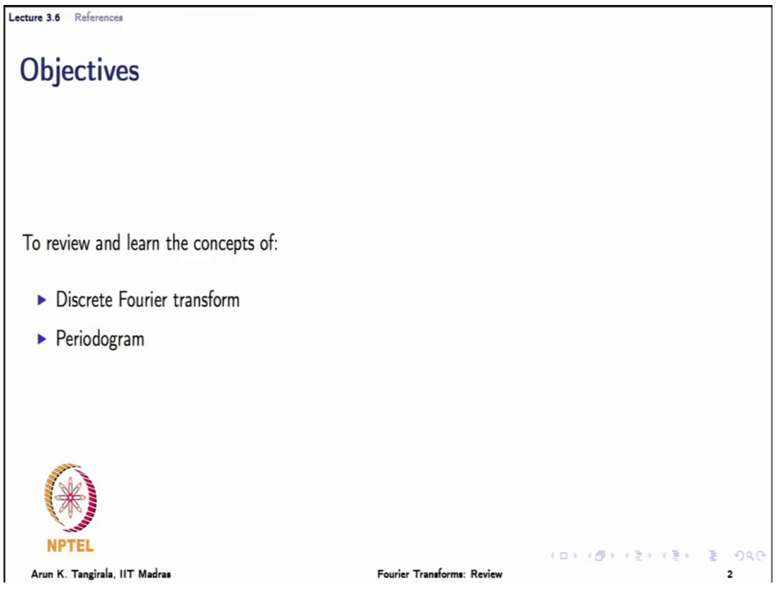
Introduction to Time-Frequency Analysis and Wavelet Transforms

Prof. Arun K. Tangirala
Department of Chemical Engineering
Indian Institute of Technology, Madras

Lecture – 3:6
Discrete Fourier Transform and Periodogram

Hello friends, welcome to lecture 3.6, which is the final lecture of the module in the third unit, on the review of Fourier transforms. This particular module is going to be presenting review of the Discrete Fourier Transforms and also the concept of Periodogram towards the end. So, this is a very practically relevant transform. Because, this is the transform that, we apply to sample data. Until now, we have studied all the theoretical transforms. But, now we are going to ask certain questions that are relevant to us in practice.

(Refer Slide Time: 00:59)



Lecture 3.6 References

Objectives

To review and learn the concepts of:

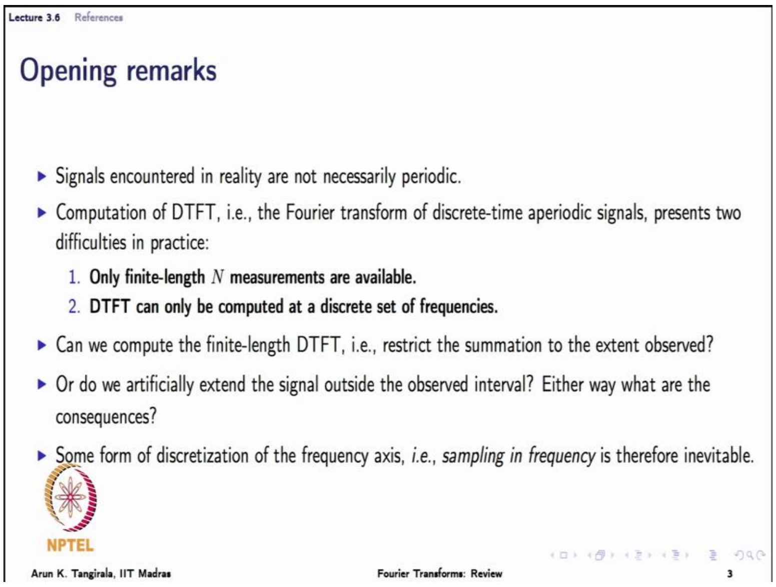
- Discrete Fourier transform
- Periodogram

NPTEL

Arun K. Tangirala, IIT Madras — Fourier Transforms: Review — 2

So, the objectives of this module, is to review and learn the concepts of discrete Fourier transform and conclude that with the Periodogram.

(Refer Slide Time: 01:10)


Lecture 3.6 References

Opening remarks

- Signals encountered in reality are not necessarily periodic.
- Computation of DTFT, i.e., the Fourier transform of discrete-time aperiodic signals, presents two difficulties in practice:
  1. **Only finite-length N measurements are available.**
  2. **DTFT can only be computed at a discrete set of frequencies.**
- Can we compute the finite-length DTFT, i.e., restrict the summation to the extent observed?
- Or do we artificially extend the signal outside the observed interval? Either way what are the consequences?
- Some form of discretization of the frequency axis, *i.e.*, *sampling in frequency* is therefore inevitable.

NPTEL

Arun K. Tangirala, IIT Madras Fourier Transforms: Review 3

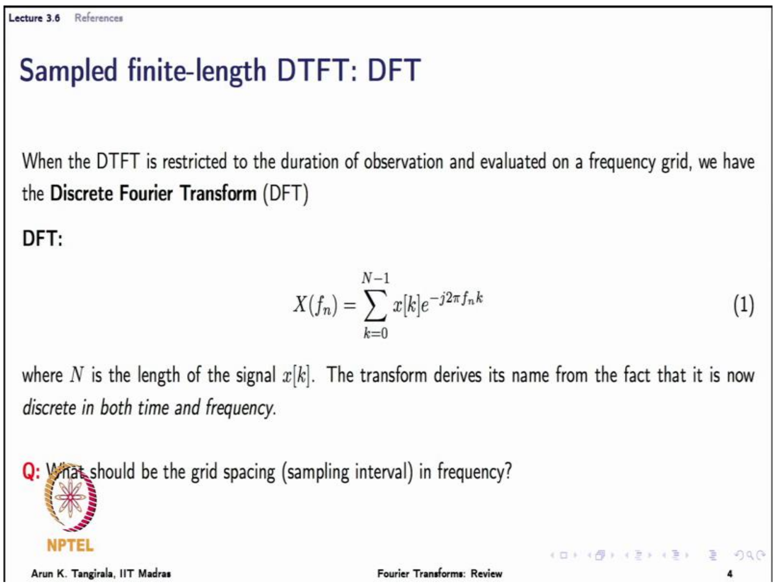
Now, the motivation for looking at discrete Fourier transform, as I have explained to you, is to consider the practical scenario where, first of all, the signals encountered in reality are not necessarily periodic. More importantly, the signals that I have for analysis are finite line. I cannot conduct an experiment for infinitely in time. So, how do I deal with the finite line data?

