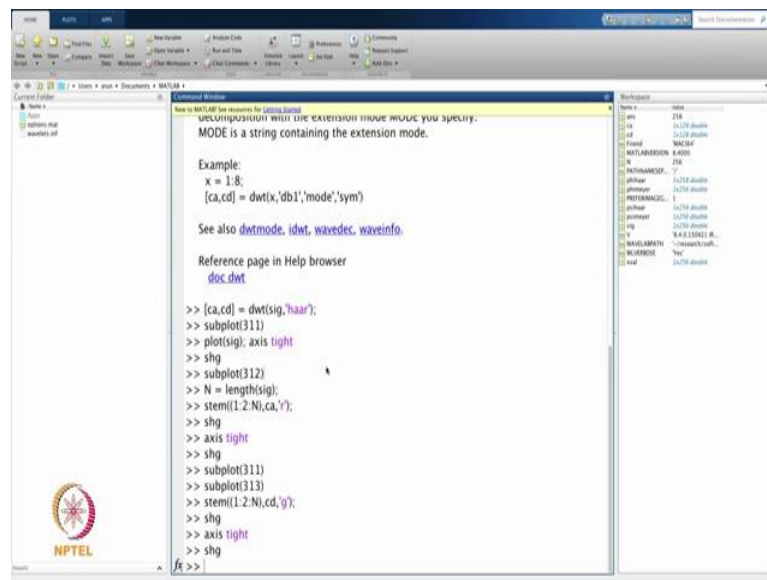


Introduction To Time-Frequency Analysis and Wavelet Transforms
Prof. Arun K. Tangirala
Department of Chemical Engineering
Indian Institute of Technology, Madras

Lecture – 08.5 B

Do I use the DWT repeatedly well fortunately not you will there is a function called wave decomposition, wavedec.

(Refer Slide Time: 00:15)



```
How to MATLABize the resources for getting started
decomposition with the extension mode mode you specify.
MODE is a string containing the extension mode.

Example:
x = 1:8;
[ca,cd] = dwt(x,'db1','mode','sym')

See also dwtmode, idwt, wavedec, waveinfo.

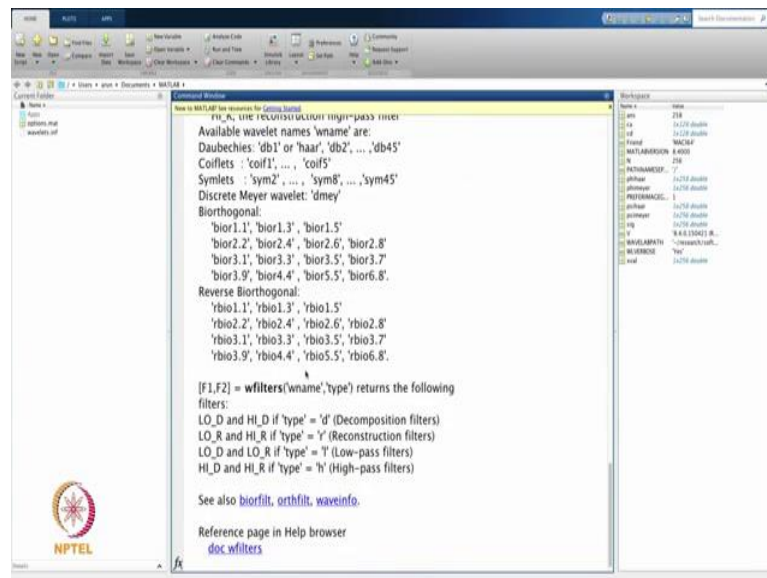
Reference page in Help browser
doc dwt

>> [ca,cd] = dwt(sig,'haar');
>> subplot(311)
>> plot(sig); axis tight
>> shg
>> subplot(312)
>> N = length(sig);
>> stem(1:2:N,ca,'r');
>> shg
>> axis tight
>> shg
>> subplot(311)
>> subplot(313)
>> stem(1:2:N,cd,'y');
>> shg
>> axis tight
>> shg
>>
```

And if you look at the syntax it says, it performs multilevel 1D wavelet decomposition. And you have to pause the signal X specifies the level of decomposition that the you want to go down to and of course, the wavelet name. Alternatively you can also specify the low pass and high pass filters. How do you get the low pass and high pass filters?

Well, there is a command called W filters, which gets you the filters associated with these different families. For whatever families the filter exists the W filter will be able to get a filter coefficient is Daubechies, Coiflet, Symlets. Or if you want to use Bi-orthogonal wavelets or reverse Bi-orthogonal wavelets and so on.

(Refer Slide Time: 01:16)



How to MATLAB has resources for getting started

Available wavelet names 'wname' are:
 Daubechies: 'db1' or 'haar', 'db2', ..., 'db45'
 Coiflets: 'coif1', ..., 'coif5'
 Symlets: 'sym2', ..., 'sym8', ..., 'sym45'
 Discrete Meyer wavelet: 'dmey'
 Biorthogonal:
 'bior1.1', 'bior1.3', 'bior1.5'
 'bior2.2', 'bior2.4', 'bior2.6', 'bior2.8'
 'bior3.1', 'bior3.3', 'bior3.5', 'bior3.7'
 'bior3.9', 'bior4.4', 'bior5.5', 'bior6.8'.
 Reverse Biorthogonal:
 'rbio1.1', 'rbio1.3', 'rbio1.5'
 'rbio2.2', 'rbio2.4', 'rbio2.6', 'rbio2.8'
 'rbio3.1', 'rbio3.3', 'rbio3.5', 'rbio3.7'
 'rbio3.9', 'rbio4.4', 'rbio5.5', 'rbio6.8'.

