

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

Lecture – 5.3
Practical aspects of STFT

Welcome to lecture 5.3 of the course on introduction to time frequency analysis and wavelet transforms. Now, what we are discussing in this unit 5 is essentially the topic of short time Fourier transform. In the previous 2 lectures we have examined the theoretical properties, the definitions, and so on. Now it is time to look at the practical aspects. Of course, I have shown you how to implement the short time Fourier transform in mat lab, if not in detail, but at least I have given you a quick preview of what routines are available.

In this lecture, we are going to talk about the practical aspects of short time Fourier transform, in other words, the implementation itself. The objective, therefore is to talk of this discrete short time Fourier transform. Now, this is an analog of the discrete Fourier transform that we studied earlier. Remember, in the case of Fourier transforms, we went from continuous time Fourier transform to discrete time Fourier transform, and then talked about discrete Fourier transform.

(Refer Slide Time: 01:22)


Lecture 5.3

References

Objectives


To learn and study the practical aspects of STFT:

- ▶ Discrete STFT
- ▶ Effects of window length and type


NPTEL

Arun K. Tangirala, IIT Madras

Short-time Fourier Transform



2

But here we are making a big jump from continuous time short time Fourier transform to discrete short time Fourier transform. So, we are avoiding this intermediate one, where we would discuss the discrete time short time Fourier transform. As expected as we saw in the case of discrete Fourier transform, we would be discretizing the time and frequency axis here. In the case of DFT, we only discretize the frequency axis, because that was only variable.

So, we are going to discuss very briefly the definition of discrete short time Fourier transform, and especially the mat lab routines that are available to do that. The second aspect that we are going to talk about is the effect of window length and also the window type, because this plays a critical role in the ability of the short time Fourier transform to extract the time frequency varying features of the signal.

(Refer Slide Time: 02:20)

Lecture 5.3 References


Discrete STFT

The **Discrete STFT** (also known as the Gabor transform) is given by

$$X[m, l] = \langle x[k], g_{m,l}[k] \rangle = \sum_{k=0}^{N-1} x[k] w[k-m] e^{j \frac{2\pi l k}{m}} \quad (1)$$

where $g_{m,l}[k] = w[k-m] e^{j \frac{2\pi l k}{m}}$ is the discrete windowed Fourier basis or atom.

Note: Increments in m determine whether one achieves a **redundant** (overlapping windows) or **orthogonal** representation.



MATLAB: WindowFT, tfirstft, tfrgabor

Arun K. Tangirala, IIT Madras
Short-time Fourier Transform
3

So, the discrete time short time Fourier transform is defined as I have shown here in equation 1. As I just mentioned, the idea is very straight forward, pretty much similar to what we see in DFT. In DFT we had only a single variable frequency, and we talked about how to discretize this frequency grid, essentially sampling the frequency. Now, the short time Fourier transform is a 2 dimensional function. It is a function of time and frequency. Particularly, what we mean by time here is the center, time center of the window and the frequency center of the window.

In the previous 2 lectures, we had held these 2 quantities that is the time center and the frequency center as continuous valued variables. Now, in practice, I cannot evaluate

them, evaluate the short time Fourier transform over a continuum. So, I have to discretize these variables, τ and χ ; and the discretized ones are now represented by this grid points m and l . So, you can think of τ being now discretized. And typically the discretization interval for τ , the finest interval, it has to be a sampling interval itself. That is what we mean by discretization is of the τ is that essentially how I am going to march ahead in time with the window; that is what is kept track by m . And likewise, how I am going to march ahead in frequency is kept track by l . And the smallest value for m is 1 itself. Remember, m is not an continuous valued quantity; it is an integer; it is essentially telling how many samples in time I am going to march forward. If m is 1, then I am going to move the window 1 interval, 1 sample at a time.

And likewise, l also corresponds to the number of points, that is it keeps track of the grid point on the frequency axis. So, the understanding should be fairly straight forward here because one side window I am doing a DFT. So, you can think of l being this way integer or that keeps track of the grid point on the frequency for the DFT of the windowed segment.

By definition, the short time Fourier transform is an inner product between the signal; and now the discretized time frequency atom; earlier we had this as a continuous time frequency atom. And we had $g_{\tau, \chi}$ as a continuous function of time t , now I have $g_{m, l}$ as a function of the discrete time index k . And as in DFT, I restrict my summation from 0 to $n - 1$, exactly the way I did in DFT. So, there is not much of difference.