Because, if you look at all the four theoretical variance of transform or even the discrete time Fourier transforms, that we have learnt. They assume that, the signal is either periodic, which may not be the case in practice. Or if the signal is aperiodic, it assumes that it exist forever or that, you have the knowledge of the signal over infinite time. Neither of this is true, so now how are we going to deal with this?

And there is a second issue with the use of discrete time Fourier transform. Even if I am given, let us say the signal for a very long period of time as far as computing the DTFT in practice is concerned, I can only compute it, over a grid of frequencies or at a discrete set of frequencies. It is obviously not possible to compute the DTFT over a continual. So, can I compute the finite length discrete time Fourier transform? That is, restrict the summation to the extent that I have. Can I do that? If I do that, what is the consequence or I would still like to use the discrete time Fourier transform as is but, by first constructing artificially, the infinitely long signal from the finite line data. Is that another possibility? And if I do that, what kind of extensions are possible, outside the observation interval. And what does a particular extension of the signal lead to and so on.

So, the message is I need to handle finite line data and I can only compute DTFT at discrete set of points. So, some form of discretization of the frequency access. In other words, sampling in frequency is therefore, inevitable.

(Refer Slide Time: 03:32)



Lecture 3.6 References

Sampled finite-length DTFT: DFT

When the DTFT is restricted to the duration of observation and evaluated on a frequency grid, we have the **Discrete Fourier Transform** (DFT)

DFT:

$$X(f_n) = \sum_{k=0}^{N-1} x[k]e^{-j2\pi f_n k} \quad (1)$$

where N is the length of the signal $x[k]$. The transform derives its name from the fact that it is now *discrete in both time and frequency.*

Q: What should be the grid spacing (sampling interval) in frequency?

NPTEL

Arun K. Tangirala, IIT Madras Fourier Transforms: Review 4

With these remarks, we introduce the discrete Fourier transform. And we ask, what is the consequence of computing this discrete Fourier transform with respect to the theoretical discrete time Fourier transform. So, when I restrict the DTFT’s. So, I will say well. I will ignore, what has happened outside the interval of my observation. And I am going to only compute the transform, for the length of data that is available to me.

And as I mentioned earlier, I am go to compute it at specific frequencies. So, now this subscript n returns to the forum. Earlier, in the discrete time Fourier transforms, the frequency axis is continuous. Now, because I am going to compute it, over a grid like in any other software, like numerical integration and so on. Now, I have f subscript n this n keeps track of the grid point, at which I am evaluating the Fourier transform.

So, this is not much different from the discrete time Fourier transform. It is taking birth from DTFT, by restricting it to the interval of observation and evaluating it, on the grid. So, the expression is more or less the same. I replace f with f subscript n. And I restrict the limits of summation to lower limit to 0 and upper limit to n minus 1. That is a convention that, we follow you have to be careful.

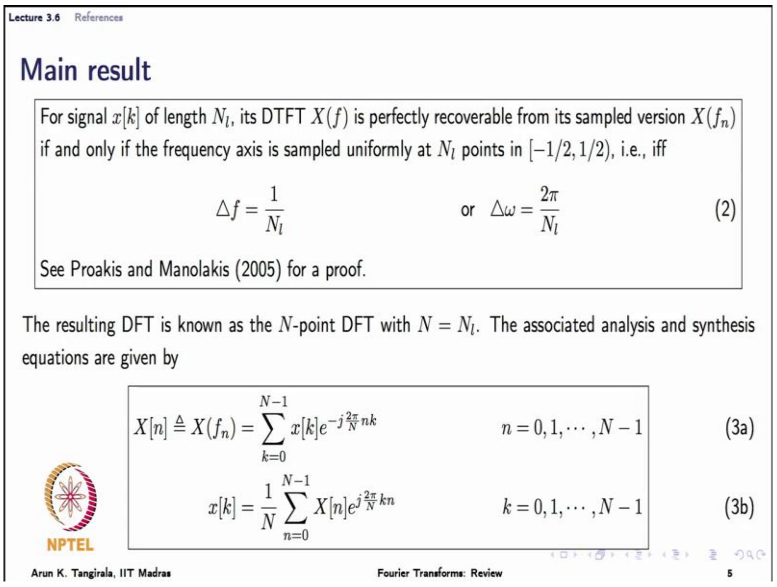
In software packages, starting index is N equals 1 so, we have to adjust the code for the computation of DFT accordingly. Now, here N is the length of the signal. The reason for calling this as discrete Fourier transform is, because now you are not only dealing with sample data that is in time. But, also you have sampled the frequency. So, it is no longer, just discrete time Fourier transform. It is a discrete time discrete frequency Fourier transform that is a mouthful.

So, therefore we simply call it as discrete Fourier transform. Understanding that, it is discrete in both time and frequency. That is a reason for this name. Now, the big question that we have is, what should be the grid spacing? That is, how fine should be the grid, should the grid be uniform, non-uniform. If it is uniform, then what is the spacing that I should choose? What is the consideration and so on?

Now, interestingly there is a parallel question. There is a similar question that arises, in sampling continuous time signals, which we have discussed earlier, in the context of sampling and sampling theorem and so on. There we ask a question, how fast should I sample, whether, I should sample uniformly? And if I sample uniformly, what should be this spacing between two sampling instances? Exactly, although it is now we are dealing with frequency, the questions are very similar.

There sampling in time, our consideration was that I should not lose information. What I mean by this is, if necessary, I should able to reconstruct the continuous time signal from its sample version. We can place the similar requirement here. And we can say that, I would like to choose the grid spacing. Assume that, we are going to deal with uniform grid. I am going to choose a grid spacing such that, I will be able to recover the continuous function. The continuous Fourier transform which is X of f that, I would computed otherwise, from the sample version of X of f. That is, I would like to recover X of f from X of f subscript n.

(Refer Slide Time: 07:09)



Lecture 3.6 References

Main result

For signal $x[k]$ of length N_l, its DTFT $X(f)$ is perfectly recoverable from its sampled version $X(f_n)$ if and only if the frequency axis is sampled uniformly at N_l points in $[-1/2, 1/2)$, i.e., iff

$$\Delta f = \frac{1}{N_l} \qquad \text{or} \qquad \Delta\omega = \frac{2\pi}{N_l} \tag{2}$$

See Proakis and Manolakis (2005) for a proof.