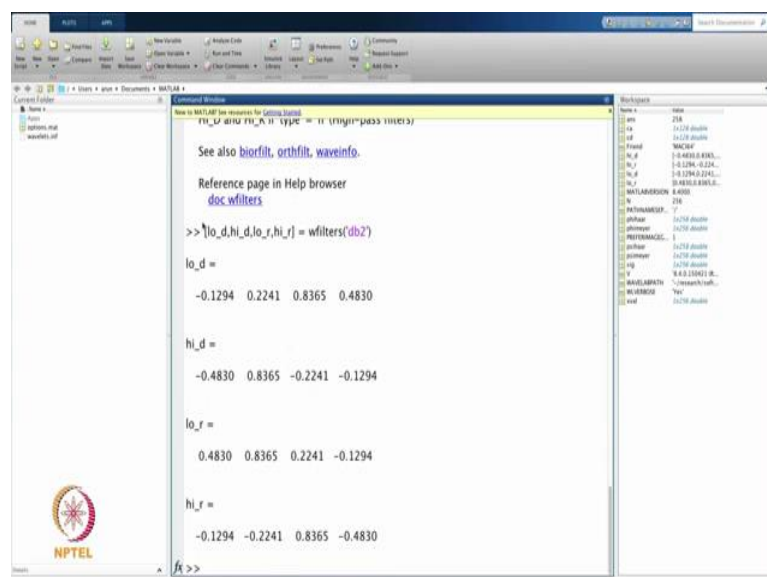
[F1,F2] = **wfilters**('wname','type') returns the following filters:
 LO_D and HI_D if 'type' = 'd' (Decomposition filters)
 LO_R and HI_R if 'type' = 'r' (Reconstruction filters)
 LO_D and LO_R if 'type' = 'l' (Low-pass filters)
 HI_D and HI_R if 'type' = 'h' (High-pass filters)

See also [biorfilt](#), [orthfilt](#), [waveinfo](#).

Reference page in Help browser
[doc wfilters](#)

So, for example, if I want the filters associated with that is a analysis and synthesis filters associated with Daubechies 2.

(Refer Slide Time: 01:40)



See also [biorfilt](#), [orthfilt](#), [waveinfo](#).

Reference page in Help browser
[doc wfilters](#)

```
>> [lo_d,hi_d,lo_r,hi_r] = wfilters('db2')
lo_d =
-0.1294    0.2241    0.8365    0.4830

hi_d =
-0.4830    0.8365   -0.2241   -0.1294

lo_r =
 0.4830    0.8365    0.2241   -0.1294

hi_r =
-0.1294   -0.2241    0.8365   -0.4830

>>
```

Then, I you use these syntax, say d b 2. And you can see, it has given may be h of n, that is low underscore d. High underscore d which is g of n. And low underscore r is the low pass filter and reconstruction low pass filter. But, we have already seen these filters are orthogonal wavelets. Therefore, the synthesis analysis filters are going to be identical. Therefore, low underscore r is actually, it what it has done here is, it is given a reflected

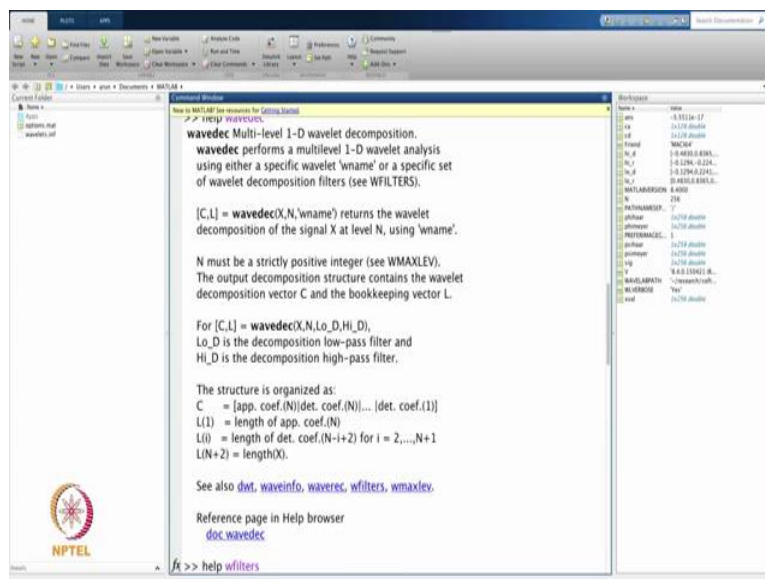
version.

In fact, this is slide reflection there g tilted of n is not exactly g of n , but reflected version and that is what you see here. And high underscore r , r is a reflected version. Because, when you implemented you will have to be implemented to the reflected versions. And that why given me that. But, you can verify if these filter satisfies the conditions here.

For example, if have been design correctly the some of the filter coefficients should be root two and that is what it is? For the low pass filter, for the high pass filter what should be some of the coefficients? It is a high pass filter. Therefore, the some should be 0. Because, g of ω should be 0 at ω equals 0. And g of ω at 0 frequency is nothing but, the some itself.

So, if I look at the sum of it is 0 up to working position. So, you can where I if these for other filter as well. Now let us get back to the wavelet decomposition. We will not supply the filters, we will directly specify the name of the wavelet that we want to use. And we shall use again the haar wavelet here, just for the purpose of demonstration.

(Refer Slide Time: 03:35)



Now, what does wave dec returns? It returns the coefficients and a vector or an array called denoted L. What does this L contains? First of all what is vector C contain? It contains the vector of coefficients arranged in the order that I had describe in the previous lecture. If I performed two level decomposition I have a 2, d 2 and d 1. So, it is

going to arrange in the sequence a_2 followed by d_2 followed by d_1 .

And if the signal is of length n a_2 will have a length $n/4$, d_2 will have a length $n/4$ and d_1 will have a length $n/2$. And those lengths are given return in these vector L ((Refer Time: 04:21)). So, let me show you that. So, let say get this $c_a d$; that means, essentially it contain an approximation and detail coefficients. And L now perform a two level decomposition here of the signal.

