



**Electrochemical Impedance Spectroscopy**  
**Prof. S. Ramanathan**  
**Department of Chemical Engineering**  
**Indian Institute of Technology – Madras**

**Lecture – 08**  
**FFT Details, Frequency Range and Resolution, Cross Correlation**

**(Refer Slide Time: 00:14)**

Previous class	Today
<ul style="list-style-type: none"><li>• Overview of other techniques</li><li>• Experimental aspects of EIS<ul style="list-style-type: none"><li>• Type of equipment – Oscilloscope, Lockin and FFT analyzer</li><li>• Short cables, inductance at high frequency</li><li>• High frequency – double layer, mid &amp; low freq – reaction and diffusion</li><li>• Single sine and multi sine choices</li></ul></li></ul>	<ul style="list-style-type: none"><li>• Non-linear, Higher harmonics</li><li>• FFT, digital filters</li><li>• Multi-sine</li></ul>





In the previous section, we saw overview of the other techniques and also started with the experimental aspects of performing EIS. We saw different types of equipments based on that. There are certain things one need to take into account when you do the experiments. You can use something called Faraday cage. Faraday cage is a metal box in which you can keep the experimental setup. You have to use short cables if you want to avoid inductance of the high frequencies. Usually we will get information in the middle and low frequencies when we want to look for kinetics and mass transfer. High frequency information is usually for the double layer capacitance. There are different types of sinusoidal waves; one is called single sine or pure sine and another is called multi sine, which is addition of few sine waves together.

In this part, what I want to discuss is to look at the electrochemical aspects. When you go to electrochemical cell, the current versus potential is not linear. Earlier, we saw that current is related to potential in exponential fashion, at least for basic elementary reactions. For actual reactions in many cases, you will find that the elementary reactions can be exponential, but still the net current may not be exponential.

However, mostly it will still be nonlinear. Therefore, I want discuss on the cases when you have nonlinear current potential relationship. In addition to that, I will discuss few aspects of FFT and digital filters because the experimental data that you get it goes through some processing before you get it. Therefore, when you face any problem, you should be aware of how this is processed. So you can decide whether it is because of the electrochemical system or whether it is because of the way it is processed.

Now we will start with multi-sine.

(Refer Slide Time: 01:54)

**Single Sine**

- Linear Systems
  - $Y = A + B X$ .
  - Sine input  $\rightarrow$  Sine output
- Nonlinear system
  - Sine input  $\rightarrow$  Sine output + Harmonics....

$$y = 5x^2 + 3x + 2 = 5[E_{ac0} \sin(\omega t)]^2 + 3[E_{ac0} \sin(\omega t)] + 2$$

$$= 5E_{ac0}^2 \sin^2(\omega t) + 3E_{ac0} \sin(\omega t) + 2$$

$$= 5E_{ac0}^2 \left[ \frac{1 - \cos(2\omega t)}{2} \right] + 3E_{ac0} \sin(\omega t) + 2$$

$$= 2 + 2.5E_{ac0}^2 + 3E_{ac0} \sin(\omega t) - 2.5\cos(2\omega t)$$

$$= 2 + 2.5E_{ac0}^2 + 3E_{ac0} \sin(\omega t) - 2.5\sin\left(2\omega t + \frac{\pi}{2}\right)$$

$E(t) = E_{dc} + E_{ac0} \sin(\omega t)$

$$i = Fk_0 e^{bE}$$

$$= Fk_0 e^{b(E_{dc} + E_{ac0} \sin(\omega t))}$$

$$= Fk_0 e^{bE_{dc}} e^{bE_{ac0} \sin(\omega t)}$$

$$= Fk_0 e^{bE_{dc}} \left[ 1 + bE_{ac0} \sin(\omega t) + \frac{(bE_{ac0} \sin(\omega t))^2}{2!} + \frac{(bE_{ac0} \sin(\omega t))^3}{3!} + \dots \right]$$

$$= Fk_0 e^{bE_{dc}} [1 + bE_{ac0} \sin(\omega t)]$$

**Note:** When we say that "dc potential is not important" we mean that  $E_{dc}$  is not important to decide if it is OK to assume that the system response is linear

All of us are familiar with the linear system where you have a relationship saying  $Y = A + BX$ .  $X$  can be a vector;  $Y$  can be a vector. Assume you are giving a sinusoidal input (call it as  $X$ ),  $Y$  will be a sinusoidal output, with a constant offset or with a phase difference. Whatever it maybe, it is going to be sinusoidal. When you have a nonlinear system, a sinusoidal input will give sinusoidal output + harmonics. Harmonics means, if the frequency of the input, say 0.1 hertz, the output will contain 0.2, 0.3, 0.4 Hz. It means the output contains integer multiples of the base frequency and that base frequency is called fundamental frequency. We will call it as the response at fundamental, response at the second harmonic, third harmonic, fourth harmonic and so on. The responses other than fundamental frequency are higher harmonics. For example, consider this equation  $Y = 5x^2 + 3x + 2$  and let us substitute the sinusoidal wave as  $E_{ac0} \sin(\omega t)$ . That is what we normally use for potential sin wave. When you substitute it, you will get.

$$5[E_{ac0} \sin(\omega t)]^2 + 3[E_{ac0} \sin(\omega t)] + 2 .$$

When you expand it, u will get as:

$5E_{ac0}^2 \sin^2(\omega t) + 3E_{ac0} \sin(\omega t) + 2$  and replacing  $\sin^2(\omega t)$ , you will get:

$$5E_{ac0}^2 \left[ \frac{1 - \cos(2\omega t)}{2} \right] + 3E_{ac0} \sin(\omega t) + 2$$

I do not want to write as  $\sin^2$ ,  $\sin^3$ ,  $\sin^4$  etc. But I want to write it such that, the first one would come in Taylor series expansion, the second part which is  $\cos(n\omega)$  or  $\sin(n\omega)$  i.e.  $2\omega$ ,  $3\omega$  etc will come as Fourier series expansion. So the above equation after rearranging, you can get:

$$2 + 2.5E_{ac0}^2 + 3E_{ac0} \sin(\omega t) - 2.5 \cos(2\omega t)$$

So the first part is the DC offset (refer video). This is the fundamental frequency response. 2<sup>nd</sup> part is the second harmonic. So instead of writing it as cosine, I can also write it as  $\sin 2\omega$  with a  $\pi/2$  phase. (Notice that square term has given rise to a constant and  $2\omega$ ). If I have a cubic term (power 3), it will give rise to  $\sin^3$ ,  $3\omega$  and  $\omega$ . If I have  $\sin^4$ , I will get  $4\omega$ ,  $2\omega$  and 0 which is basically a constant. So if I give  $\sin^n$ , I will get  $n\omega$ ,  $(n-2)\omega$  and so on until it ends at  $\omega$  for odd number of n and ends at a constant for even number of n.