The resulting DFT is known as the N-point DFT with $N = N_l$. The associated analysis and synthesis equations are given by

$$X[n] \triangleq X(f_n) = \sum_{k=0}^{N-1} x[k]e^{-j\frac{2\pi}{N}nk} \qquad n = 0, 1, \cdots, N-1 \tag{3a}$$

$$x[k] = \frac{1}{N}\sum_{n=0}^{N-1} X[n]e^{j\frac{2\pi}{N}kn} \qquad k = 0, 1, \cdots, N-1 \tag{3b}$$

NPTEL

Arun K. Tangirala, IIT Madras Fourier Transforms: Review 5

And it turns out that, you can do this if and only if, you choose the interval in frequency as 1 over N subscript l. N subscript l is now the length of the signal that you have. So, if I have a signal of length N subscript l, then X of f that is the continuous function is perfectly recoverable from its sample version, if and only if, the frequency axis is sample uniformly with this spacing. Of course, we are only going to sample in the fundamental frequency range.

That situation does not change, because we are still dealing with these discrete time signals. So, in terms of the cyclic frequency, this spacing grids spacing is 1 over N. In terms of the angular spacing, it is angular frequency spacing is 2 pi over N subscript l. Now, you can see a proof. You can refer to this book by Proakis and Manolakis for proof. But, there are other books as well, that will give you proof of these statement.

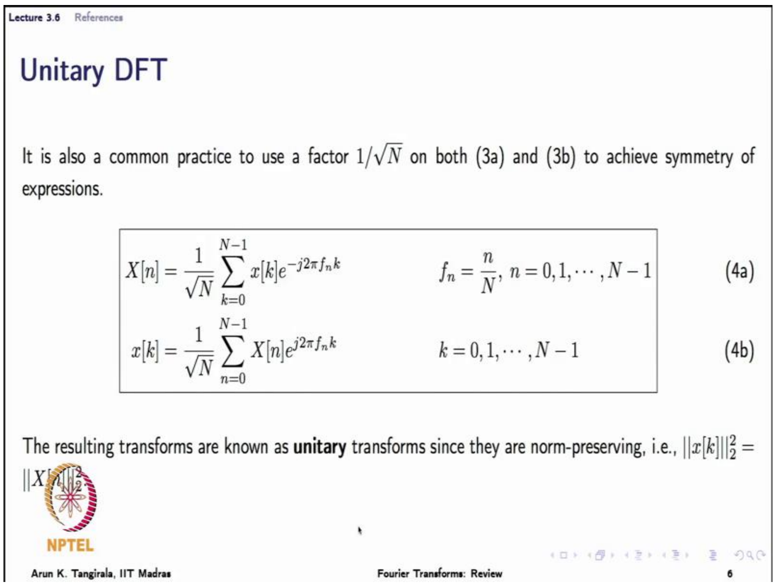
That is, it tell proof to you, that you can recover X of f from X of f n. Now, with this result having accepted these result. Now, we can talk about the DFT, a bit more in detail. And also study the consequences of restricting the summation to the observation interval. Now, formally we say X of f n is X of n. The way, the same notation that, we used for discrete time signals. In discrete time, in time we use k to keep track of the sampling instant.

In frequency, we use n to keep track of the frequency point. So, X of n is nothing, but what you are seen earlier. And we evaluate N from ze. That is, we evaluate the DFT for all values of n, running from 0 to N minus 1. And given at the bottom, in 3b you have the synthesis equation. So, this is a reversal of what we have done, until now.

We here presented the analysis equation first and then the synthesis equation. Now, we are presenting the synthesis equation first. Because, now we are really concern with the practical situation. The synthesis equation allows me to recover the signal, whenever I want. Notice again, the difference of 1 over N as a factor, in front of the summation. In the synthesis equation, I have a 1 over N. Whereas in the analysis equation, I do not.

But, the forms of the summation look the same. Expect that in the analysis, I am summing over time. And in synthesis, I am summing over frequencies. Now, again the similarity of these, allows me to write, simplifies the coding problem. I just need to write one algorithm and change the sign here.

(Refer Slide Time: 10:00)



Lecture 3.6 References

Unitary DFT

It is also a common practice to use a factor $1/\sqrt{N}$ on both (3a) and (3b) to achieve symmetry of expressions.

$$X[n] = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} x[k] e^{-j2\pi f_n k} \qquad f_n = \frac{n}{N},\ n = 0, 1, \cdots, N-1 \qquad (4a)$$

$$x[k] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} X[n] e^{j2\pi f_n k} \qquad k = 0, 1, \cdots, N-1 \qquad (4b)$$

The resulting transforms are known as **unitary** transforms since they are norm-preserving, i.e., $||x[k]||_2^2 = ||X[n]||_2^2$

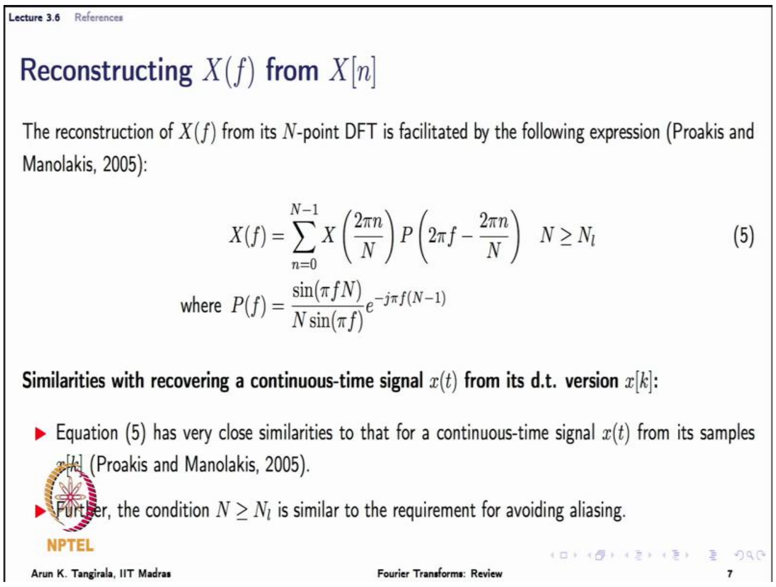
NPTEL

Arun K. Tangirala, IIT Madras Fourier Transforms: Review 6

Now, in fact it is further simplified. If you consider, what is known as a unitary DFT, where I deliberately introduces factor of 1 over root N in my analysis equation. So, that the synthesis equation also has 1 over root N. This is called unitary DFT. And the reason is that, it preserves the square 2 norm of the signals. In other words, if I take the square 2 norm of the signal that I observed. That is the same as the square 2 norm of the DFT that, I have computed.