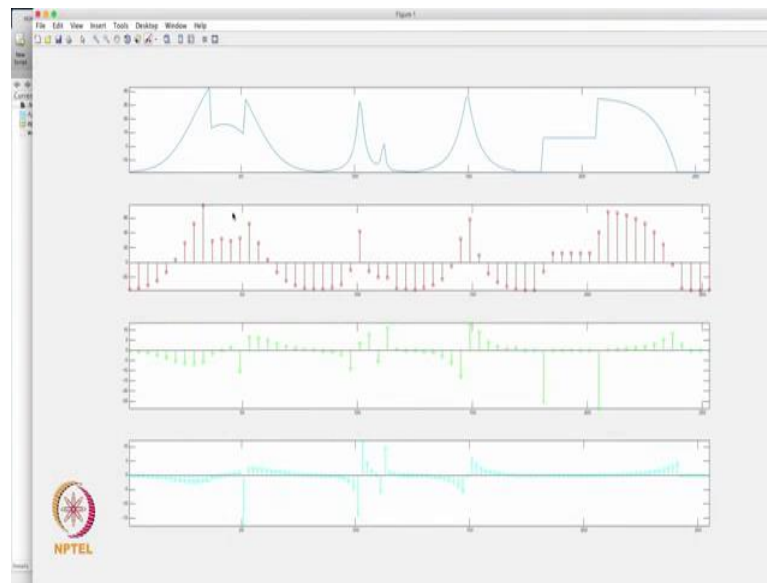
So, have done that. And now let us look at L . So, what it as return is 64, 64, 128 and 256. Now what happens here? Why is it given me 4 whereas around three scales here. The last one corresponds to length of the signal. The last number in L corresponds to the length of the signal. So, strictly speaking only the first three are of interesting to me.

And let us looks at the cad . cad is a big vector of length 256 naturally I have 64 plus 64 plus 128. So, when I plot here. Now, I would like to plot of course, the signal and its coefficients. Let us go through this bit $t d s$ exercise, but we will skill do it. So, will plot the signal. And then, I have subplot 412, where I will plot the approximation coefficients. And here will stays stem, once again 1 to not to anymore.

Because, now the resolutions as follow in my factor of 4. So, 4 to N and what do I want C of 1 to L of 1. Because, L of 1 tells me what is the length of the first set of coefficients. And let us plot is in that cad . So, let see the plot is coming out alright. So, let us plot the next coefficients. And here which should be L of 1 plus 1 to L of 1 plus L of 2 those are the coefficients I require. And the resolution for d_2 is also $N/4$ and we shall use a green color for this. And finally, in this case the resolution d_1 is at resolution of $2 t s$.

And here I shall use L of 3 minus 1, L of 3 is 128. So, L of 3 plus 1 128 L of 3 plus 1 to luckily I have this signal length stored in the last element and that is L of 4. So, let me use a on that color here, let us use a sign color.

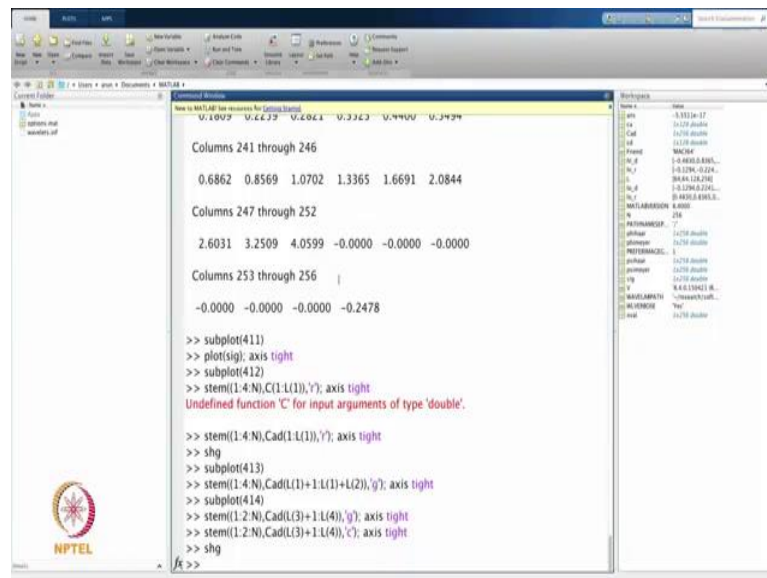
(Refer Slide Time: 07:51)



So, now I have all the coefficients. So, on the top I have the signal and in the second panel I have a 2 at a resolution of t by 4. But, sampling interval of $4t$ s and detail coefficient d_2 at a sampling interval of $4t$ s. And detail coefficients are the first level, these are the same that we have seen earlier d_1 does in change. What does change is, now a_1 has been split into a_2 and d_2 .

Now, these are the decomposition and the approximation and the detail coefficients. In fact, something interesting the dwt has a nice property, that it preserves the energy of the signal. What does it mean? That if I square these number in coefficients in each of these bands. And add up the energies of the coefficients in each of these bands. I should require the same square of the signal itself. So, let us see if that is true.

(Refer Slide Time: 08:54)



```

Columns 241 through 246
0.6862 0.8569 1.0702 1.3365 1.6691 2.0844

Columns 247 through 252
2.6031 3.2509 4.0599 -0.0000 -0.0000 -0.0000

Columns 253 through 256
-0.0000 -0.0000 -0.0000 -0.2478

>> subplot(411)
>> plot(sig); axis tight
>> subplot(412)
>> stem(1:4:N,Cad(1:L(1)), 'r'); axis tight
Undefined function 'C' for input arguments of type 'double'.

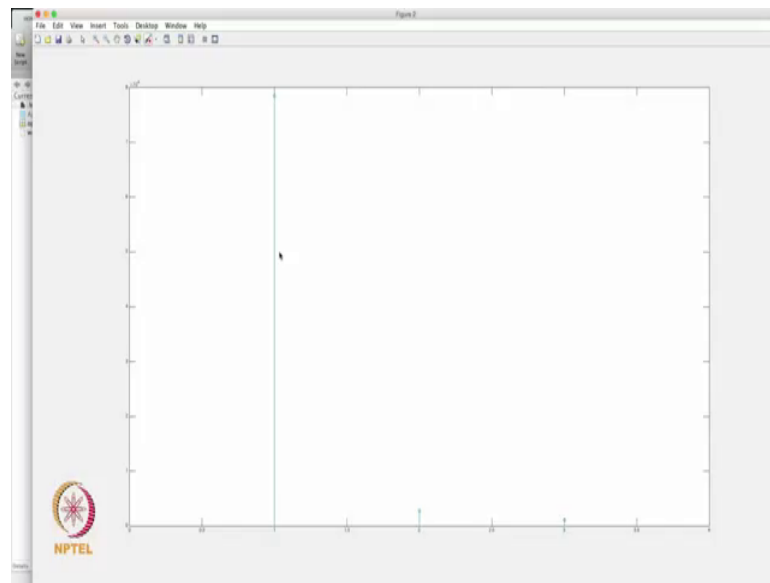
>> stem(1:4:N,Cad(1:L(1)), 'r'); axis tight
>> shg
>> subplot(413)
>> stem(1:4:N,Cad(L(1)+1:L(1)+L(2)), 'g'); axis tight
>> subplot(414)
>> stem(1:2:N,Cad(L(3)+1:L(4)), 'y'); axis tight
>> stem(1:2:N,Cad(L(3)+1:L(4)), 'c'); axis tight
>> shg
>>