In fact, the only difference between DFT and discrete short time fourier transform, DSTFT, is that the presence of this window here. Again, I had $w(t - \tau)$ previously for the continuous time short time Fourier transform, now I have $w(k - m)$; m is once again keeping track of the grid point of τ that is of the center of the window. So, that is it does not much more to discuss.

Of course, the same connotations apply here. The most important thing being when what we learnt in DFT, the moment I compute the discrete Fourier transform of a given sequence of data, implicitly I am assuming that the underlying long signal, infinitely long signal that I have not observed is periodic with the same period as a length of the observation. Here also when I am evaluating discrete short time Fourier transform I am

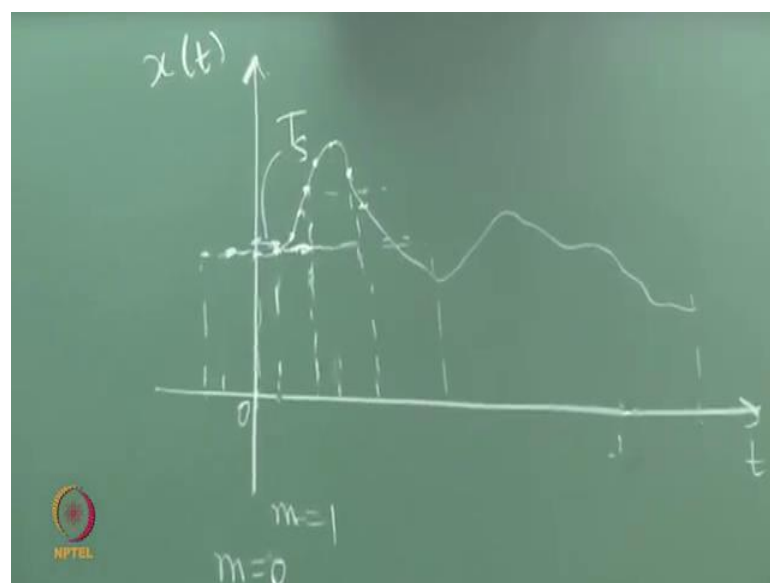
assuming that the signal that has not been given to me which is the infinitely long one, is actually periodic of the same interval as the observation interval itself.

Now, we had mentioned this point about redundant and compact representations. The m parameter controls whether you will generate a redundant or a compact representation. Remember, what you are doing in discrete short time Fourier transform is, you are going from this numbers x_k which is in the one dimensional space, to numbers X which is a function of m comma l . It is in a 2 dimensional space, so you are going, you are representing a 1 dimensional signal in 2 dimensional plane.

Obviously, you will run into redundancy; means, you will be using more numbers than necessary. But, whether you are using more numbers than necessary or not, is actually governed by the choice of m the user makes; m , for marching, you can remember in that way. If your marching is 1 interval at a time, then you will definitely generate a redundant representation, because what you are doing is, the moment you move your window 1 interval ahead in time, you will be recalculating a lot of calculations that you have already done with the previous windowed segment, because there is a huge overlap between the previous segment and this segment.

If you choose the m proportional to the window length itself, so that the next center location has no overlap, is such that the resulting segment has no overlap then you will generate a compact representation. Let me just illustrate that on the board for you.

(Refer Slide Time: 08:13)



Again here, let us just consider some wave form. I am still going to consider a continuous time wave form just it does not make any difference to our illustration. So, let us say this is the wave form that I have, and I have observed over a period of time here. What I am talking about with respect to m is, let us say, that you place your window initially here at this point 0. So, m is 0, or you can say m is 1 depending on how your software keeps track of this m . So, let us say m is 0 here.

And my window, it is let us say, rectangular window; so center of the window is located here. The question is now how do I move this window forward? How do I march ahead? And let us say, I have samples at these points in time and so on. And this interval is my sampling interval. If I move the window such that the next center is at the next sample, then I, this is my new window location. So, now m is 1; in fact, sorry, so m is 0, now m is 1 and so on.