So when I get higher and higher nonlinear terms, correspondingly I will get higher and higher harmonics. Now we know that for simple reactions, the current and potential are related exponentially, which means when I write potential as DC + AC:

$$E(t) = E_{dc} + E_{ac0} \sin(\omega t)$$

I would write current for a very simple reaction as:

$$i = Fk_0 e^{bE}$$

F is the Faraday constant, bE is the exponential factor, k is the pre-exponential factor.

We assume concentration of species involved is a constant. So I instead of multiplying by the concentration of the reactant, I can combine it in the equation if I assume C is constant throughout the experiment. Here, we can substitute for E, I will get:

$$i = Fk_0 e^{b(E(t)=E_{dc}+E_{ac0} \sin(\omega t))}$$

At this level DC potential may or may not be small.

It may be 0, it may be 100 millivolts, 200 millivolts, 600 millivolts, maybe -600 millivolts with respect to the equilibrium potential. Now we are not making any assumption about the DC potential. At this stage I would separate the DC and AC component and then expand the AC component in Taylor series:

$e^x$  you can write it as  $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!}$  and so on.

Now if I assume that the product  $b$  and  $E_{ac0}$  is a small number, then I can truncate this series after this term (Refer video). I can say  $E_{ac0}$  is very small. Therefore, the product  $B_{ac0}$  is also small.  $E_{ac0}^2$  is significantly smaller than  $E_{ac0}$ . if you say  $E_{ac0}$  is  $10^{-3}$  volts, which is 1 millivolt, it is going to be  $10^{-6}$  for  $E_{ac0}^2$ . So you can neglect those terms. With this assumption, we can make an approximation and say this is approximately equal to:

$$i = Fk_0 e^{bE_{dc}} [1 + bE_{ac0} \sin(\omega t)]$$

So if I say that  $E_{ac0}$  part - amplitude of the sin wave be small, then I can truncate this and say I will get a linear response. System is nonlinear, but my input is very small. Therefore, the output I can approximate it as a linear. So if I draw it in a curve, this curve is nonlinear. I can take this and approximate this area by a small straight line. I can approximate this area by a different straight line (refer video). I cannot approximate a larger area by a straight line because that will be a poor approximation. So DC potential is not important for this approximation. The AC perturbation or amplitude has to be small. When that is small, we can say that this can be linearized. When it is linearized, the analysis becomes lot simpler and that is the purpose of doing this.

**(Refer Slide Time: 07:53)**

If  $E_{ac0}$  is small, then we can assume that the response is linear, regardless of the magnitude of  $E_{dc}$ .  
The actual value of  $E_{dc}$  will affect the response current value and is important!

**(Refer Slide Time: 07:57)**

### Single Sine

- Linear Systems
  - $Y = A + B X$
  - Sine input  $\rightarrow$  Sine output
- Nonlinear system
  - Sine input  $\rightarrow$  Sine output + Harmonics....

"At various levels" means

"at various dc bias values"

$$i(t) = i_0 + i_1 \sin(\omega t + \phi_1) + i_2 \sin(2\omega t + \phi_2) + \dots + i_n \sin(n\omega t + \phi_n) + \dots$$

- In general, higher harmonics will be present
- When input perturbation amplitude is small, higher harmonics will be negligible
- How small is small?



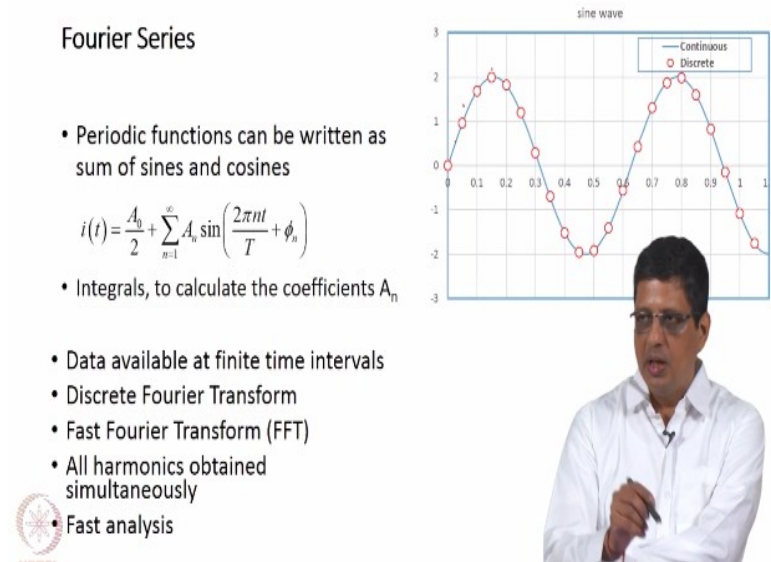
So normally, you would expect the current would come in the first order and with harmonics. First order is the fundamental and with harmonics. However, if the perturbation is small, you can say that it is going to be linear system or it is going to respond like a linear system at that DC potential. If we move to another DC potential, it is going to respond as if it is slightly different linear system.

We are not saying that system is linear throughout the entire potential or current range. We are saying, around a particular range, if you move a little bit I can approximate this by a linear system. Therefore, if it is linear system, you can do this perturbation in one level and then say that throughout this range I can tell this is what it is going to be. But because it is nonlinear in real life and nonlinear analysis is hard therefore we want to perform linear analysis.