And that is why it is called unitary transform and so on. Notice again, this is very important until, it becomes a practice. You should keep telling yourself that, you are computing a frequencies n over big N, where n runs from 0 to big N minus 1. And big N is a length of the signal. Of course, we have dropped a subscript l here. This is called an n point DFT, because I am evaluating the DFT at n points on the frequency axis.

(Refer Slide Time: 11:01)



Lecture 3.6 References

Reconstructing $X(f)$ from $X[n]$

The reconstruction of $X(f)$ from its N-point DFT is facilitated by the following expression (Proakis and Manolakis, 2005):

$$X(f) = \sum_{n=0}^{N-1} X\left(\frac{2\pi n}{N}\right) P\left(2\pi f - \frac{2\pi n}{N}\right) \quad N \geq N_l \tag{5}$$

$$\text{where } P(f) = \frac{\sin(\pi f N)}{N\sin(\pi f)} e^{-j\pi f(N-1)}$$

Similarities with recovering a continuous-time signal $x(t)$ from its d.t. version $x[k]$:

- Equation (5) has very close similarities to that for a continuous-time signal $x(t)$ from its samples $x[k]$ (Proakis and Manolakis, 2005).
- Further, the condition $N \geq N_l$ is similar to the requirement for avoiding aliasing.

Arun K. Tangirala, IIT Madras Fourier Transforms: Review 7

So, now just to obtain an insight of, how DFT is working or if I want to know, how x of f can be recovered from its sample version. That is a continuous Fourier transform. Not continuous time Fourier transform. The continuous function Fourier transform from its sample version. How do I do it? Well, this is the expression that allows you to recover x of f, from its sample version. I am avoiding the derivation here.

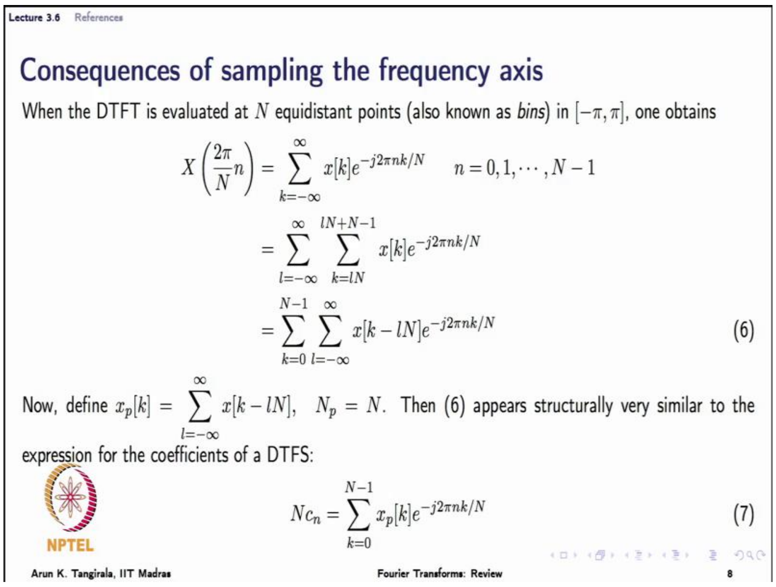
But, you can refer to any standard signal processing in text like such as Proakis and Manolakis to see, how this recover expression recovers the x of f for you. Now, it turns out that, there is a strong similarity of this expression, with that of recovering x of t from its sample data, which we have not seen in this course. But, again any standard signal processing text will give you, the expression for recovering x of t, the continuous time signal from its sample version.

So, the reason for pointing out the similarities is again, there is a lot of similarities and dualities between time and frequency domains. So, now the other condition that is required. One more point of similarities that is, the number of points at you choose on the frequency axis or frequency grid, should be greater than or equal to the number of observations, that you have. At that means, you should at least have as many points on your frequency axis.

You assuming, uniform spacing as the number of observations that you have, which means I am allowed to evaluate at finer points. But, then what is the consequence of doing that? We will discuss shortly. First, let us understand what is the consequence of sampling the frequency axis? Early on in this lecture, we said, is there a possibility that I can construct and artificial infinitely long version, corresponding to the finite length signal that I have observed.

By some extension, what we mean by extension is. For example you could assume, x of k, the original infinitely long signal to be let say 0, outside the interval of observation. So, I can say that, that is one possibility. Or the other possibility is that, I assume a periodic extension of the finite line signal and so. So, there are several possibilities. I can hold, I can assume that the signal was constant before and after I observed and so on.

(Refer Slide Time: 13:41)



Lecture 3.6 References

Consequences of sampling the frequency axis

When the DTFT is evaluated at N equidistant points (also known as *bins*) in $[-\pi, \pi]$, one obtains

$$\begin{aligned} X\left(\frac{2\pi}{N}n\right) &= \sum_{k=-\infty}^{\infty} x[k]e^{-j2\pi nk/N} \qquad n = 0, 1, \cdots, N-1 \\ &= \sum_{l=-\infty}^{\infty} \sum_{k=lN}^{lN+N-1} x[k]e^{-j2\pi nk/N} \\ &= \sum_{k=0}^{N-1} \sum_{l=-\infty}^{\infty} x[k-lN]e^{-j2\pi nk/N} \qquad (6) \end{aligned}$$

Now, define $x_p[k] = \sum_{l=-\infty}^{\infty} x[k-lN]$, $N_p = N$. Then (6) appears structurally very similar to the expression for the coefficients of a DTFS:

$$Nc_n = \sum_{k=0}^{N-1} x_p[k]e^{-j2\pi nk/N} \qquad (7)$$

Arun K. Tangirala, IIT Madras Fourier Transforms: Review 8

Now, we are asking when I compute the DFT and when I am sampling the frequency axis. So, in DFT I am doing two things. I am restricting the summation limits to the observation interval. And I am also sampling the frequency axis. So, what is the consequence of sampling the frequency axis? That is, what we are studying here. So, I am just showing you the derivation here. I am not really going to the details, but as a simple excise for you.