```

So, let say a 1 s q. So, this a 2, so a 2 s q. So, this is the some square of the approximation coefficients in a 2. And likewise d 2. And this is the some square at d 2 coefficients. And finally, we have d 1. So, this is what I have for d 1. And as you can see, the sum squares at decreasing. What is the sum square tell me? It tells me, it give me an idea of the energy of contain in the signal in that frequency band. Each scale corresponds to the frequency band.

Approximately these frequency bands are zero to one fourth of omega max. And then, one fourth of omega max to half of omega max for d 2. And then, half of omega max to omega max which is ((Refer Time: 10:50)) the frequency for d 1. So, this clearly tells me, if our have to simply draw these numbers in just as a stem plot you can see that...

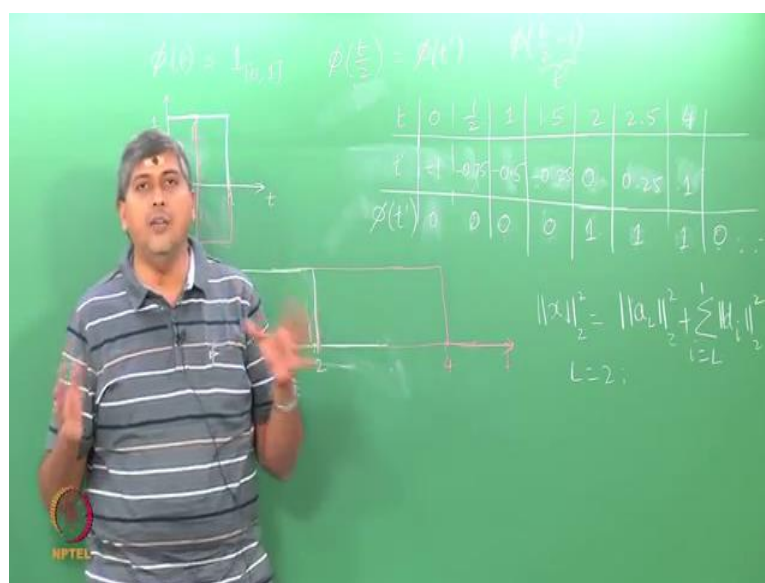
(Refer Slide Time: 11:19)



So, these are the three point here, let me actually make it more visible for you. Let me said the exclamatory. Such that, you are able to see the plot clearly. The first spy here corresponds to these sum square of the a^2 . And the second corresponds to the sum squares of d^2 . And the third here corresponds to f sum square of d^1 . It is very clear, that most of the signals energy.

First of all as I told you, the sum of these three should get me the energy of the signal. Let us quickly verify that and come back to these figure ((Refer Time: 12:00)). So, the sum of a^2 , this is the sum square of the a^2 square plus b^2 square plus b^1 square. And now let us add up.

(Refer Slide Time: 12:40)



So, what we have shown is the energy is preserved. In other words, if x is your signal it is a vector. So, sum square of these x is if a perform l level decomposition. Then, it is a l sum square two norm of a l plus the square two norm of a l plus as these sum square two norm of detail coefficients of i equals L to 1 are you can say 1 to L . I they strength from L to 1 . So, that when you expand is you will get d_1 first and then you will get d_1 minus one up to d_1 .

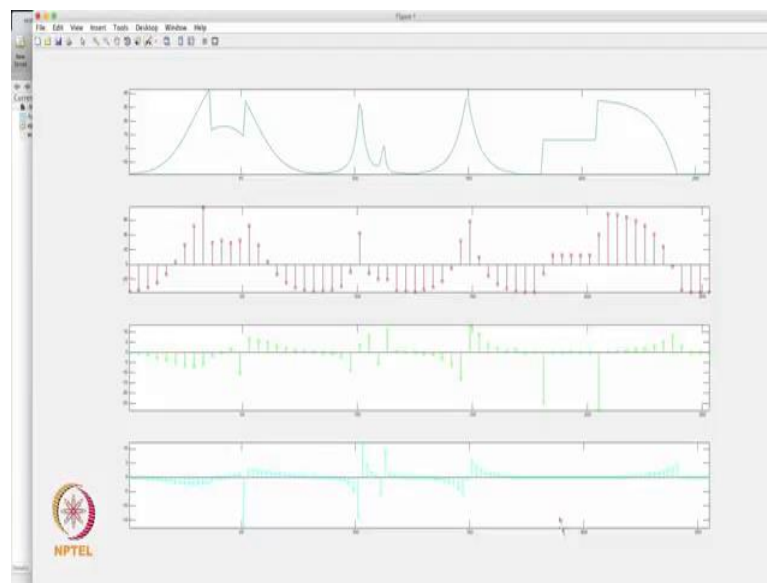
This is a very beautiful way of decomposing the energy of the signal into the respective frequency bands. Of course, this was even true for the Fourier case we have seen by virtual perusal theorem. But, this is nice also, because now I am able to collect the frequencies in to a single band. And know I now clearly at least for this example we have chosen L to be 2 we have seen that are more than 80 percent. In fact, we can get the extract number, as significant fraction of the energy is surely contain in the approximation coefficients.