So, the difference between 2 successive values of m will determine whether you are generating the compact or redundant representation. So, you can see clearly if I march this way, this segment contained by this window has a huge overlap with the segment captured by this window. However, if I march ahead in such a way that the new center, let us say, the window has exactly covered this much sample, now my window will should start here, the next window should start here; and let us say, the window here is 5 samples long, right. So, there are this many samples here, is 5 samples long just for the sake of discussion.

Then, I move the window here such that now the center of the window is some way here. I am not drawing it exactly according to this sampling interval here, but you should get the idea. Now what happens is this new window segment will contain completely different part of the signal that is not seen by this window. If I march ahead this way then I will generate what is known as a compact representation. This is also going to be the idea in discrete wavelet transform. So, it is important to get it right.

So, when we discretize the τ in such a way, and march ahead in such a way that I generate compact representations then what we are working with are known as orthogonal family of basis functions. That is, orthogonal discrete windowed Fourier basis. It is as if I have this family of discrete windowed Fourier basis which are placed in such a way that there is no overlapping between them. Then, I am essentially projecting

the signal on to this orthogonal family. So, we will see similar things happening when we go from CWT to DWT.

Now, the MATLAB routines that are available to me for computing this discrete short time Fourier transform are `window ft`; this comes from `wave lab` and `tfr stft`. Of course, this is mostly used in generating a very high and dense short time Fourier transform. So, I would not recommend this if you want to compute discrete short time Fourier transform. The better routine for that is `tfrgabor`. The reason for this name, as I have indicated here, the discrete short time Fourier transform is also the discrete gabor transform, or, sorry, the gabor transform itself.

So, it was proposed by Gabor, even in parallel to the short time Fourier transform where Gabor used a Gaussian window. So, here we have not specified the window. Generally, when you read up Gabor transforms you will see that Gaussian windows are the default. So, read up the documentation on `tfrgabor`. In the rest of the illustrations in this lecture we will only use `window ft`; `tfrgabor` comes from the time frequency tool box.

(Refer Slide Time: 12:52)

Lecture 5.3 References

Signal and energy recovery


- ▶ One of the primary uses of a time-frequency transform is to decompose the signal in the time-frequency plane, "clean up" the undesired part and then reconstruct the signal.
- ▶ Discrete STFT facilitates perfect reconstruction:

$$x[k] = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{l=0}^{N-1} X[m, l] w[k-m] e^{\frac{j2\pi lm}{N}} \quad (2)$$

- ▶ Parseval's formula gives expression for obtaining the total energy

$$\int_{-\infty}^{\infty} |x(t)|^2 dt = \frac{1}{2\pi} \int \int |X(t, \xi)|^2 dtd\xi = \frac{1}{2\pi} \int \int S_{xx}(t, \xi) dtd\xi \quad (3)$$

$$\sum_{k=0}^{N-1} |x[k]|^2 = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{l=0}^{N-1} |X[m, l]|^2 \quad (4)$$



Short-time Fourier Transform

4

The signal and energy recovery are fairly straight forward. Essentially, they have the same properties as that of the DFT. In DFT, we were able to recover the signal exactly; and we had a 1 over n in the recovery expression or the synthesis expression for the inverse discrete Fourier transform; and that is exactly what you see here. I have 1 over n. And of course, I need double summation, because I am reconstructing from a 2

dimensional function. And notice that in the reconstruction I have e to the j . And in the analysis, I have e to the minus j .

Very often we talk about analysis and synthesis; likewise, you can say, w_k minus m times this, can we considered as a synthesis function or a synthesis basis function; whereas, w_k minus m times e to the minus j , this is called an analysis function. We should get used to these terms because when we move to wavelet transforms we will talk about analysis filters and synthesis filters, or analysis and synthesis basis functions.

Now, energy preservation is through this equation, exactly again very strong similarity to what we had seen in the case of DFT. The top expression here is just a reputation of what you have seen in the case of theoretical short time Fourier transform, alright. So, please keep that in mind that there is not much to discuss here. It is just that we should be guaranteed that the discretization does not spoil the signal and energy recovery properties of the continuous time short time Fourier transform. These are very important when you want to perform filtering using short time Fourier transform or discrete short time Fourier transform.