We have to conduct experiment at multiple locations and then say at multiple locations we are approximating by few linear segments and this will help us understand or reconstruct this entire system behaviour at various levels. Now how small is small? People use 1 millivolt, 5 millivolts, 10 millivolts etc. 20 millivolts will be probably stretching it. If you say 1 millivolt, people would not question that it is linear or nonlinear, it is definitely good enough to approximate as a linear system, but the noise level will be high. Noise level in the system is random noise from outside. That is going to be independent of the perturbation we are applying. So if you apply a large input, you will get a large current output. Compared to the noise this may be significant.

Therefore, the so-called signal to noise ratio will be better or higher. If you give a small amplitude perturbation, signal to noise ratio will be poor; but you can claim that I can approximate that the response as a linear system without any problem. If you go to large perturbation, the response will be good. However, you may have nonlinear effect which means you may have  $2\omega$  and  $3\omega$  contributions (higher and higher contributions).

**(Refer Slide Time: 10:02)**



Now you can take any function in the time domain and convert it to frequency domain. So any periodic function can be written as a sum of sin and cosine. Here, I have shown it as sin with a phase. It is same as sin and cosine. Also, you can write  $i$  as function of time, you can write it in terms of constants and other coefficients. The coefficients can be calculated using integrals. You can find it in any of the standard books or internet. I am not going to describe them here.

I want you to note the point that the blue line here is the continuous function, the red colour circles represent equal time interval, data (refer video or slide). It is a discrete function. So when you take data (current or current which is converted to potential), it is recorded in the system only at specific intervals. It is not recorded in between these intervals.

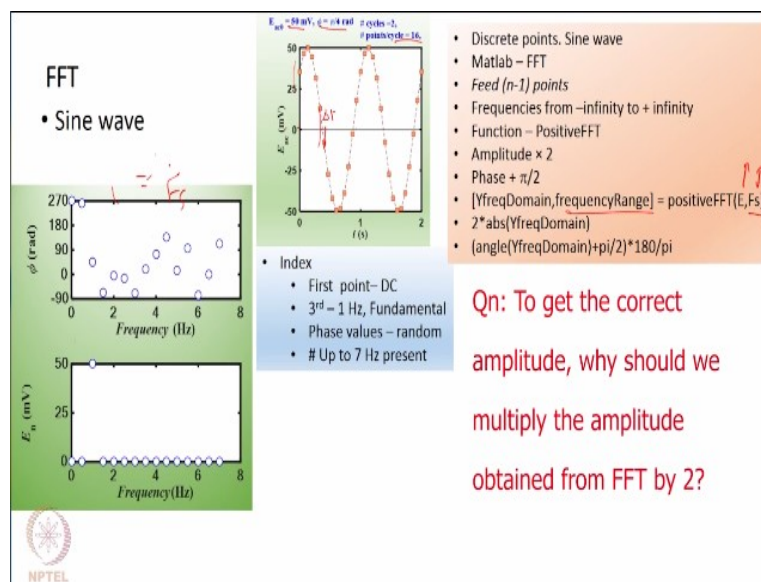
So in between, if the data jumps up and down, we do not know that. When you do Fourier transform of this discrete data, you can employ algorithms, which are meant for discrete data. If you have a function and if I want to specify a function in an interval, in theory, I actually have to give you infinite number of points. I cannot just tell you this function has this value.

When I specify this, I do not tell you anything about what is the value of the function in-between.

If I want to specify at all the time intervals, I will have to give from zero to 1 second, but infinite number of data points. We have finite number of data points which we are going to handle and for that, we use discrete Fourier transform and in discrete Fourier transform, one of the algorithm is called fast Fourier transform FFT. The advantage of Fourier transform is that, you can obtain all the harmonics simultaneously. With one transformation, you will get all the harmonics, which means, all the harmonics that can be taken from the data. If you have infinite number of points, you will get infinite amount of harmonics. If you have finite set of data, you will get only finite set of harmonics. The system may actually have more harmonics, but you will not be able to extract it from the data.

The analysis is fast and I want to show you some examples (not formula). You can get the formula from books and you can get the libraries for Matlab or any other programs. Most of them will have inbuilt libraries.

(Refer Slide Time: 12:25)



Here, I have constructed a sine wave (this is done in Matlab). 50 millivolts is the amplitude and you can see it goes from 0 to 50  $\pm$  50. I have given a phase of  $\pi/4$  radians or 45 degree. Instead of starting at 0 at zeroth time, 0 will actually occur here (refer video). At 0 time, it has a phase and corresponding to 45 degree, you are getting actually 90 degree and so on. And I have constructed 2 loops and I have given 16 points per cycle. You will get a finite number of

points per cycle in actual data acquisition and we use a function called FFT in Matlab. But it is wrapped around another function called positive FFT. In theory, your data goes from - infinity to + infinity or the sine wave goes from - infinity to + infinity. Any of the programming language by default will give you frequencies also from - infinity to + infinity. But they are folded in a function called positive FFT. It is just a wrapper library around the FFT program. It gives you frequencies only from 0 to whatever positive number you get. The way it works is:

(Refer video for better understanding of this paragraph). I have a sine wave; I will not give it until the end of the sine wave here. I will give it with one point +, because the way it works is, it is going to take this and it is going to replicate this  $n$  number of times. Let us pretend and say this is the first point, second point, third point, fourth, fifth. So what I am going to do is, it will come here and in next time interval, it will step and then take the first point and put it. In next time interval, it will take and put this point and so it will continue. Instead of this, if I give you points until this, means first point is 0, origin and so on and the last point is here, it is going to shift to the next time interval and take this as the point and then reconstruct that. That is not a sine wave; it is a distorted sine wave. So if I want to give 3 cycles, I will give 1 cycle, 2 cycle, third cycle and I will stop just before the end point. So I will give data for  $n$  number of cycles, I can give one full cycle, second cycle or whatever  $n$  number of cycles. But the last point I will cut it before I feed into this, otherwise you will be scratching your head why it is not coming the way you expect.

This is the implementation in Matlab. If you write your own program or if you take libraries, you have to be aware of what the constraints are, what the way to give the input and take the output is. Because of the way that works (it cuts off the frequencies from - infinity to + infinity), we are wrapping it and getting from 0 to +, whatever value of amplitude you get you have to multiply by 2.

