Signal Processing Algorithms & Architecture
Dr. Anirban Dasgupta
Department of Electronics & Electrical Engineering
Indian Institute of Technology Guwahati
Lec 16: Short-Time Fourier Transform (STFT)

Hello all, welcome to a fresh new lecture on the topic of short-term Fourier transform. This is Dr. Anirban Dasgupta and let us get started. So, we have analyzed the signal in the frequency domain using the discrete Fourier transform, and using the DFT, a lot of problems can be solved, even practical problems. But there are some issues with the discrete Fourier transform, and what are they? So let us study these one by one. So first are boundary effects or edge artifacts.

So, the limitation is that when a finite duration signal is used in the DFT, we assume that the signal is periodic. Why? Because we are getting line spectra, we know that line spectra will occur for periodic sequences. So, DFT assumes that your x[n] is a periodic extension; this is the discrete Fourier series of the periodic extension of your signal. So, say if this is my signal, so let us say I have 8 points.

Now what will happen is, let me use a different color and this will be repeated. So, this comes here. Similarly, here this tall value will come and it is assumed that this goes up to infinity in both directions. So, this sudden change these are called the boundary effects and this will introduce additional frequency which are inherently not present.

This will probably be clear if I say this sample had a smaller value, then you can acknowledge the boundary effect nicely; say this value is small. So now you see this sudden change is creating an artificial high frequency that is not at all present in the signal, or if this value is negative. The same thing will happen if I do change this value because it is a periodic extension. So, if this value is very small like this, this is an artificially induced high-frequency component. So, this is taken care of by transforms like the discrete cosine transform and the discrete sine transform.

So, what will the discrete cosine transform do? So, if this is your signal, it will create an even extension of this signal and then assume that the even extension is periodic. So, if this is my signal and if this is my index 0, if I do an even extension, again let me use another color. If I do an even extension, you'll see that this is quite smooth and even; since this is an even extension, I will get this repeated or this very smooth. So, this is a smoother version; this is what DCT does. So, DST will do an odd extension.

Now, you can keep in mind that we studied odd and even signals. So, this is one application to understand the properties of the discrete cosine transform and the discrete sine transform. Another is real and complex signals. So, even though we have a really

nice-looking practical signal, if we do the DFT, it will produce a lot of complex things. Now I agree that this is to encapsulate the phase of things, but this is not a real-valued transform, and it is difficult not only to store it but also to analyze it, as it becomes very complex with imaginary values.

So, this is again solved by DCTs and DSTs which are real valued transforms and this is more efficient for real valued signals. And in this manner, it will improve computational efficiency. How? Now, if I have two complex numbers, then I need to do the multiplication of the reals, the multiplication of the imaginaries, and then the cross multiplication of the real and imaginary parts. But if everything is real, the number of multiplications and additions will be reduced. Also, this is efficient to store because if I have 8-bit memory and it is a real 8-bit signal, then one memory unit will be sufficient.

But if it is a complex signal, then I need a paired storage unit for storing the real and complex values. Then signal compression and reconstruction. So, we also see that the DFT is not very good at compression and reconstruction because there will be a large gap in the frequency. What do I mean? So, compression means that if I have a sine wave of a single frequency, it is easy to store. If I just know the frequency, I will store that.

But in the real world, my signal will be composed of a lot of signs, and probably you will have to store all the components, which are not memory efficient in compression. But DCT coefficients will be compact, so you do not need to store all the n values; you may only need to store those values which are high enough and truncate low-order DCT values. And this is commonly used in our image compression, which is JPEG. Of course, this is one of the most popular image formats that all of you have been using: the JPEG or JPG image format, and the MP3 audio files are so popular. So, this MP3 and this JPEG use compression that is based on the discrete cosine transform.

Another problem is the lack of time localization. So, as I showed in the introduction to time-frequency analysis, suppose I have a slowly varying wave of, say, 10 hertz, and suddenly I have a fast wave, and then it goes slow. So, I know that both slow frequencies and high frequencies are present. So, this is 10, this is 30, and this comprises the 30. So, I know which components are present in the frequency domain, but I do not know where they are present.

Also, say I am singing, so while I am singing, both vocals and instruments are present. But then there will be some prelude or interlude in my song where there are only instruments. So, I know what frequencies are present, but where they are present in time is difficult to localize by just looking at the spectra. So, this is solved by something called the short-term Fourier transform, which is the topic of discussion in this lecture. So, what the short-term Fourier transform does is instead of looking at the whole spectra, we will be interested in looking at small-sized windows and evaluating the spectra of that

window.
Then handling non-stationary signals. So, what is non-stationarity? The signal properties will vary. Like I gave the example of the chart signal, where the frequency is increasing linearly with time. So, real-world signals are often non-stationary in nature. So, just looking at the holistic spectrum will not give a good analysis.