(Refer Slide Time: 14:38)

Lecture 5.3 References

Window functions

Several window functions are available for STFT (these are the same functions used to mitigate spectral leakage). Each of the windows below have a width N . The **rectangular window** offers **best resolution**, but **poor** spectral leakage

Window	Function, $w[k]$	6dB BW [bins]	PSL [dB]	SLR [dB/oct]
Rectangular	1	1.21	-13.3	-6
Bartlett	$\begin{cases} 2k/N - 1, & k \leq N/2 \\ 2 - 2k/N - 1, & k > N/2 \end{cases}$	1.78	-26.5	-12
Hanning ¹	$\frac{1}{2} \left[1 \pm \cos \left(\frac{2\pi k}{N-1} \right) \right]$	2	-31.5	-18
Hamming ²	$0.54 \pm 0.46 \cos \left(\frac{2\pi k}{N-1} \right)$	1.81	-42.7	-6
Blackman	$\sum_{m=0}^2 c_m \cos \left(\frac{2\pi m k}{N-1} \right)$ $c_0 = 0.42; c_1 = -0.5; c_2 = 0.08$	2.35	-57	-18
Kaiser ($\alpha = 3$)	$J_0 \left(\pi \alpha \sqrt{1 - \left(\frac{2k}{N-1} - 1 \right)^2} \right)$ $J_0(\pi \alpha)$	2.39	-69.6	-6

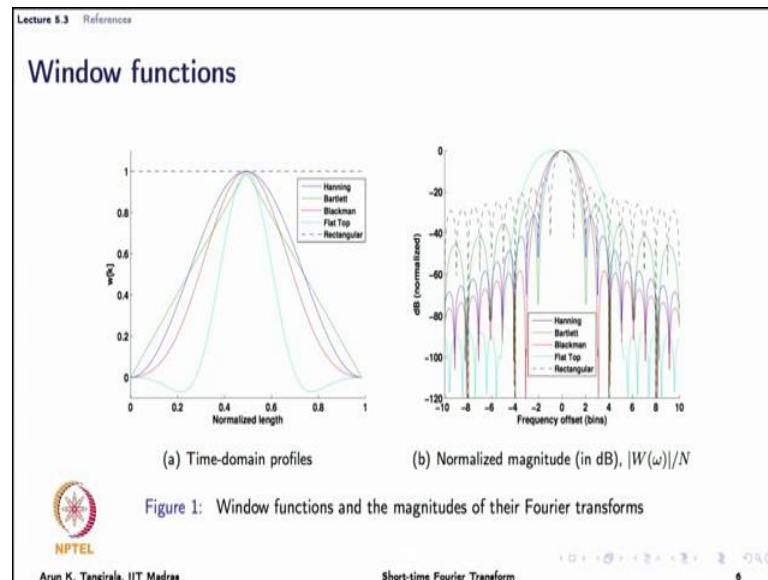
 Arun K. Tangirala, IIT Madras

Short-time Fourier Transform

Now, let us move on to the discussion on window functions. So, the theory of window functions is quite well studied right now; it is well established. We have talked about this windows briefly before. There are different types of windows that are available. We have discussed mostly with the help of rectangular windows. But, in the introduction to short time fourier transform I had shown you Gaussian type window functions as well.

And this Hanning and Hamming windows which are very popularly used, have the bell like shape of the, similar to the Gaussian windows. And there are a number of window functions. What is the difference essentially between these window functions?

(Refer Slide Time: 15:26)



Let us understand with the help of this slide here. I have the time profiles of the window functions shown here. Now, first of all you have to ask why I cannot live with rectangular window itself? Why do I need a different kind of window function? Now, I had briefly mentioned this in the context of spectral leakage when I talked about DFT. The rectangular window causes abrupt beginnings and ending of a segment.

This abrupt beginning and end of a segment introduces spurious frequencies which we say it introduces a lot of spectral leakage. Because, you are saying, the signal suddenly began and suddenly closed or suddenly concluded and that is what rectangular window implies. But, if I use a bell shaped or a nice smoothly tapered window function such as this, then you are not introducing abrupt changes in the beginning and the end of the segment, rather you being very kind. Therefore, you should expect the spectral leakage to come down.