So this is the command we have used here.  $E$  is the vector. We do not give time vector; we just say this is the sampling frequency. So if the time is going from 0 to some value, we have 10 number of points, we have so many intervals,  $\Delta t$  is time interval. From one point to another, what will be the time interval. Take that and take an inverse of that and it will tell you what the sampling frequency is.



So you need to give sampling frequency, vector and it will give you the frequency range and the Fourier transform result. This is the implementation. I do not know how it is done inside the Matlab and I am not going through that. You can use an FFT program and get this. When you get the value, multiply by two in this implementation to get the amplitude. You have to add a phase of  $\pi/2$  because your exponential value is  $\cos \omega t + j \sin \omega t$  and this is done as a complex transform. The Y vector in the frequency domain is actually a complex vector. The absolute value tells you the magnitude and the phase value can be obtained by using the command `angle` in Matlab.

You get the phase because you are giving a sinusoidal wave and not  $e^{j\omega t}$ . We are giving only sinusoidal wave and you have to account for the phase offset. So whatever value you get in the phase add  $\pi/2$ . For presentation, I am multiplying by  $180/\pi$  so that it is converted to degree. Otherwise you will just plot it in the radians.

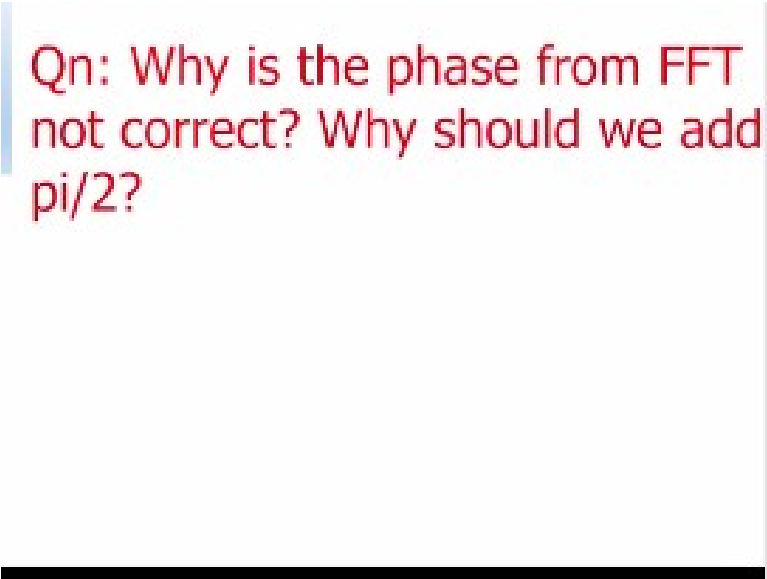
What has happened in the FFT is, it has split the energies or whatever amplitude you are getting and gave it in + and - frequencies. It has distributed it equally. That is why the FFT algorithm has been implemented in matlab. Now, we are throwing away the entire negative one. Half the energy is gone there. Only the remaining half energy is here. So we have to multiply by 2 to get this.

(Refer video for better understanding). Let us say you get a vector like this and what you get in the Y frequency domain is going to be some number a, b + jc, d + je etc. First number is with 0 phase. It is given some phase here, but it is the DC. So when you write in Fourier series, first you write  $a_0$  and then you write  $a_1 \sin \omega t$ ,  $a_2 \sin 2\omega t$  with phase. So  $a_1$  is without any phase, there is no sine or cosine there. In this case, because we are giving a pure sin wave, we do not expect to see any constant value. All that we need to get out of that is, it is a  $\sin \omega t$  with a phase of  $\pi/4$ . First point is the DC and that is 0, magnitude is 0. Second point; it is not showing anything. Third point, it is at 1 hertz, this is 0 magnitude, this is half hertz. This is 1, 1.5, 2 and so on (refer video). Therefore, what it says is, the magnitude is 50 millivolts at 1 hertz and everywhere else it is 0 and that is what we expect. We are giving a pure sine wave. It does not go up to infinity; it goes up to number 7. We can see only up to 7 hertz, beyond that we do not know what it contains. Phase value at this 1 hertz is 45 or  $\pi/4$  radian. Phase value everywhere else is meaningless because if you take a vector and say it is magnitude is 0, phase value does not have a meaning. Phase has meaning only when

magnitude is nonzero. It does not mean the Matlab will give you as 0 phase in those things, it just means that you should learn to ignore those. Everywhere else phase is meaningless.

When you have non-zero component, you should look at the phase and see whether it is coming correctly. So up to 7 harmonics are present, we have 16 point per cycle. There is a criterion called sampling criteria. If you want to see 8-hertz data, you should sample at least at 16 hertz. Therefore, you should have minimum twice the frequency that you want to sample. And typically, you want more because there is going to be noise in the system.

**(Refer Slide Time: 19:15)**

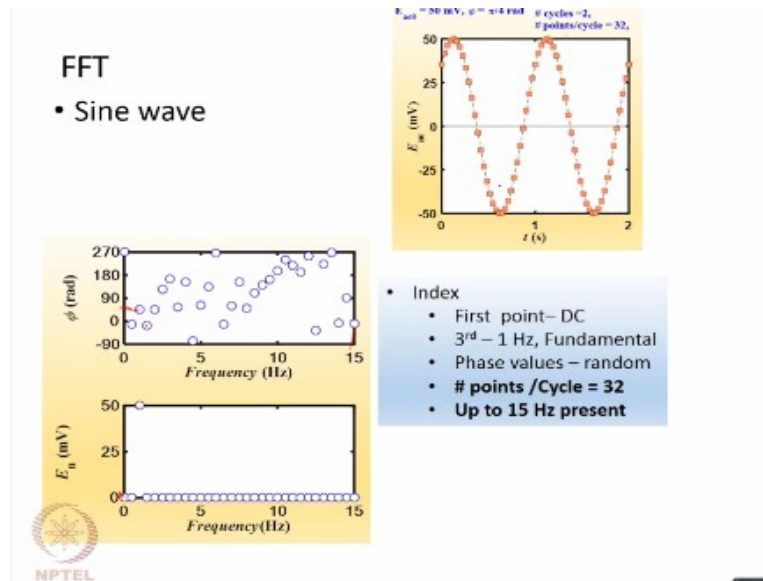


Qn: Why is the phase from FFT not correct? Why should we add  $\pi/2$ ?