How much of the energies contain in the a_2 alone, that is the frequencies band 0 to ω_{\max} by 4 . I can clearly do that I can say well I can compute the fraction here a_2 square 95.4 percent of the signals energy is contain in these frequency band. Now, for those of few familiar with principle component analysis. This is similar to principle components analysis for single signal, where we have decomposing the signals variants or energy in to different bands which are non overlapping, which are orthogonal.

Even in principle component analysis we do that. But, in the multivariate sense. Where we decomposes the variants covariance of the a signal of the set of multivariate signals that we have in to what are known as principle components. And the variants of the total variants of the multivariate signals can be expressed as the some of the variances of the principles components themselves.

So, that is the similarities here what is p c a for multivariate signal is dwt for the univariate signals and that sense. Then, many researchers have exploited these parallelism between p c a and dwt and returns quite a few nice articles on that. So now, why this is information of use to me where I can user information. Let us go back to the plot here, this plots of course, clearly tells me that I can really work with the coefficient in a 2 alone.

(Refer Slide Time: 16:02)



If I look at this plot here. How many coefficients do I have in a 2? In the signal I had 256 coefficient, in a 2 had I had 64. Now, I know that a 2 contains 95.4. Let us as say for the discussions 95 percent. 95 percent of the signal energy is contain in these a 2 coefficients which are 64. From a signal compression view point, this is excellent, if I am willing to going for the loosely compression. Then, I need to store only these a 2 coefficients, which is 64 where as the original signal is 256 numbers belong.

Therefore, I have achieve the comparison factor of four. So, suppose I adapt that compression strategy, where a 2 will now be used instead of the signal. Then, of course it

is a loosely compression. But, if I recover this signal from a 2 alone do I miss out on any key features. Obviously, I am going to miss out on some features. Because, if you look at the plot here, what I am doing is, I am really throwing away d_2 and d_1 .

And the information contain here due to this picks may be lost. Not completely though. But, depending on how these features are important you are not, you cannot read them at significant. But a part of the information contain here at these pikes will be loss. And we would like to know, how much of it lost? How can I know well what I do is? I just reconstruct the signal from a 2 alone and that to tell me, what my compression has actually loss out on?

And let us do that on, that part here is implemented by wave r e c which performance a multilevel wavelet reconstruction. Of course, if you supply the coefficients that you obtain straight away to wave r e c, it recover the full signal for you. But, what we hear what we are saying here is, I will only store a 2 from a signal compression view point. And I will ask what kind of a reconstructed signal I would get. And then I will have an idea of how good the compression is in terms of restoring the original signal how much losses occur.

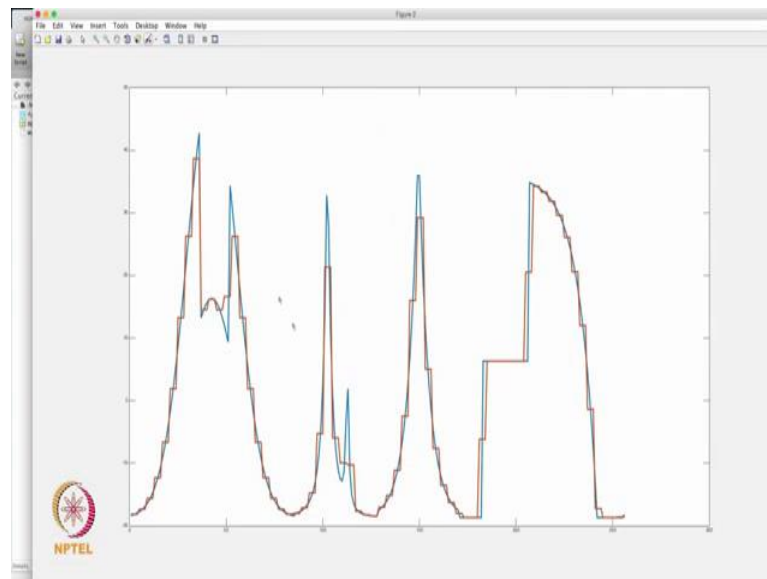
So, for these purpose, where let us say c a d has modified to some thresholds it is as if now I am applying a certain threshold. What I am doing is? I am saying all the coefficients in a certain scale are going to be 0, which scales d_2 and d_1 . So, I am going set here first of all I going to create a new cad the coefficient vector to this. And then, in this I am going to set all the coefficients from 65 to end. That is whatever is there is in d_2 and d_1 to 0. And then, I am going to use this for reconstructing the signal. So, let us called as \hat{x} .

The thresholding here, that is being done is a called a scale based thresholding. What I had done is, I just discarded coefficients simply over certain scales. Very quickly we will learn what is known as a an amplitude based thresholding. Where, I will perform thresholding I will retain are through away coefficients depending on their magnitude.

So, cad threshold here is being supplied. And also what it requires is the length of the coefficients. And the filter that you use for decompositions. Obviously you have specified that and I have my \hat{x} . So, now let us plot here in this figure. Let us plot both the signal and the \hat{x} . And let us force them to column vectors. And see how could

they... Let us have line with 2 or even 3 this just for better visibility.

(Refer Slide Time: 20:37)



So, what has happened here? The blue one is the original signal and the red one here is the reconstructed signal. That is, the signal that you have recovered from its compressed version. The compressed version was 64 coefficients that contain in the a_2 . So, we have recovered most of the signal for shower. Now, you have to ask whether. So, if you look at this point as expected the sharp points have been now flattened out a bit. And they are flattened out differently.

Because, each of these sharp points at different features some of these have strong features in d_2 and d_1 . And some of these have mild features in d_2 and d_1 . Therefore, the signal has been reconstructed with some different errors at these different points. Now, whether these are important to you or not depends on the application. If you think that the sharp points are due to outliers, due to some faulty sensors.

Then, you are willing to let go. But, if you think that the sharp points carry valuable information about the process. Then, you have to store d_2 also with a_2 and d_1 also with a_2 . So now, there you can start seeing the number of combinations that you can think of in signal compression. But, hopefully the basic idea of signal compression then has been conveyed to you. And in this process we have also looked at that discontinuity detection.