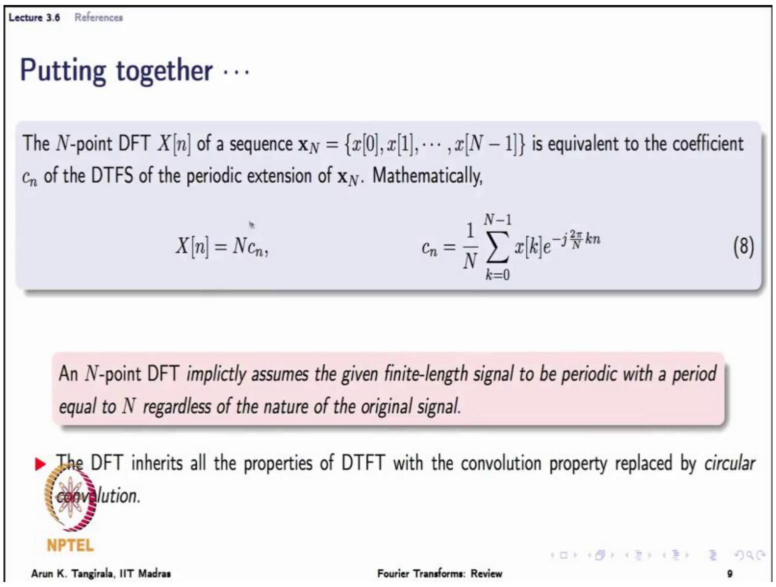
Go through these derivations here. These are again found in standard text. So, we start with the expression for the DFT coefficient, discrete time Fourier transform. At that point n, then ask what this means? Then, you can re express this summation. You can rewrite the summation as a double summation. We have deliberately introduced that. So, double summation. And one more change of variable will brings, this third expression for us, in equation 6.

Now, we introduce a new signal x subscript p, which is this finitely summed up x k minus l n. And there is a reason for doing that. Here, you can see that, this particular signal x p subscript k is periodic. It is actually periodic with period N. That means, you can show clearly that x p of k plus N is nothing, but x p of k itself. And that is why, we uses a subscript p. To indicate that this infinite sum is nothing, but a periodic signal.

Plugging in, this new variable into equation 6, gives me equation 7. Well, not exactly. But, the right hand side of equation 7, where I have sigma k equals zero N minus 1 x p of k e to the j minus 2 pi k over N. Why are we paying attention to this expression? That is because; this has a very strong similarity to what the expression that I had in the discrete time Fourier series. When I am looking a periodic signal, remember I would use discrete time Fourier series.

When I am computing the coefficient of discrete time Fourier series, I use C n. So, what I am showing you here for your comparison. Replace this infinite summation, inner summation in equation 6 with x p k. And then, compare with the expression that we have for N times C n, that you had for in a discrete time Fourier series expansion of x p. Then, you will see, that DFT has very strong similarity with discrete time Fourier series. The only difference is, I am computing N C n instead of C subscript n.

(Refer Slide Time: 16:41)



Lecture 3.6 References

Putting together ⋯

The N-point DFT $X[n]$ of a sequence $\mathbf{x}_N = \{x[0], x[1], \cdots, x[N-1]\}$ is equivalent to the coefficient c_n of the DTFS of the periodic extension of $\mathbf{x}_N$. Mathematically,

$$X[n] = Nc_n, \qquad c_n = \frac{1}{N}\sum_{k=0}^{N-1} x[k]e^{-j\frac{2\pi}{N}kn} \tag{8}$$

An N-point DFT *implictly assumes the given finite-length signal to be periodic with a period equal to N regardless of the nature of the original signal.*

- The DFT inherits all the properties of DTFT with the convolution property replaced by *circular convolution.*

Arun K. Tangirala, IIT Madras Fourier Transforms: Review 9

In the other words, the discrete Fourier transform coefficient is nothing, but N C n itself. So, that is the point. It is X of n is N C n. What is the consequence of, what is the point in actually comparing these two? The main point is, in restricting the summation to the observation interval and sampling the frequency uniformly with a frequencies spacing 1 over N, amounts to assuming that, the infinitely long signal which have not observe is periodic with the period equaling the duration of the observation.

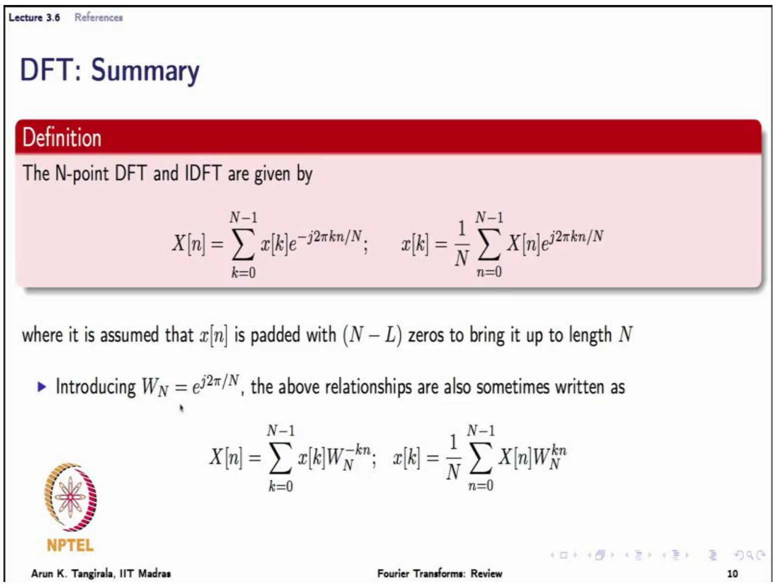
So, that is regardless of the nature of the underline signal. Which means, implicitly when you compute DFT, you are assuming that the underline infinitely long signal is periodic. That is a major assumption that you are making. And many of us probably, even do not take note of this, when I compute the Fourier transform in practice, in all the software packages. When I am using f of t, which is an algorithm to compute DFT.

Implicitly, I am assuming that the infinitely long signal that I have not observed is actually periodic, with the period equaling the length of the observation. So, that is a very important assumption. Remark and observation that, we will use later on. So, X of n is nothing, but N C n. What is this C n? C n is the coefficient of the discrete time Fourier series for the signal that you observe. So, I take the signal that I have observed.

I compute DFT. That is, X of n. My friend takes the same signal and assumes that, it is a periodic signal. And that, he or she has observed, the once full period of this periodic signal and computes the Fourier series coefficients. When I take the Fourier series questions and multiplied with N, it will exactly match the DFT question that, I have computed. That is a point.

So, this is something that should be kept in mind and should be reinforced in our minds. Until, we are very clear about it and know it, for sure. So, moving on just as a passing remark, the DFT inherits all the properties of the discrete time Fourier transform with the only difference that we replace convolutions with circular convolutions. Because, remember DFT involves finite summation, whereas the discrete time Fourier transform involves infinite summation.