So, the FFT or the DFT will give a snapshot of the entire signal's frequency content. So, we should be looking at a small overlapping window, which is solved by our short-term Fourier transform. And then we can select different types of windows to control something that is called frequency leakage or spectral leakage. So, this is what the short-term Fourier transform does. So here we have this signal, and now if I do the DFT of the whole signal, say this is of length 100, I will get the frequency content of the total signal, but I do not want that.

There are some events like this and this where there are a lot of variations I need to capture. So, what I will do is let us select a window that is size 20. So typically, I will have five windows that are non-overlapping. Of course, we can also use overlapping windows. So, if I select this first window, which is when the signal is x[n], then I would say this sub-window or sub-signal is x[0] to x[19], and I will compute the DFT and have a spectrum for this.

So, this is a vector; the DFT magnitude spectra will be another vector. Next, I will move this window from index 20 to index 39, and then I will do the same for index 40 to index 59, and similarly, we will continue to other non-overlapping windows. So, this is the mathematical representation for the STFT of a signal x[n] for a window that is given by-

$$X(t,\omega) = \sum_{n=-\infty}^{\infty} x[n] \cdot w[n-t] \cdot e^{-j\omega n}$$

where t is the translation of this window, and this is the index that represents the position of the window in the time domain. In this window, you can select anything.

So, the window, if you just take the samples directly, is called a rectangular window. But the problem with a rectangular window is that there will be sudden changes in the signal values from a finite value to a sudden zero value. A rectangular window is easier to use, but we understand that it will cause a lot of spectral leakage. So, there are other windows like the Hamming window, the Henning window, the Bartlett window, and the Gaussian window; there are a lot of windows. So, the window function in this case is typically moved across, and in the short-term Fourier transform, this length is fixed.

The frequency resolution depends on the length of the window, and that is true for any DFT algorithm. So, when we do an endpoint DFT, if my n is large, my frequency resolution will be high, and vice versa. So, if we have a larger window, we will have better frequency resolution. As I said, if I see a long part of the signal in the time domain, I can better understand what frequency content is present. But if I reduce the window size, say I have just two samples, what frequency information will I know about the signal with the two samples? Kind of nothing.

$$X[k,m] = \sum_{n=0}^{N-1} x[n] \cdot w[n-m] \cdot e^{-j\frac{2\pi kn}{N}}$$

So, the more I am localized in time, the more uncertain I am about what frequencies are present. But more I am localized in frequency, as exactly 50 hertz is present, which requires me to have an infinite length window in time. So, this trade-off has to be managed depending on the problems. If you are localizing an event, then we need to know what the typical duration of that event should be. Like in a seismic signal, we are localizing some events, such as an earthquake or some vibration of the ground.

So, in that case, we should know what the existence of that high-frequency component in the signal is, so that I can select my window length in the STFT. And the resultant plot is called the TFR or time-frequency representation, where the x-axis represents time and the y-axis represents frequency. So, this is the mathematical definition, where I have selected the window as a discrete index m. Now let us take an example of computing the STFT.

$$x[n] = sin\,(2\pi f_0\, n) + sin\,(2\pi f_1\, n)$$

$f_0 = 5\ Hz$ and $f_1 = 10\ Hz$, and sample at $f_s = 100\ Hz$

So here I have the sum of two sinusoids of frequencies f_0 and f_1, and for simplicity, let us say I take $f_0$as 5 hertz and f_1 as 10 hertz, and we will use a sampling rate of 100.

So, this will be $2\pi f_0$; so, if this is already in n, there is no problem, but if this was in t, you have to do $\frac{n}{f_s}$. So, the first thing is you have to select your n, and then I will divide the signal into segments. I can select overlapping segments; for example, I can take a 25 percent overlap or a 50 percent overlap. This also needs to be taken care of because the more overlap you take, the more computations you have to do, but you get better localization in time and vice versa. Then I selected some segments, so I have to do the Fourier transform for each segment.

That is, first I will take one window, do the Fourier transform, then slide the window, then do the Fourier transform again, slide the window, and do the Fourier transform. And typically, these sub-signals, overlapping segments, or even non-overlapping segments can

be computed in parallel because these transforms are independent of each other. So, if you have the computation facilities, then you can do them in parallel. So, in that manner, you can also do faster processing. You need not wait for the completion of the previous stage and then finally, we will plot this in something called a spectrogram. So, here we can clearly see that this yellow color represents the highest magnitude. And we can see that in this region 10 and in this region 5 we have very high values of magnitude, which indicates that yes, my 5 Hertz and 10 Hertz were my original frequency content. And the remaining values are low, which appear because of factors like windowing and spectral leakage. But overall, we are able to get the picture that these two frequencies are present.