Now, let us actually quickly review, what an amplitude based thresholding would do? An amplitude based thresholding has explained earlier, would threshold coefficient. That is zero out the coefficients, which are below a certain thresholding in magnitude. That is the idea of the amplitude based thresholding. And then, there are variation in that some estimation or some compression algorithms. The basic idea and compression estimation is same.

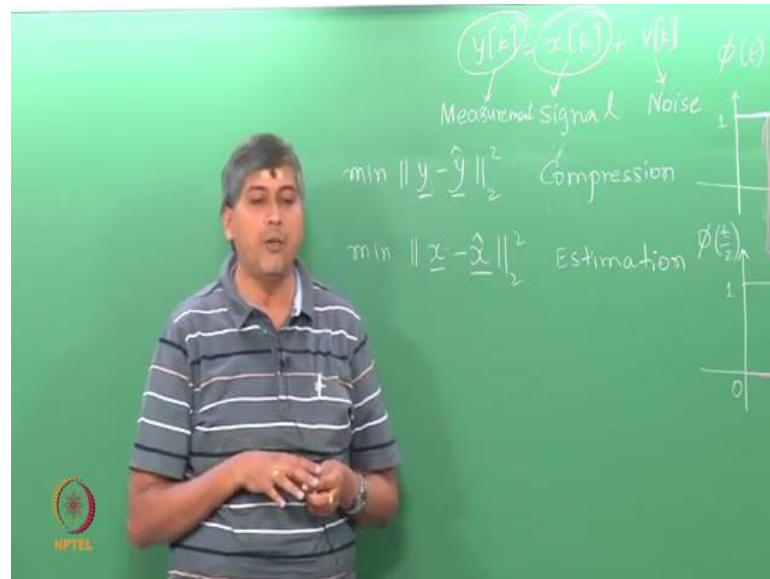
For example, here we discussed from signal compression view point that d_2 and d_1 can be thrown away. Because, they contain insignificant amounts of energy. If I am a person who is working in signal estimation I can treat d_2 and d_1 as due to some unnecessary features in the signal. Undesirable feature, these could be due to noise or whatever. In that case throwing away d_2 and d_1 amounts to throwing away noise and I know already that, this is the what is the noise characteristic.

So, I should know a priori what are the signal and noise characteristics roughly. And then, I apply a threshold. Once again, applied scale based threshold in here. But, I could apply an amplitude based thresholding. And as I said in amplitude based thresholding, there are different flavors. I can say, that I will not touch the approximation coefficient at all. Because, they contain global features of these signal.

And I will only apply threshold to the detailed coefficients. And within these there are different variants and so on. So, the number of different types of combinations of these vanilla idea is just infinite. But, the basic idea is performed decomposition, applied thresholding and then recover the signal. In comparison you do not recover the signal. You only store the coefficients. You only recover when it is needed in signal estimation you do recover the signal, that is the primary difference.

And of course, the objectives are also slightly different it make the big difference. In compression I would like to recover the original measurement as much as possible. Whereas, signal estimation I would like to recover the true signal.

(Refer Slide Time: 24:27)



So, let me explain that you on the board. So, assume that I have a measurement y which is made up of two components in signal estimation. Where, x is a signal and v is some kind of a noise and y is a measurement. So, in single estimation my goal is to recover x . Whereas, in compression I would like to recover y as much as possible. So, the objective is different, but the principles is a same. In compression and estimation both what I am going to do is perform a wavelet decomposition of y . And then applied thresholding.

Then, the difference is in how you apply this thresholding, in signal compression I would like to the goal is to store the signal in as when you few coefficient as possible. While still minimizing this norm here. In fact, let me is say here y minus \hat{y} here. Where, y and \hat{y} had are vector y is the measurement vector, \hat{y} had as recovered vector that from your compressed version. This is the compression objective in estimation the objective is this.

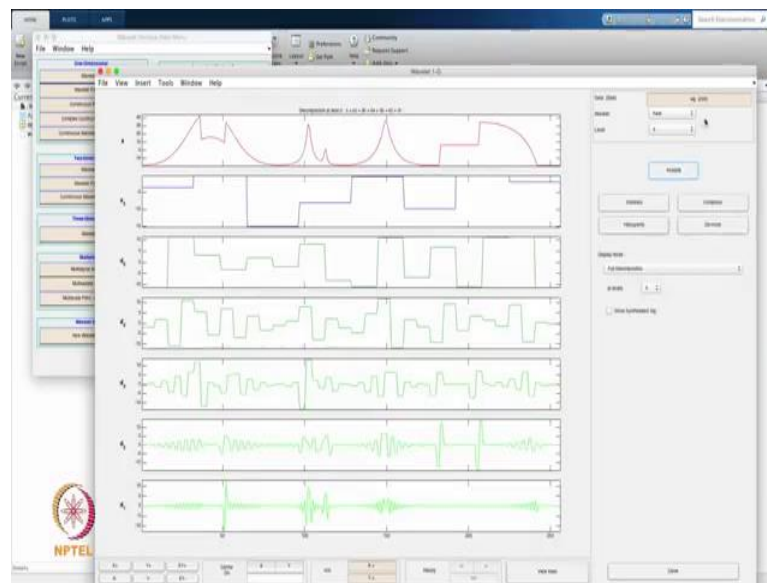
Of course, the big difference in estimation is I do not know what is x . Of course, then you can show that when you adapt a certain estimation algorithm, what kind of norm value due you get and so on. But, as you can see the compression goal is different from the signal estimation. But, both uses the same algorithms. So, I shown you how signal compression is done? You can add noise to x and then repeat the same exercise and I shown how to scale based thresholding is done?

But, you can repeat the exercise with an amplitude based thresholding I am not going to show you the amplitude how to perform the amplitude based thresholding from the

command line. Very quickly let us go through the menu the g u i that the wavelet toolbox comes with which will allow you to flavor with the number of different thresholding techniques. And I am just going to show you couple of examples here.

So, let us bring up the g u i which is brought a typing wave menu. And where are seeing these before. Now, we pick wavelets 1D by the way, if we want to simple have a display of wavelets, you can go here and get the display of wavelets. But, right now it interested in d w t.