(Refer Slide Time: 19:14)



Lecture 3.6 References

DFT: Summary

Definition

The N-point DFT and IDFT are given by

$$X[n] = \sum_{k=0}^{N-1} x[k]e^{-j2\pi kn/N}; \qquad x[k] = \frac{1}{N}\sum_{n=0}^{N-1} X[n]e^{j2\pi kn/N}$$

where it is assumed that $x[n]$ is padded with $(N-L)$ zeros to bring it up to length N

- Introducing $W_N = e^{j2\pi/N}$, the above relationships are also sometimes written as

$$X[n] = \sum_{k=0}^{N-1} x[k]W_N^{-kn}; \quad x[k] = \frac{1}{N}\sum_{n=0}^{N-1} X[n]W_N^{kn}$$

NPTEL

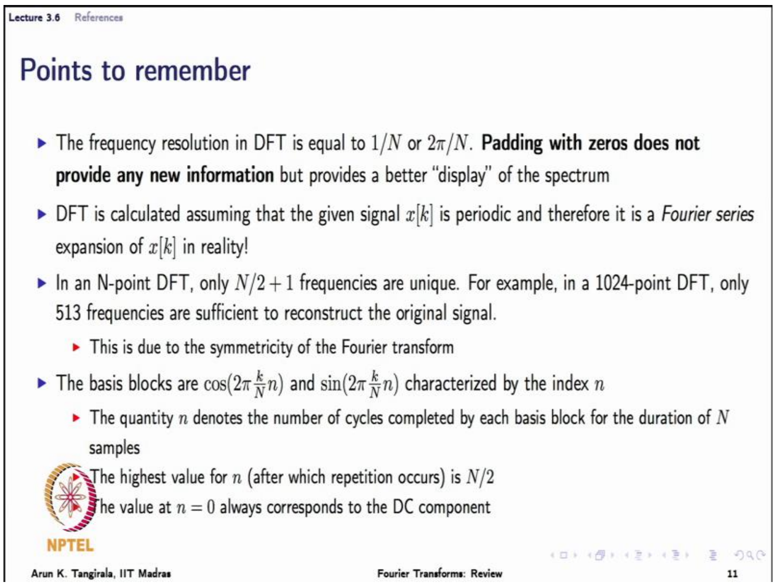
Arun K. Tangirala, IIT Madras — Fourier Transforms: Review — 10

So, the summarize this is the expression. These are the expressions for the analysis and synthesis, associated with DFT. So, the synthesis also known as inverse DFT. You have to be careful in many software packages. The 1 over N may not be there or may be there and so on, when you are computing the inverse DFT. You should check the documentation of that particular software. And then, manually include incorporate this 1 over N factor.

Because, normally in any software package, you will have the routine FFT to compute the DFT in a computational efficient manner. And the inverse FFT will simply implement the same thing, with x k replaced by the Fourier coefficients. And e to the minus j replace with e to the j. So, the inverse Fourier transform typically only give you this summation. Omitting this 1 over N factor, you should manually accommodate that.

Now, this zero padding something that is something, that I am mentioned here. We will talk about it a bit later. You can also introduce, rewrite this DFT equations by introducing a new quantity W subscript N. So, that it becomes easy for short hand calculations.

(Refer Slide Time: 20:36)



Lecture 3.6 References

Points to remember

- The frequency resolution in DFT is equal to $1/N$ or $2\pi/N$. **Padding with zeros does not provide any new information** but provides a better "display" of the spectrum
- DFT is calculated assuming that the given signal $x[k]$ is periodic and therefore it is a *Fourier series* expansion of $x[k]$ in reality!
- In an N-point DFT, only $N/2+1$ frequencies are unique. For example, in a 1024-point DFT, only 513 frequencies are sufficient to reconstruct the original signal.
  - This is due to the symmetricity of the Fourier transform
- The basis blocks are $\cos(2\pi\frac{k}{N}n)$ and $\sin(2\pi\frac{k}{N}n)$ characterized by the index n
  - The quantity n denotes the number of cycles completed by each basis block for the duration of N samples
  - The highest value for n (after which repetition occurs) is $N/2$
  - The value at $n=0$ always corresponds to the DC component

NPTEL

Arun K. Tangirala, IIT Madras Fourier Transforms: Review 11

But in practice, we use what is known as the fast Fourier transform. That, I will talk about it shortly, as well. So, there are certain useful points to remember, when we are using the DFT. First of all, the frequency resolution in cyclic frequency is 1 over N angular frequency, 2 pi over N. And many a times, you see these recommendations and suggestions of padding with zeroes.

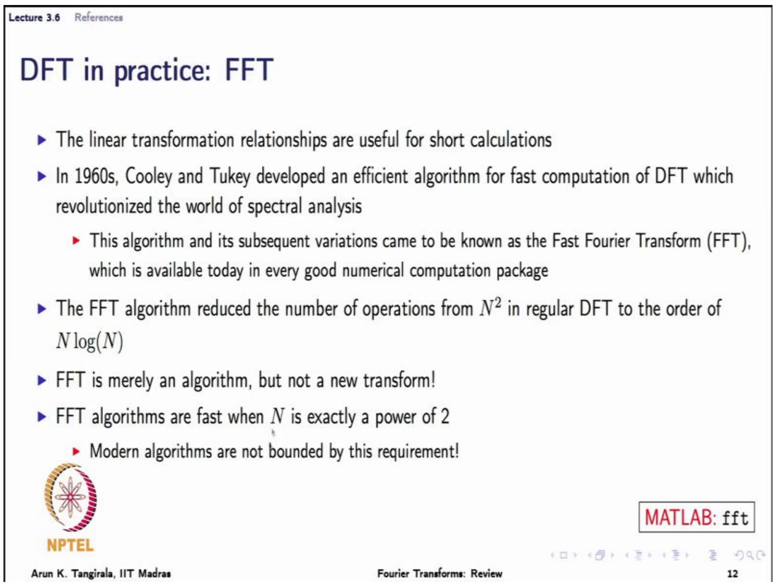
What we mean by padding with zeros is, if I have 98 or 100 observations, then the software may pad another 100 zeros to your data. And make you believe that, now you have 200 observations. Whereas you had 100 observations, now padding with 100 zeroes, brings it up to 200. And you compute the DFT of this. You feel, that you have much better resolution, because 1 over 200 is smaller than 1 over 100's. So, you feel that you have computed DFT over a finer grid. Unfortunately, padding with zeros does not provide any new information. That means, you will not get any finer information or finer resolution and so on. All it does is, it can improve the display. In fact, the other point associated with padding with zeros is, let us say I have 100 observations. I compute the DFT. So, I have computed DFT at 100 frequency points.