Student is asking doubt to faculty:

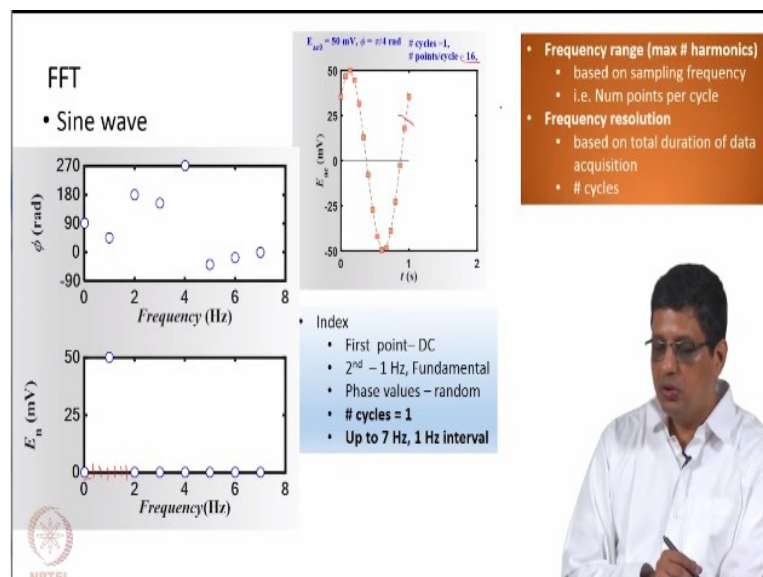
If you give only sin, you will not get the correct phase and it is off by 90 degree. If you give the complex number, cosine + sine in the complex format for this, then it will give you the correct result. Just like the frequency split into negative and positive, this also expects a complex input for this. We are giving only sine here, so I would get the effective phase if I add  $\pi/2$  here. Because I am not giving it the complex feed, it is more of the implementation in the Matlab. It is more of the way it is implemented for a general purpose, we are writing a program to wrap it and just cutting it to the positive frequencies. If you have image processing you will have 2D data, you can do FFT on that, you can have 3D data, do FFT on that. But you have to be aware of how to use that library.

**(Refer Slide Time: 20:02)**



Now what I have done here is to increase the number of points. Sampling frequency is higher. Now I have 32 points per cycle, you can see it is more crowded here and if I do FFT now, still first point is DC, second point is half hertz, 1 hertz is 50 millivolts, phase values are random for everything else, but at 1 hertz, it is still 45 degree. It will show you 45 degree if you do it correctly. Now the thing is, it is present up to 15 hertz. So if I want more harmonics, I should sample at a higher frequency.

(Refer Slide Time: 20:41)



Sample frequency tells us how frequently we take data and if we take data at 100th hertz, the best I can get out of this is 50 hertz. I cannot get a sine wave of let us say 100 hertz by sampling at 100 hertz frequency, because in one cycle I will sample only one point. The minimum I require is 2. Typically, I will need more. So here, it is a perfect sine wave. Therefore, I show you data up to this ((...)) (21:09).

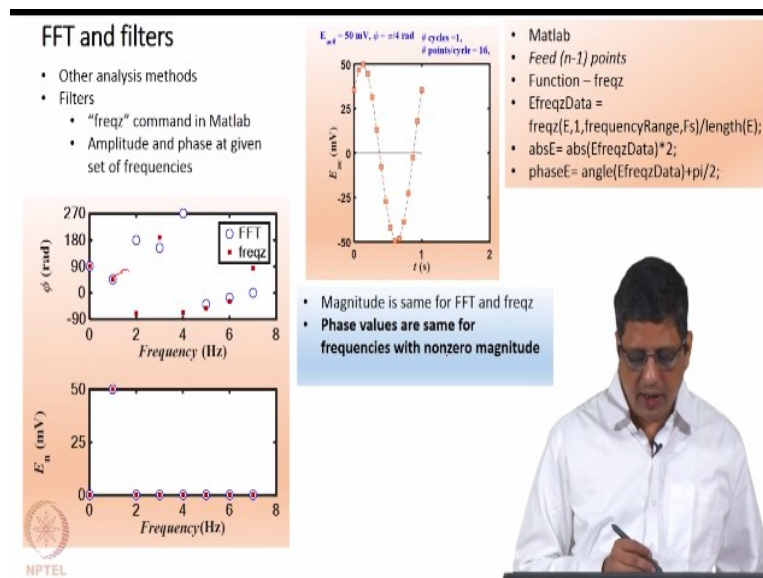
In reality, if I have other waves and it is distorted, and if there is noise, I will not get good accuracy in high frequency regime because I am close to the Nyquist criteria in terms of sampling frequency. Here it is not going to be a much of a problem because it is a pure sine wave. Now what I have done is sampled only up to 1 cycle. I have gone back to 16 hertz of sampling, 16 points per cycle.

(Refer video) I have shown it in the same scale and then shown it here that I have truncated up to this, when I feed it to the program I will feed up to this level and then do FFT. 16 means, I should expect up to 7 hertz or I get up to 7 hertz. Also notice the interval is 1 hertz now. So if I want to get data at fine intervals, I want to know the amplitude and phase at fine intervals of 0.1, 0.2, 0.3, 0.4 hertz. Then I need more time, I need to go for more cycles here. So the summary is, the frequency range 0 to whatever number (7 or 8 here), 0 to 15 or 16 (depends on number of points per cycle) is the sampling frequency. The resolution 0, 1 hertz, 2 hertz; that is one way of moving, 0, 0.5, 1, 1.5, 2; that is another way of going, 0, 0.1, 0.2, 0.3; that is another way of going. The resolution I can get in the frequency depends on total duration.