In fact you should refer to the literature on spectral leakage, and you will see that the use of window functions is primarily there to mitigate the spectral leakage. If you recall, we have discussed a periodogram routine in MATLAB. There are choices of different windows that you can use, and the default window is rectangular, but you can change it to Hanning or Hamming and so on. And you should go through this as an exercise in

MATLAB, and you will be able to see that the use of rectangular window will cause most spectral leakage than the use of a Hamming window.

So, the major advantage of choosing this bell shape like window functions over the rectangular window, is that the spectral leakage is going to be mitigated. But, obviously, I have to pay a price somewhere. What is a price that I pay? The price that I pay can be understood by looking at the right hand side broad, where I am showing the magnitude of the Fourier transforms of the window function. Remember, I have here w of t on the left, I have w magnitude of w of ω on the right.

Now, this w of ω is very important because it tells me also what is the effective bandwidth? And if you recall, we said short time Fourier transform is also filtered with constant bandwidth, and the bandwidth is governed by the width of the, bandwidth of the w of the window itself, right. So, look at what we have here. The rectangular window represented by this black dashed line is shown here. It has a very narrow bandwidth; in the sense, it is very nice when it comes to resolving 2 different, closely spaced frequencies.

But then you can see that the rectangular window has very broad lobes, we call this as side lobes. These side lobes should ideally decay to 0 very quickly if I want very less spectral leakage. The broader, that is the longer these side lobes take to decay, the more the spectral leakage. So, it is as if, it is now bringing, the power is actually leaking to the neighboring frequencies as well.

So, compared to the other windows that I have, the rectangular window leaks the power around the frequency of interest, significantly to the neighboring frequencies. Look at the other frequencies, particularly the Hamming and Hanning, the side lobes role of very quickly. But, what is the price that I have paid? The main lobe is now wider, right. That we say, we say effectively, we have lose of resolution.

What is a resolution essentially? That is, how finely I can resolve to closely space frequencies. When I take a rectangular window and I have n observations in this window, and I perform DFT, this resolution is 1 over n in frequency. But, when I use a window function such as this, Hamming or Hanning, what I am doing is, I am giving less importance to the samples in the beginning and in the end; it is like as if I am doing a waiting here on this observations.

Effectively, I will not have, therefore, as many observations as I would have with a rectangular windowed segment. In other words, with rectangular windowed segments if I have n observations then I will have fewer effectively than n observations with the use of these windows; and that is what is reflected in this 3 columns here. So, all the properties that we have discussed are kind of summarized by this 3 properties. There are other properties of window functions, but these are the 3 key ones.

This 6 dB bandwidth is a measure of the width of the main lobe that we have seen here; smaller the value of the 6 decibel bandwidth, better is the resolving ability of the window. So, as you can see, rectangular window has the lowest 6 dB bandwidth value. Therefore, it has a maximum resolving ability. But, look at the peak side lobe value for the rectangular window, it is minus 13.3 because these are negatives.

What this means is, that the peak is much higher, peak of the side lobe is much higher for the rectangular window compared to other windows. As a result, you should expect more spectral leakage because the side lobe is as tall as the main lobe, you can say so. So, it is a measure of how low the peak side lobe is compared to the main lobe. And the final one is the side lobe roll off; how quickly this side lobes decay? Again, we have seen here the rectangular window actually decays very slowly or slowly, in fact compared to the Hanning window.

In fact, if you refer to Mallet's book, it also gives you the decay rate. The Hanning windows decay a order higher than the rectangular window. So, that is why this Hanning windows are preferred because spectral leakage is going to be minimal. So, what you are doing is you are mitigating spectral leakage at the cost of resolution; that is the tradeoff that is involved in window functions.

Now you have to ask yourself, what is ideal window function that I would like to have? The ideal window function that I would have is a peak, a simple single peak in the Fourier transform which is not possible. When is it possible? When I use a very wide window, then by the duration bandwidth principle it is going to have a very narrow bandwidth, right, and that will get me exactly the frequency content of x of ω .

But, in short time Fourier transform I cannot use a wide window because the purpose is defeated. I am going to use a window of finite width, but definitely not as wide as a signal, much narrowed than the signal. And therefore, the bandwidth is not going to be 0, but some finite value, and some kind of spectral leakage is bound to occur. And which

means that the actual features present in the signal are going to be distorted in your spectrogram; this is again a reiteration of what we have seen earlier.