I would like to know, the value of the DFT, in between these two frequency points that I have computed. So, I need some interpolation. Padding with zeros is one form of interpolating the DFT, between two points that you have computed. So, it is a one form of interpolation. You can actually have many other ways of interpolation. So, what zero padding does is, it gives you better display of this spectrum. It does not provide any new information. In fact, sometimes padding with zeros is not recommended because, zero padding at the front and the end of the signal can introduce artificial discontinuities, which is also not recommended. So, there are a few articles on the web, that you can search for where you critically study the zero padding business. So, again to reiterate DFT calculated, assuming that the infinitely long signal that have not observed is periodic with the same as the length of the observation.

So, what you are actually computing is not Fourier transform. We are actually computing Fourier series coefficients you should remember that. And in an N point DFT, only N by 2 plus 1 frequencies are unique. For example, if I have a 1024 point DFT, then only 513 frequencies are sufficient, because of the conjugate symmetric and so on. So, this is simply the property, due to the property of the Fourier transform.

And the basis blocks are still cosines and sines cosine 2 pi k over N on n and sine 2 pi k over N on n. And the quantity n, the small n denotes the number of cycles completed by each basis block, in the duration of your data record. And the value at n equals 0, always corresponds to the DC component. We have mentioned this earlier also.

(Refer Slide Time: 23:51)


Lecture 3.6 References

DFT in practice: FFT

- The linear transformation relationships are useful for short calculations
- In 1960s, Cooley and Tukey developed an efficient algorithm for fast computation of DFT which revolutionized the world of spectral analysis
  - This algorithm and its subsequent variations came to be known as the Fast Fourier Transform (FFT), which is available today in every good numerical computation package
- The FFT algorithm reduced the number of operations from N^2 in regular DFT to the order of $N \log(N)$
- FFT is merely an algorithm, but not a new transform!
- FFT algorithms are fast when N is exactly a power of 2
  - Modern algorithms are not bounded by this requirement!

MATLAB: fft

Arun K. Tangirala, IIT Madras Fourier Transforms: Review 12

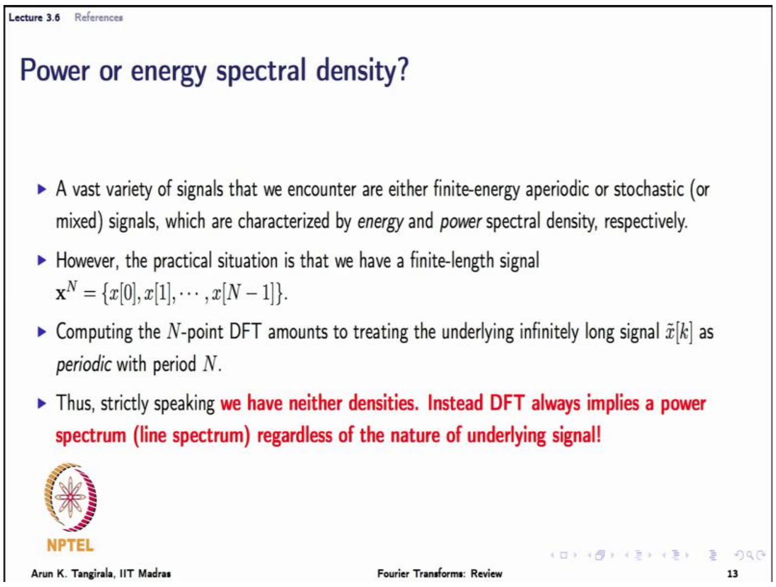
So, in practice as I said DFT uses an algorithm called fast Fourier transform, which was developed by Cooley and Tukey in 1960’s. It is a computationally efficient algorithm. Tier to that, it was quit difficult or in sense of computational burden to compute the DFT. The moment, this FFT algorithm came about it revolutionized the world of spectral analysis. Now, fast Fourier transform is not a new transform. It is essentially a simple, an algorithm for computing DFT.

Whereas, DFT is indeed at different transform from DTFT. What did the FFT do? It reduces a number of operations from N square. If you look at this summation in DFT and you have to compute over N frequency points. Then, the number of operations required is N square. FFT reduce that to N log N. Log N here, logarithm is to base 2. That is a considerable reduction. Otherwise, you require a huge compute us to or supercomputing to compute the Fourier transform.

And although for a long time FFT algorithms were fast, when N was exactly power of 2. That was also another reason, why zero padding would be done. If I have, I cannot always collect observations which have power of 2. I would, if I have 100 points, I would add 28 zeros to bring it up to the next power of 2, which is not such a requirement too, now with modern algorithms.

They can work without even zero padding. So, the mat lab routine that is used to compute the Fast Fourier Transform, its FFT itself. And so is the case with many other packages and so. But, you should read up the documentation on FFT.

(Refer Slide Time: 25:45)



Lecture 3.6 References

Power or energy spectral density?

- A vast variety of signals that we encounter are either finite-energy aperiodic or stochastic (or mixed) signals, which are characterized by *energy* and *power* spectral density, respectively.
- However, the practical situation is that we have a finite-length signal $\mathbf{x}^N = \{x[0], x[1], \cdots, x[N-1]\}$.
- Computing the N-point DFT amounts to treating the underlying infinitely long signal $\tilde{x}[k]$ as *periodic* with period N.
- Thus, strictly speaking **we have neither densities. Instead DFT always implies a power spectrum (line spectrum) regardless of the nature of underlying signal!**

NPTEL

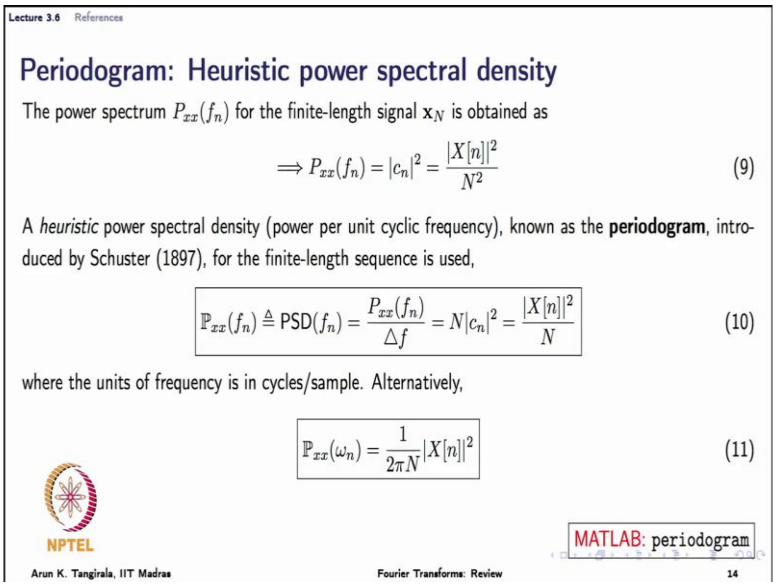
Arun K. Tangirala, IIT Madras Fourier Transforms: Review 13

Now, finally having computed the FFT, I would like to compute the power spectrum or spectral density and so on. So, the question is now, I am of all of this that we have learnt, which is that computing DFT amounts to assuming the underline infinitely long signal to be periodic and so on. Am I actually going to compute power spectral density, energy spectral density or what is that, that I am going to compute?