(Refer video) Another way to look at is number of cycles. So here, 1 is because of the time I had taken here, you can take  $F$ ,  $2F$ ,  $3F$ ,  $4F$  and if you want less than  $F$ , 0.1, 0.2, 0.3 and so on it needs to be done for longer time. Now in all these arguments, we assume that this frequency or sampling interval is identical. We are taking data at 0, 0.1, 0.2, 0.3 so many seconds or milliseconds, it is also possible that you can have data at non-uniform time interval. And there is a method of doing Fourier transform for that we are not going to get that at all. We are going to say our sampling intervals are uniform, but we are going to look at data only with that assumption. Now when you actually take data, you send the commands, software sends a command and it usually goes to a data acquisition board, collects data, you would say take data at this interval for so much time. And there is a clock within that chip or board which is used to collect at those intervals. It senses all the time, but it will save data at particular interval. That is stored in a buffer and then transferred to the main CPU or to the hard drive. It does not send command; sends another command after 10 millisecond; sends another command from the operating system. It sends the command to the chip saying acquired data for 'this' much time at 'this' interval.

And the chip has a buffer, you cannot ask it to hold infinite number of data points which means you cannot say to give high frequency sampling for long time. Each chip comes with a buffer, so there are limitations. Depending on which one is used in this, you can say sample every second for 100 second. It is possible to do it. You cannot say sample every 100 second for 1 year. The chip size may be sufficient. But the chip also has a limitation on what is the maximum and minimum sampling frequency it can handle and you cannot exceed that. Synchronisation will not happen by sending it from operating system. So time interval is done at the chip level. So you cannot tell, I want many harmonics and fine resolution. Only up to some level, you can do it. It is not that you have lot of storage and I do not mind waiting for long time. If the command is going from the operating system and data is taken up every minute, time accuracy would not be good. If it is done at the chip level, time accuracy is better. In general, because of all these, there are limitations on how it is done in the actual chip level. There are limitations on how data is acquired and given to us at the instrument level.

(Refer Slide Time: 25:25)

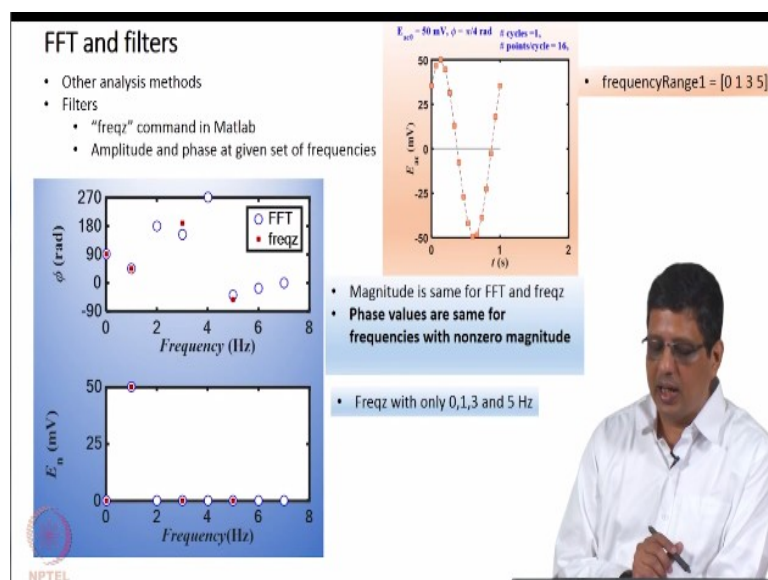


There are other filters, digital filters that can be used to process the data; an example is freqz command in Matlab. If you look at the Fourier transform earlier, you will get data at finite time intervals, you will get data at  $f$ ,  $2f$ ,  $3f$ , or you can get data at  $1/2 f$ ,  $1f$ ,  $1.5 f$ ,  $2f$ . If I go for longer time, you can also imagine I get 0.1 hertz, 0.2 hertz but the frequencies are also at the even intervals.

But let us say I do not want all this, I want data at few frequencies, I want to look at data at 1, 2, 5 and of course you can take the vector and do this. But you can also use what is called filters or digital filters and achieve the same objective. Again you need to feed n-1 points for this implementation. The way it called is, it can be used to generate many other types of filters. This is the general-purpose command (refer slide or video). This is the vector of data 1 and I give the frequency range, I can specify here that it is going to be 0 1 2 3 4 5 6 7 hertz or 0 1 2 5 hertz whatever I want here. Sampling frequency has to be given and because of the way it is implemented, you also had to divide by the length of the vector to get the correct value, equivalent of FFT and you need to multiply by 2 to get the magnitude add phase offset of  $\pi/2$ .

And you can do that and compare with the results in FFT. Here I have taken 1 2 3 4 5 6 7 8 just like what we would get in FFT. Now when you look at this the red - fill square is the freqz result and FFT is given by the open circle (refer video). And they are the same for 1 hertz where we have significant amount of information. At all other locations this magnitude is 0, phase value may or may not match, it does not matter. Whenever the magnitude is nonzero, this command works correctly. I can also say I do not want it everywhere, I want it only at 0, 1, 3, 5.

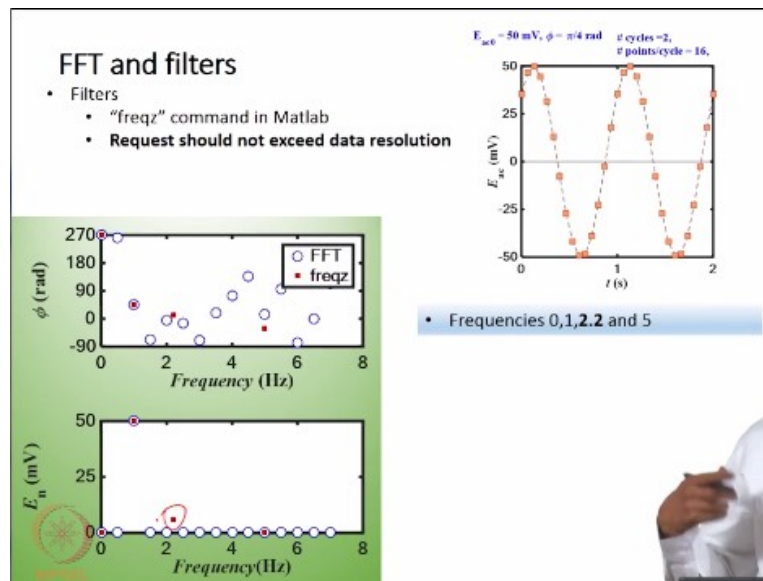
**(Refer Slide Time: 27:20)**



And of course at 1, it will match, it does not matter whether the remaining of it may or may not match. However, with this data I can ask this filter to give me data at 1.5, 2.5 or any

number I want. Here I got only 16 points per cycle and I got one cycle. So I can go up to 7 hertz, I can go at a frequency step of 1 hertz.