(Refer Slide Time: 22:57)

Lecture 5.3 Reference


Effects of windowing

There are primarily two factors that affect the STFT:

- Window length:** This is important since by virtue of the D-B principle,

A narrow window in time produces very good energy localization in time, but results in a loss of localization in frequency domain, and vice versa.

Thus, a major challenge in STFT is to choose the "appropriate" window length for a particular application. Very often **it is required to choose analyzing windows of different lengths in order to extract different features.**
- Window type:** This factor mainly influences the spectral leakage or the distortion of $X(\omega)$ (the windowed part). Typically one chooses a Hanning or a Kaiser window.



NPTEL

Arun K. Tangirala, IIT Madras

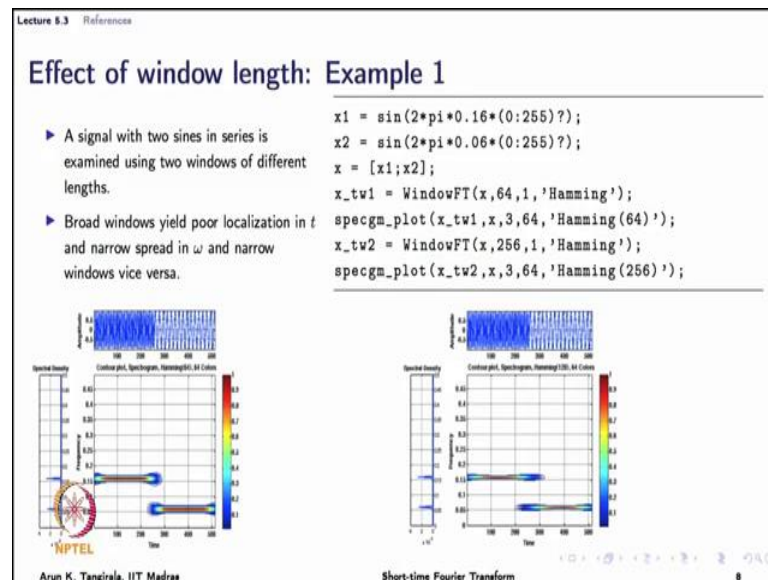
Short-time Fourier Transform

7

So, we will conclude the discussion with 2 main properties of the window length. We have already in fact discussed one of this, which is a window type. So, effectively, we have only 1 parameter left to discuss, which is the length of the window. And we have actually a fair good idea of what is window length does to the spectrogram. Narrower the window, better is its time resolving, time localizing ability of the energy, but then it loses out on the ability to resolve frequencies because the bandwidth is going to be much larger of the window.

And therefore, it will not be able to tell you exactly your pin point the frequency range and vice versa. Again, this is a consequence of the duration bandwidth principle. So, let us look at couple of examples here.

(Refer Slide Time: 23:43)



So, let us look at 2 examples that will throw light on the effect of window length. In the first example I have the signal that we have seen earlier on the introduction lecture where a high frequency sign wave is followed by a low frequency sign wave. The effect of window length is something that we expect; that we have already discussed. Wider windows will result in poor time localization, but get me better frequency localization. So, look at what is happening here.

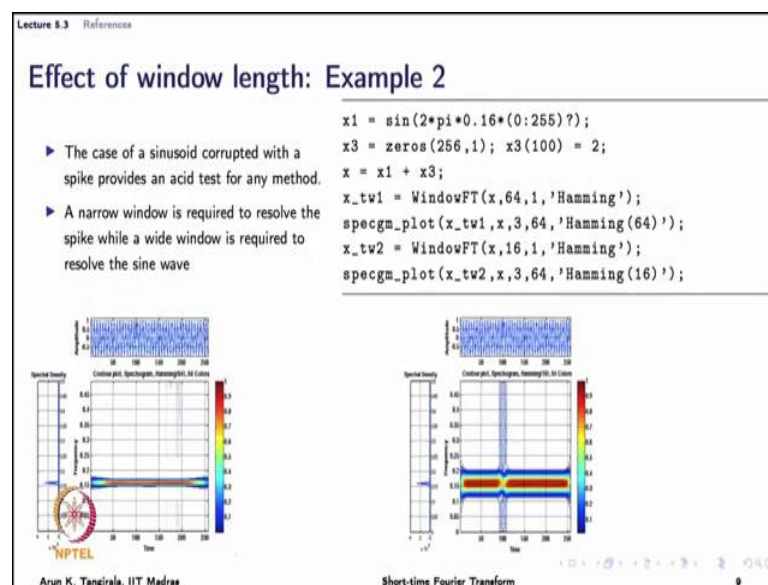
I am using the window ft to compute the short time Fourier transform. And then a routine spec gram underscore plot which I shall supply; it is a routine that I have written. Essentially, it takes a short time Fourier transform, return by the window ft, and computes the spectrogram, optionally plots the series and spectrum for you. Also note, that the window Fourier transform by default uses rectangular window, but I do not want to use rectangular window because it results in more spectral leakage. Therefore, I use Hamming window of width 64.