(Refer Slide Time: 27:37)



Let us pick the one dimensional wavelet transform. And one thing that I strongly, very strongly recommend is to you go through this many, many examples that is toolbox. There are number of example that you can work out with it is obviously impossible for me to go over for all of these. So, let me show you here how to import the signal from the work space. And we would like to actually import these signal, but we have generated.

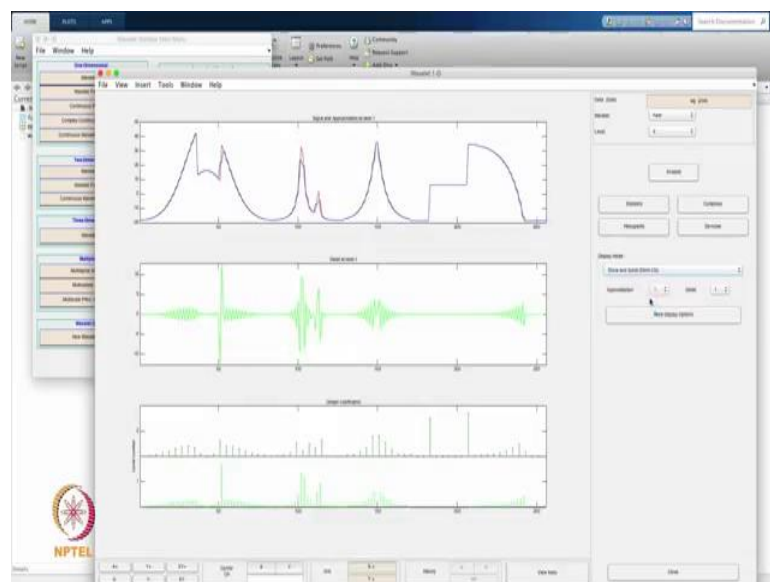
So, this the signal that we have generated. And now we would like to perform dwt. We have performed haar wavelet analysis. So, the default here is haar on the right panel you can chose the wavelet. And the levels to which want to perform. So, we have chosen the haar wavelet. And I have perform five level decompositions. But, to generate the same thing that you have seen hear earlier. We perform the two level decomposition.

Now, a big caution here which is the notation that is used here unfortunately the notation, that is used here is inconsistent strength with general literature. The small a , the small d is here are not the coefficients unfortunately. There actually the reconstructed signal themselves. And if you want the coefficients, then go here there are different ways of looking at the decompositions. That is the very nice feature here you can look at a full decomposition by the way.

How do you know that these are reconstructed components. Well, look at the top it is says the signal s is the sum of a^2 plus b^2 plus d^2 . And that is true only if these a s b s and d s are reconstructed components. You cannot add the coefficients. So, it is cleared that these are reconstructed signals. Reconstructed signals are to be treated as filters signals.

But, for most of the analysis that is done in the literature and application in the literature. Normally one works with coefficients, not reconstructed signal. In fact, the energy decomposition that we wrote was for the coefficients, not necessary for the reconstructed signal. So, you have to keep that in mind.

(Refer Slide Time: 30:04)

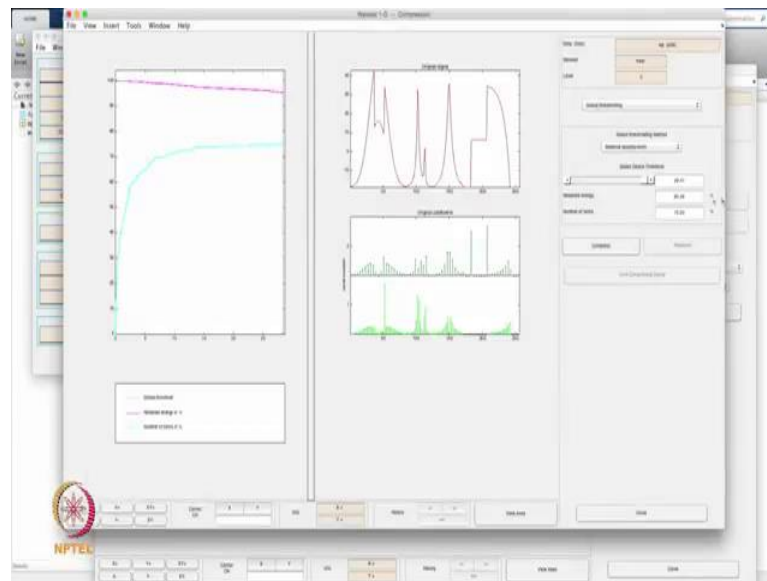


So, how do you get my coefficients well, to a certain extent you can get that by going to the last option here. Where, it shows the reconstructed signal in a particular band. And the coefficients here. So, the bottom one here shows the detail coefficients. In fact, again the problem there is a red shows magnitude of the detail coefficient. It does not show the

absolute coefficient. So, some of the features of these g u i eyes are not show user friendly unfortunately at least in these version.

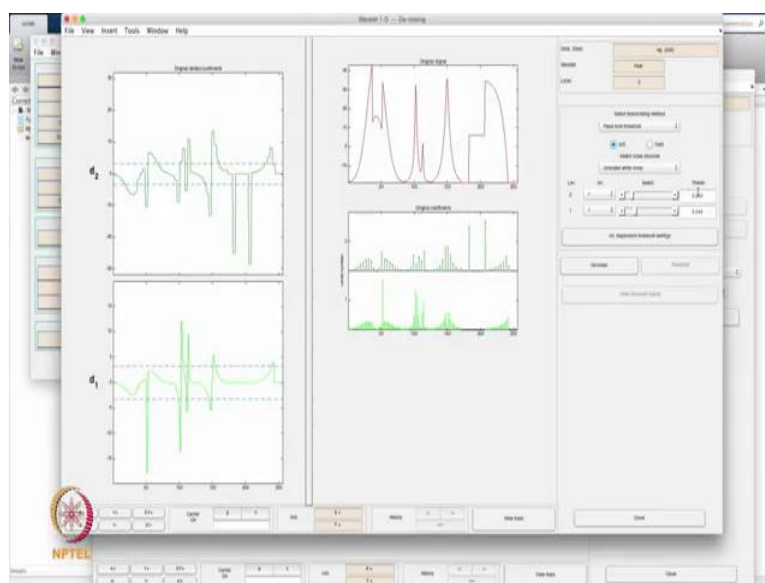
But, some of them are and if you want to do a manual analysis yourself. Then, you can expert for example, you can export to the work space the coefficients. And then, you can work with those coefficients. But, the nice thing here is of course, I can do a compression and I can do a de noising.