(Refer Slide Time: 27:50)



I can go to 2 cycles, I can go at a frequency step of 1/2 hertz. If I say give me data at 0, 1, 2.5 and 5, it gives data at 0, 1, 2.5, and 5. I can also tell the filter to give data at 0, 1, 2.2 and 5. It will do the mathematical calculation except that it will give me incorrect information. Unless you are aware of it you will process it further and get results, but they would not be correct.

When they implement it in the hardware also, they will have to use some tricks, single sine is usually not a problem, you will have problem only for multi-sine. I want you to go through the details and see, but right now I want to emphasize that if you ask it to give data at a higher resolution than possible theoretically, here I want data at 0.2 resolution, it gives data at 0, 1/2, 1, 1.5, 2, 2.2 that resolution is not possible. I can get only at 0.5 resolution, if I want data at 0.2 resolution; I need more number of cycles. If you do Fourier transform by FFT and get the array you will see you do not have 2.2 there. If you do this filter, it will give you an answer, but you will get a wrong answer, you would not get any indication saying we have a problem.

(Refer Slide Time: 29:09)

## Lock-in Amplifier

- Cross Correlation

$$i = i_0 + i_1 \sin(\omega t + \phi_1) + i_2 \sin(2\omega t) + \dots$$



Qn: We specify resolution by setting points, right?

**“Professor - student conversation starts”**

**Student:** Sir for the freqz, you specify the resolution by setting points.

**Prof:** You do not specify resolution; you will tell that these are the frequencies at which I want the power, magnitude and the phase.

**(Refer Slide Time: 29:21)**

Qn: So, it decides the resolution based on the decimals (in data given), right?

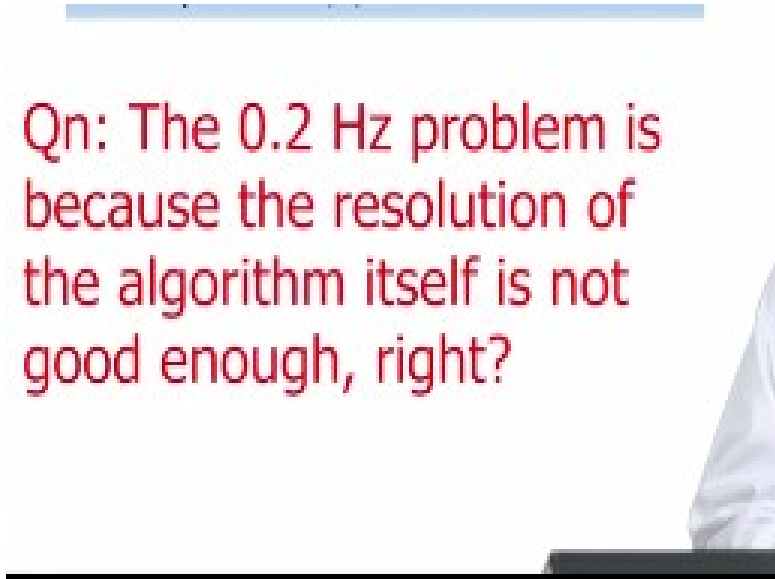
**Student:** So it decides the resolution based on the decimal.

**Prof:** It does not decide the resolution. It applies the algorithm whatever it has inside, I do not know the algorithm there. Internally it uses FFT at some level, but basically you have digital filter and when you are using it, you have to be careful. Whether you are using it in the software, or whether it is implemented in the hardware, you have to be careful in the sense that when you ask, that time ‘you’ have to verify whether your request is correct or not for the



given set of data, for the given time interval and given total time. If it is not correct, you will not get the correct result.

(Refer Slide Time: 29:55)



Qn: The 0.2 Hz problem is because the resolution of the algorithm itself is not good enough, right?

**Student:** So the 0.2 Hz problem is because the resolution of the algorithm itself does not go with the point, right?

**Prof:** No, no, if I give data up to 10 cycles, it will go in 0.1 interval. It is not the algorithm, which has the problem. My feed and request of the algorithm; that combination is not correct. If you give correct feed and the correct request, it will give me the correct data. If you give this feed, I should not ask anything better than 0.5. I can ask at 0.5, I can ask at 1 hertz. I cannot ask 0.7. If I want 0.1, I should give that many number of cycle. If I want 0.1 resolution, I should give 10 seconds. If I want 0.01 resolution, I should give 100 seconds of data, and then it will give me the correct result. Problem is, if I give a wrong request I will get the wrong data and probably I will proceed with the analysis without knowing that there is a problem. That is more dangerous. If you do not get, it is okay. You have a problem, but you know that you have a problem. The real problem is when you get the data, which is not correct, and you assume or you believe it is correct. It is true in experiments also. The equipment gives you problem and you do not get data it is not pleasant, but you know there is a problem and you will try to fix it. If it gives you garbage data and you do not realize it is garbage data it is really a problem, because you will come to conclusions, which are not correct.

**“Professor - student conversation ends”**

(Refer Slide Time: 31:00)

### Lock-in Amplifier

#### • Cross Correlation

$$i = i_0 + i_1 \sin(\omega t + \phi_1) + i_2 \sin(2\omega t) + \dots$$

$$\sin(A + B) = \sin(A)\cos(B) + \cos(A)\sin(B)$$

$$i_1 \sin(\omega t + \phi_1) = i_1 [\sin(\omega t)\cos(\phi_1) + \cos(\omega t)\sin(\phi_1)]$$

$$= i_1 \cos(\phi_1) \sin(\omega t) + i_1 \sin(\phi_1) \cos(\omega t)$$

$$= i_{1s} \sin(\omega t) + i_{1c} \cos(\omega t)$$

$$i = i_0 + i_{1s} \sin(\omega t) + i_{1c} \cos(\omega t) + i_{2s} \sin(2\omega t) + i_{2c} \cos(2\omega t) + \dots$$

$$i_{1s} = \int_0^T i(t) \sin(\omega t) dt$$

$$i_{1c} = \int_0^T i(t) \cos(\omega t) dt$$



- Implementation – slower
- Better Signal to noise compared to FFT
- Multiple harmonics need multiple calculations



Similar to FFT, there is another technique called cross correlation. Let us see you have a sine wave or you have a signal, which can be written in Fourier series like this.