Now, when I use this width here compared to the width of 256 samples, that is window of 256 samples width, I have better time localization. What we mean by better time localization is, this spectrogram is able to pick the conclusion of the first frequency and the onset of the low frequency in a much better way than what a wider window does. But, the wider window is able to get me much better frequency localization than the narrower window.

In fact, if I said the window width equal to the length of the signal I will recover essentially the Fourier transform. You can just imagine now what would happen if I set the window length to 512 which is a length of the signal. In that case, these lines in spectrogram will run across the entire time, but they will be much finer than what you see; in fact, as fine as the spectral spectrum itself.

So, that is the extreme case; obviously, you do not want to do that; you might as well do a Fourier transform for that. Essentially, this illustrates what the window length does for you in terms of getting the time local and the frequency local features.

(Refer Slide Time: 26:10)



Another example will bring out this property in a better way or it reinforces the concept. Here, the signal is made up of a sign wave and an impulse, alright. And I want the spectrogram to tell me where the impulse is located; of course, because I am synthesizing I know it already; pretend that I do not know, once again I use the window ft to compute the short time Fourier transform, I use a Hamming window. But now, I use windows of 2 different widths – 64 and 16; there I use 64 and 256, the signal length is now 256.

So, what has happened? When I use a window of width 64, what you see here is actually a plot of the, again the spectrogram; it is able to detect the frequency, persistent frequency that is present, but very vaguely only it picks up somewhere this impulse as you can see in the contour here, but that is not convincing enough. In fact, if you apply a threshold, if there was noise it would be lost, right. These are all deterministic signals. Ideally you should workout in your exercises with noise also.

Now, come to the case of using the window of width 16. Here, the spectrogram is in a much better position to pick the impulse because I have a narrower window and therefore, it suits that impulse features nicely. But, it does not suite the sine wave feature nicely, in the sense that the energy of the sine wave in frequency is smeared quite a bit compared, in fact, by the duration bandwidth principle you should expect a 4 time more smearing because I have reduced the window width or duration by a factor of 4, right, or even probably higher depending on how things work out.

Qualitatively, you will see a large smearing and that is what exactly you see here. Again, the same story; if I choose a narrow window, then I will be able to pick features that are localized in time very well in the signal, but I will lose out on the frequency localization. So, this is a tradeoff. Now, what do I do? Do I choose a narrow window or a wide window? Unfortunately, you cannot just work with a single window length. In order to pick different features of the signal you will have to use different window widths.

And therefore, you will generate spectrograms for different window lengths, and you have to analyze all of them together. That is the major advantage with wavelets where yes you will use different windows of different widths, but you do not have to refer to scalograms of different, corresponding to different widths. All of them can be plotted in one single plot. Narrower windows will only get you the high frequency contents. They will not really extract or search for the low frequency content in signal.

And wider windows in wavelets will only search for low frequencies; they will not search for high frequencies. So, wavelets are ideally suited for these kinds of signals because I have a feature here which is highly localized in time, but spread all over in frequency. So, the narrow wavelet window will pick this. And I have a sine wave which is not localized in time, but highly localized in frequency; a wide wavelet window will pick this. So, that is why wavelets are very popular as you can see compared to the spectrograms. So, with this we come to a closure of this lecture.

Again, I have a few references for your reading. The most important thing is to work out MATLAB exercises so that you can understand, and at least, and also get the concepts reinforce by making mistakes, working out the examples, especially those that we have discussed in this lecture.

Thank you.