet but that is discontinuous. So, people were in need of an orthogonal compactly supported continuous wavelet, and that was the great contribution by Daubechi.

In showing that one can design such a wavelet. And in this family, once the Daubechi wavelet was discovered and designed, the symlets and coiflets that we discussed in the previous lecture also came up. And we have already studied the properties of this wavelets, and likewise. So, go through each of this family and you will get the information.

So, now, let us get to business here, and look at how DWT is computed using one of these wavelets. So, for this what we shall do is we shall take the signal that we had used in one of our lectures, lecture 8.3, where I showed you the DWT of a piece wise regular polynomial from Mallat's books. And let us generate 256 observations of this signal. Let us make sure that this is the signal indeed that we want to analyze. So, this how it is, that is it looks right.

Now, I would like to perform DWT of this. Of course, we can do a CWT, if you want to detect discontinuities and so on. So, suppose I want to perform a DWT, and maybe I want to perform signal compression, let us say, right, I want to compress this. I can perform a 1 level DWT or 2 level DWT and so on. The basic command here that performs a 1 level DWT is the DWT command itself. It performs only a single level discrete wavelet transform.

So, if you have a signal and you want to perform a 1 level DWT, or if you have approximation and detail coefficients at one level and you want to go to the next level and so on, this DWT routine is useful. So, let us just, let me just show you the usage of this routine. What we shall do is we shall say `c a, c d`; essentially, `c a, c d` are my

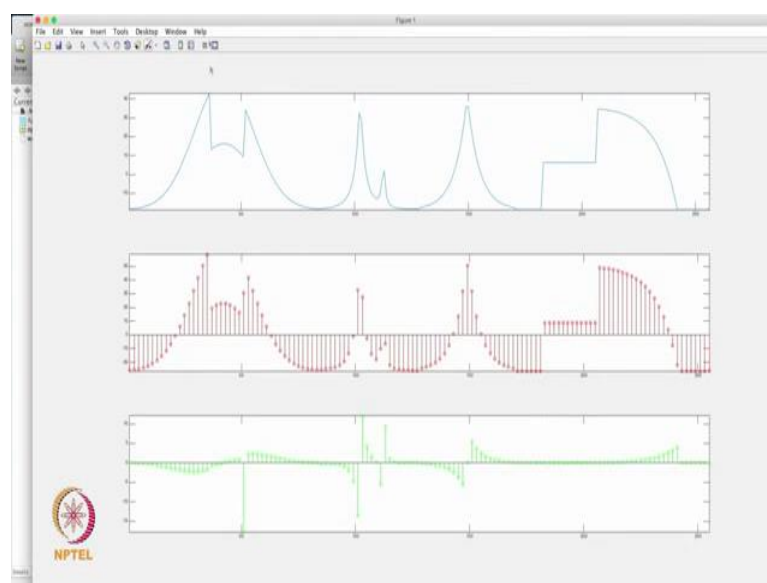
approximation and detail coefficients of a 1 level DWT that I am going to perform on the signal. And let us use a Haar wavelet, I can use a Daubechi and so on, we will do that a bit later.

So, now, I have the approximation coefficients, detail coefficients from the Haar wavelet decomposition, one level Haar wavelet decomposition. It is always convenient to plot the signal and the coefficients. In this case, I have a signal and I have 2 sets of coefficients because I performed 1 level DWT. Therefore, I will plot the signal here and ok; we have plotted the signal. And next comes the approximation coefficients. In this case, it is always useful to plot the approximation coefficients as a stem plot.

Typically, stem plot is the best way to plot. And also, make sure that you keep the time index intact. If you just ask for a stem plot, the stem plot would plot approximation coefficient verses their index, but what we want to show here is that the resolution of the approximation coefficients in time has fallen down by factor of 2. So, what we shall do is, let us say, n is the length of the signal; and then we shall plot 0 to n minus 1 or you can say 1 to n , whichever way you want it to be.

So, let us take it from 1 to n for convenience, and plot c_a ; and let us plot them in red. So, this is how, and then you shall. So, this is how the approximation coefficients look like. I will show you the plot maximum mode, maximized mode once I finish plotting the detail coefficients.

(Refer Slide Time: 25:57)



So, now stem c d, let us plot it in green. And the rest of the story is the same; next site and then maximize this. So, what do I have here? I have the signal on the top and then have the approximation coefficients here and the detail coefficient. So, something interesting emerges here. This is a level decomposition indeed. Now, if you look at a signal there are discontinuity locations at these points here.

So, let us pick 1 discontinuity location here. I have a discontinuity location around this point. And correspondingly, in the detail coefficients I see a spike, right, and that is true for other discontinuities as well. Of course, you should notice, the first thing that you should notice is that the resolution of the approximation coefficients in time has fallen down by factor of 2. I have plotted the signal as a continuous plot; ideally, I should have plotted the signal also as stem a plot, is a discrete signal. But, anyway, we know the time sampling interval for the signal. So, here I have sampling interval fallen by, multiplied by factor of 2.

So, coming back to the point; where ever there are these sharp changes in the signal, particularly of these types, I notice that there are spikes. It is not that I will have spike in these detail coefficients, at all points of changes; and that is the most important thing. In CWT also we have seen this. We have said that the sharp discontinuities or the singularity points will have a specific signature; what is that signature? That you will see spike in all the frequency bands.

So, if I look at the detail coefficients here, they are in the frequency band, $\omega_{\max}/2$ to ω_{\max} ; ω_{\max} is an frequency of the signal. And approximation is corresponding to the low frequency band, 0 to $\omega_{\max}/2$. So, not all sharp changes here belong to the high frequency band. In fact, to be able to say that a point in the signal is the singularity or not, this spikes for those points should occur across many frequency bands.

Here we have performed only one level decomposition. But if I perform, let us say 3 level decomposition, then I will be able to see those spikes coming in. So, DWT has a different signature here as compared to CWT for some singularities. Now, suppose I want to perform a 3 level decomposition, how do I do it?