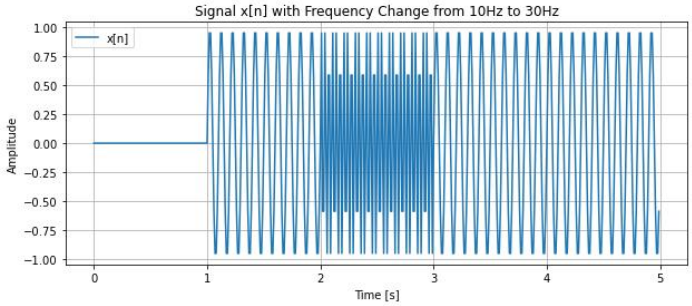
1. From $t = 1$ to $t = 5$, the signal will be a 10 Hz sine wave.
2. From $t = 2$ to $t = 3$, the signal will be a 30 Hz sine wave.
3. Outside the range of 2 to 3 seconds, it will be 10 Hz again.

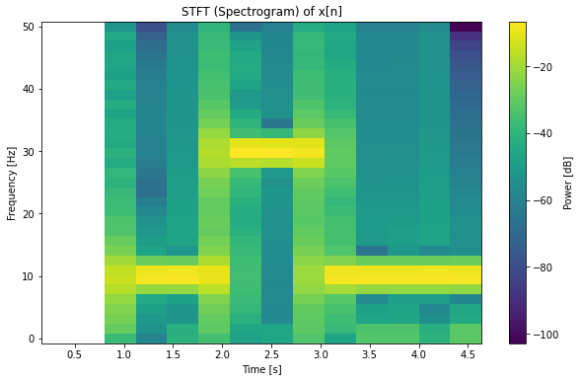
$$x(t) = \begin{cases} \sin(2\pi \cdot 10t),\ 1 \le t < 2 \\ \sin(2\pi \cdot 30t),\ 2 \le t < 3 \\ \sin(2\pi \cdot 10t),\ 3 \le t < 5 \end{cases}$$

So, this is doing what my normal DFT does and not only that it is telling me that these components are present in the entirety of the signal. What about increasing the complexity of the problem? Let us say I have a signal from 1 to 5 seconds of a sine wave of 10 hertz, but from 2 to 3 seconds, the signal will display an oscillation of 30 hertz, and outside this, it is 10 hertz again, like slow and then suddenly fast and slow, which indicates that some event has occurred here. Now, if we do the DFT as I explained, you get both the frequency content, but you do not know where this 30 hertz appeared. So, this is the piecewise representation of the signal, and this is the time-domain representation of the signal.

So, clearly we see that from 2 to 3 hertz, these first oscillations have happened. But although this is visible over time, it is not visible in the frequency. We just know that 30 hertz and 10 hertz are present. But at the same time, we know that there is something happening here, but we clearly do not know if this is 30 or what frequency this is. So, here comes STFT to our rescue, and now we clearly see that since this yellow is representing the highest magnitude, we see that in this region, say from 1 second to 2 seconds, my 10 hertz is the dominant frequency, and then from 2 to 3 seconds, I can see that 30 is the dominant frequency, which indicates that this dominant frequency has occurred in this region from 2 to 3 seconds.

And then afterwards, again from 3 onwards, we get 10 hertz as the dominant. Now I request you to try this as an assignment: keep 10 throughout and also add 30 in this region, and you will see that this yellow line is throughout, whereas this 30 Hertz





component, represented by this yellow line, is only for 2 to 3 seconds. So I will end this with some properties of the STFT.

$$STFT\{a \cdot x[n] + b \cdot y[n]\} = a \cdot STFT\{x[n]\} + b \cdot STFT\{y[n]\}$$

First is linearity, and just like all the previous transforms, this STFT is also linear, which means that if I do a linear combination of two signals xn and yn with constants a and b, I get the linear combination of the transforms in a similar way or with similar weights. So, this is useful when we are given the STFTs of smaller signals, and we can calculate the STFT for a larger complex signal that is a combination of those smaller signals.

Another property is shift invariance.

$$STFT\{x[n-k]\} = e^{-j2\pi fk} \cdot STFT\{x[n]\}$$

So, if I shift this signal in the time domain by a factor k, then only the phase will change, and the transform of the magnitude spectrum, or the STFT in general, will be preserved. So, thank you so much and have a nice day.