You can write  $\sin A + B$  as a combination of sine and cosine, same way you can write this  $\sin \omega t + \Phi$  as

$$\sin \omega t \cos \Phi + \cos \omega t \sin \Phi,$$

which means when I write it in terms of  $\omega t$  alone, I can combine this and say for a given  $\Phi$ , this is the constant (refer video). So I can expand this as:

$\sin \omega t + \cos \omega t$ .  $\sin 2\omega t + \cos 2\omega t$ . This is another representation of Fourier series.

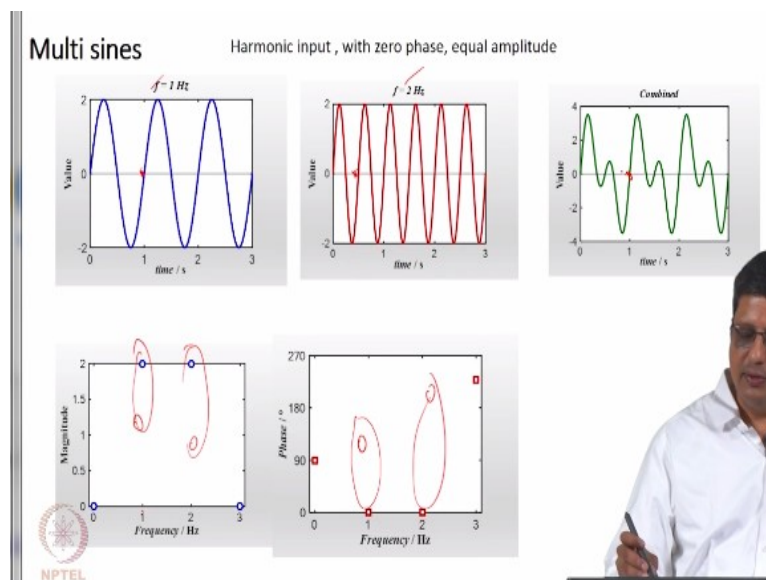
Fourier series can be  $\sin \omega t + \Phi$ , it can also be  $\sin \omega t + \cos \omega t$  with different coefficients in front of them. Now the sin functions are known as orthogonal functions. It means, if we take a period  $T$ , I will get 0.  $\sin m\omega t$  and  $n\omega t$ , as long as  $m$  is not equal to  $n$  I will get 0. If  $m = n$ , I will get a value, I can normalize it and get the factor out of this (refer video).

Likewise, you can use this for cosine also; sin and cosine function are orthogonal, which means if we take the function  $I$ , which can be expanded like this (refer video) and then I do the integral, I will get the component only corresponding to the sin and everything else will go to 0. Same way I can do it for cosine. So I can get the coefficient of 1S and 1C by doing this integration (refer video). I can get the coefficient of 2S and 2C by multiplying by  $\sin 2\omega t$  and  $\cos 2\omega t$  and perform the integral.

This method is called cross correlation. I can get the phase by doing the sin and cosine, I can get  $i_{IS}$  and  $i_{IC}$  which means I can back calculate and get  $\Phi_1$ . This method is slower compared to the FFT analysis and for each harmonic, you have to do the calculation. In FFT, once you do the calculation you will get 0 to 7, 0 to 15 whatever I have shown in the example, you will get it. It is not that you will get only the first harmonic and you have to do extra effort to get the second harmonic. You go through the effort and you get the entire vector out of it. Here you have to do it multiple number of times and these are usually done at the chip level. It is not that data is taken and then processed in the software. Then you can even try parallel processing. These are done with the chip level and usually you get only one harmonic, fundamental harmonic out of the lock-in amplifier.

Although you can set it for other harmonics, but if I want to acquire data with the normal lock-in amplifier I will have to run it once at the fundamental mode; run it again at the second harmonic, and run it again for the third harmonic and so on. So it is going to be slower and system might have changed between the runs. Run time becomes much longer than what you would get in FFT. But this also happens to be more accurate in terms of ability to reject noise. So implementation is slower, multiple harmonics need multiple calculation; it has a better signal to noise ratio when you compare to FFT.

**(Refer Slide Time: 34:30)**



So whatever we have seen so far is for a single sine, what happens when I add 2 sine waves? In this example, I am showing you 1 hertz and 2 hertz (refer slide or video). I am showing them as continuous lines; it is of course generated using discrete points. You can see the combine wave, they have 0 phase, they have equal amplitude of 2 (whatever that value it is, it

is not specified whether it is volts or millivolts or anything in the figure). Total is not going to go until four, because these are at different frequencies and you will get a combined wave. This has a period of 1 second, this has a period of 1/2 second combined wave has a period of 1 second, that means if I give you this data you can replicate it n number of times you will get the wave that we generate from this.

When we do Fourier transform, you get 1 hertz, I got 2 as a magnitude, 2 hertz also has 2 as magnitude, 1 and 2 hertz, the phase is 0, it is what we would expect to see. Now if I apply this to a circuit, I will get current. I can do the FFT on the current. I will get similar to this; I will get 2 other graphs saying that this is the magnitude of current in two different cases. This is the corresponding phase value etc. So I can get the ratio of magnitude; I can subtract the phase; I can tell this is the impedance and this is the next impedance based on these 2 values, I can also tell the magnitude and the phase of the impedance at the same 1 second data acquisition time. Normally I will have to do 1 second here and a half second here, one and a half second, that is absolute minimum. We will forget the data processing time. But if I combine these waves and send this as one wave, I can get the impedance at 2 frequencies in a shorter time compared to doing it separately, as long as I can do the Fourier transform. If I send it on 1 hertz, 2 hertz, 3 hertz, 4 hertz, I will have to add at those times and say this is the data acquisition time. Now I can combine all those waves, send one, get the output, and do Fourier transform of the input as well as the output. We should know what the Fourier transform of the input is. However, we will anyway do this process, then compare, and get the results. You can get the results of impedance as a function of frequency. However, there are certain pitfalls for that.