Because of the fact, computing DFT implicitly amounts to assuming the infinitely long signal to be periodic. Essentially, I can only talk of power spectrum. I cannot really talk of any density, strictly speaking. So, when I take the square magnitude of the DFT coefficients, I will only get power spectrum, but not necessarily densities. Because, implicitly I am assuming the underline signal to be periodic.

And we know for periodic signals, densities do not exist. So, what do we do now? How come people talk about densities and so on?

(Refer Slide Time: 26:51)



Lecture 3.6 References

Periodogram: Heuristic power spectral density

The power spectrum $P_{xx}(f_n)$ for the finite-length signal $\mathbf{x}_N$ is obtained as

$$\implies P_{xx}(f_n) = |c_n|^2 = \frac{|X[n]|^2}{N^2} \tag{9}$$

A *heuristic* power spectral density (power per unit cyclic frequency), known as the **periodogram**, introduced by Schuster (1897), for the finite-length sequence is used,

$$\boxed{\mathbb{P}_{xx}(f_n) \triangleq \text{PSD}(f_n) = \frac{P_{xx}(f_n)}{\Delta f} = N|c_n|^2 = \frac{|X[n]|^2}{N}} \tag{10}$$

where the units of frequency is in cycles/sample. Alternatively,

$$\boxed{\mathbb{P}_{xx}(\omega_n) = \frac{1}{2\pi N}|X[n]|^2} \tag{11}$$

NPTEL

MATLAB: `periodogram`

Arun K. Tangirala, IIT Madras Fourier Transforms: Review 14

Well, the densities that are talked about in practice are actually heuristic. And this is known as periodogram, which was d by Schuster as earlier as 1897. Even before, the Begets before FFT was invented. So, the power spectrums of the finite length signal… Now again, because I am assuming that the underline signal is periodic. I can only talk of powers spectrum. The power spectrum for a periodic signal, as we have seen earlier in lecture 3.3 is, mod C n square.

And we know from the relation between the DFT and the Fourier series coefficients, mod C n square can be return as mod X of n square by N square. Because, X of n is n time C n, remember. So, mod X of n square is nothing, but the square magnitude of your DFT coefficients. Your DFT coefficients are going to be complex. So, heuristic power spectral density known as a periodogram can be introduced, which is denoted by this different kind of font P.

Apologies for the confusion between the P for the power spectrum and this P for the periodogram, done. So, we will say this as P, double P. Just to distinguish. So, this double P of f of n is nothing, but a heuristic power spectral density. It is the power over delta f. Ideally, you cannot define density that way. But, density has the units of power per unit frequency. And therefore, we are dividing power by delta f. And that is nothing but, N C n square.

Because, delta f is 1 over N. Remember in DFT, delta f is 1 over N. Therefore, I have now mod X of n square over N as being the heuristic power spectral density, which is called the periodogram. And this is in terms of cyclic frequency. If I look at angular frequency, I have to include an additional factor of 2 pi. Because, then I will have to divide this by delta omega instead of delta f. That is the reason, I have 2 pi here.

Now, the mat lab comment, routine that computes the periodogram for you. It is a periodogram itself. We have to be careful. You should know whether, by default it returns the periodogram, in terms of angular frequency or cyclic frequency. It turns out, that the default is angular frequency. So, this 1 over 2 pi. That is involved. There is yet another aspect of periodogram, which I will demonstrate to you. At least, I will talk about it in the mat lab session that will follow this module.

(Refer Slide Time: 29:32)

Lecture 3.6 References

Bibliography I

- Bloomfield, P. (2000). *Fourier Analysis of Time Series: An Introduction.* 2nd edition. New York, USA: John Wiley & Sons, Inc.
- Oppenheim, A. and R. Schafer (1987). *Discrete-Time Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall.
- Proakis, J. and D. Manolakis (2005). *Digital Signal Processing - Principles, Algorithms and Applications.* New Jersey, USA: Prentice-Hall.
- Schuster, A. (1897). On lunar and solar periodicities of earthquakes. *Proceedings of the Royal Society*, 61, pp. 455–465.
- Tangirala, A. K. (2014). *Principles of System Identification: Theory and Practice.* CRC Press, Taylor & Francis Group.

NPTEL

Arun K. Tangirala, IIT Madras Fourier Transforms: Review 15

So, with this we come to a close of this module and this unit itself. In this module, we have talked about discrete Fourier transform, which is practically the most relevant one. Therefore, you should spends some time, understanding it and presented the concept of periodogram. It is a very certain concepts there, that we are not really computing any densities per say. Theoretically, it is a heuristic spectral density that we are computing.

There is a reason for introducing the spectral density. Because, we would like to know what is the power for unit frequency and more importantly, when we move to the world of random signals, which we do not in this course. Then, we can only talk of spectral densities. And I would like to have an estimator of spectral density and so on. So, there again the periodogram comes. Although, it is not a great estimator of the spectral density of random signals.

So, these are some of the references for your perusal. Please re through the relevant sections of these books. And the paper based Schuster is interesting. Now, Fourier transforms as I said, have enormous set of applications. You should really pick, go and do a quick literature review. Select any paper on Fourier transform. Application of Fourier transform, which is relevant to your feel or something that you are familiar in.

And study, how Fourier transform have been apply. In fact, there are papers which talk about, how our brain implements Fourier transforms very efficiently. And particularly, there is an article in ((Refer Time: 31:06)) which talks about, how the year implements Fourier transforms and filtering. Where it is able to recognize many sounds of simultaneously which around different frequencies without any ambiguity. So, with that note will close this module. And please go through the video lecture of the mat lab session that follows this module.

Thank you.