(Refer Slide Time: 30:55)



So, for example, hitting on the compression tells me. How changing the threshold will change the energy and so on. And I ask you to explore this option let us quickly look at the signal estimation.

(Refer Slide Time: 31:13)



So, the estimation part here is obtained by de-noising. Now, what is happening here? What is this plot showing me? Well, it is showing me the thresholds that it has calculated. How are the thresholds calculated? There is a procedure, this is a very extensively studied piece of work, that is thresholds have to be calculated. The basic idea said is to take a threshold, applied to the coefficients in each band and there are again two versions there.

One version which is called hard thresholding. What it does is, it finds that if the magnitude of the coefficient is below or threshold it will be set to zero or retain the coefficient as is. The other procedure known as soft thresholding, will not only reduce the coefficients to zero, that are below the threshold. But, also swing the coefficient that are above the threshold by the magnitude of the threshold itself. That is called a soft thresholding and that is called shrinkage.

So, there are these two versions: hard thresholding and soft thresholding. And those are the two options that you see here: hard thresholding and soft thresholding. The default here is set to soft. Both have the disadvantages and advantages and I will quickly talk about in the short lecture I have after this is just a summary of what we have discussed. So, you can now choose different thresholds if you want; it is applying the same threshold.

How did it calculate the thresholds? I will tell you how it calculated the thresholds. There is something called a universal threshold, that is given out by Donoho. Thresholds are typically denoted by λ in wavelet literature.

(Refer Slide Time: 33:01)

HT: $|d_m| < \lambda, d_m = 0$

ST: $|d_m| < \lambda, d_m = 0$

$|d_m| > \lambda \rightarrow |d_m| - \lambda$

$\lambda =$

NPTCL

And it is given as $\sigma \times 2 \sqrt{2 \log n}$. Where, n is a length of a signal and σ is a standard deviation of the noise that you. How do you estimate the σ ; obviously, I do not know different. This is known as a universal threshold. This σ is calculated based on the energy of the detail coefficients in these highest frequencies band.

And there are again variations to that idea, but if you considered the noise to be primarily in the high frequency's band. Then, the energy contain d_1 square would be an estimate. So, $\hat{\sigma}^2$ is nothing but, this itself this is a rude estimate of σ^2 . And the square root of this will give me σ . This is assuming that the noise is white, what we mean by white noise is un correlated noise in time. And that primarily the noise is the high frequency band.

Then there are variants for this depending on whether the noises, correlated, the structure of the noise, whether is colored noise and so on. And so abusively not going to that extensively. But, this is the basic or you can say similar work in thresholding base methods is called universal threshold. And what is done is in hard thresholding, if any coefficient in at a certain scale, at a certain level. If this is less than λ , then d_m is set to 0 this is hard thresholding.

And you can applied this, if you need it to be approximation coefficient as well. But, normally one does not touch the approximation coefficients. Because, that is believe to

be containing the signal features. But, you can apply there is absolutely nothing preventing you from doing that. In fact that is how different PhD and masters this is about trying out different options. And this is the hard thresholding and the soft thresholding what it does it in addition to this, it says for those which are when I say d_m here.

In fact, all the coefficients at any scale m . So, you should understand. In fact ((Refer Time: 35:42)), you can think of yourself. So, for those coefficients which are greater than λ_m , there are said to be d_m minus λ_m magnitude. So, they are shrunk to this d_m minus λ_m well in magnitude they are stuck. You can say that, first the magnitude is taken and then d_m is modified to this and then the sign is retained. So, that shrink by that magnitude, this called soft thresholding.

And then, how you applied this thresholds is that thing. So, first step is decomposition, the second step is computation of the threshold. And the third step is how you applied the threshold? Do you applied these threshold to every scale are to only select scales to half of the scale there is something called interval thresholding, I will just quickly go an example on interval based thresholding and so on. All of that depends on your signal and noise structure them. The basic idea; however, is to perform decomposition, determine the value of threshold and choose a thresholding method.

And when you have a different combination of this, then you have at least above 20 to 30 different thresholding methods. And there is absolutely no universal answer as to which method is best. It all depends on this applications itself. So, here what I am performing here is the universal thresholding. And it is showing me the thresholding limits here for d_2 and d_1 . It is showing, which coefficient said to zero.

And then, you can say here d noise and apply soft thresholding this what you get? The soft thresholding give you a nice, smooth, estimate. But, I can choose a hard thresholding if I wish and look at what the hard thresholding has done. If you have notice carefully let us go back to the soft thresholding. And I have these estimate here. The estimate here is shown in blue on the top and compare this with what you get in a hard thresholding.

What you have noticed hard thresholding retains the hard the soft features. Whereas, the soft thresholding smoother than the soft feature. And that because it is shrinking the coefficients which are significant as well. That the prime difference, if you think that the

soft feature have to flaterend in you are estimate then use a soft thresholding. If you think they should be retained as they are then use the hard thresholding, that is the basic idea.

Then, of course you can chose the way that thresholding is calculated and so on you can chose a number of different ways. But, I will not going to that what... So, these are the different thresholding ways of calculating here. But, I am not going to going to that and of course, what also matter is a choice wavelet. So, if you look at all these combinations that you have as I said there are in enormous number combinations. And I will give a reference paper which as perform an extensive study on the different choice of wavelet, different choice of wave thresholding methods and the different ways of calculating the threshold and so on.

So, I will quickly give you that reference. But, let me just finally, go through an example to interval based thresholding and then will conclude the lecture on this the discussion with the very brief lecture may be for about 10 minutes.