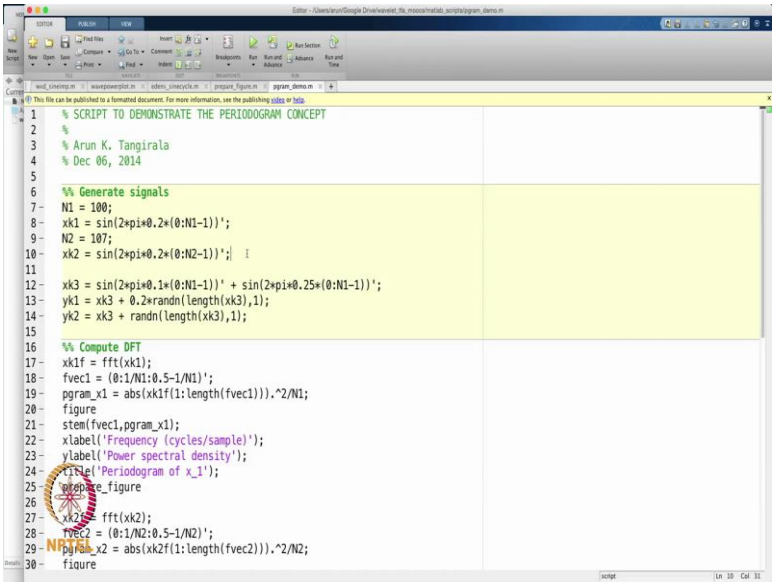
Introduction to Time-Frequency Analysis and Wavelet Transforms

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

Lecture – 3.6

Hello friends, this is a MATLAB demonstration of the concepts that we learnt in lecture 3.6, in particular the concept of periodogram. So, I am going to show you how to compute periodogram starting from scratch that is starting from FFT.

(Refer Slide Time: 00:40)



Although there exist a routine called periodogram in MATLAB that does for you, what I will do is I will show you how to do this by yourself and then you can compare the results of this with what periodogram gives you. I will briefly talk about what periodogram gives you, but I will not really walk through the routine and the utility, use of that routine in this session. So, this is an addendum to lecture 3.6.

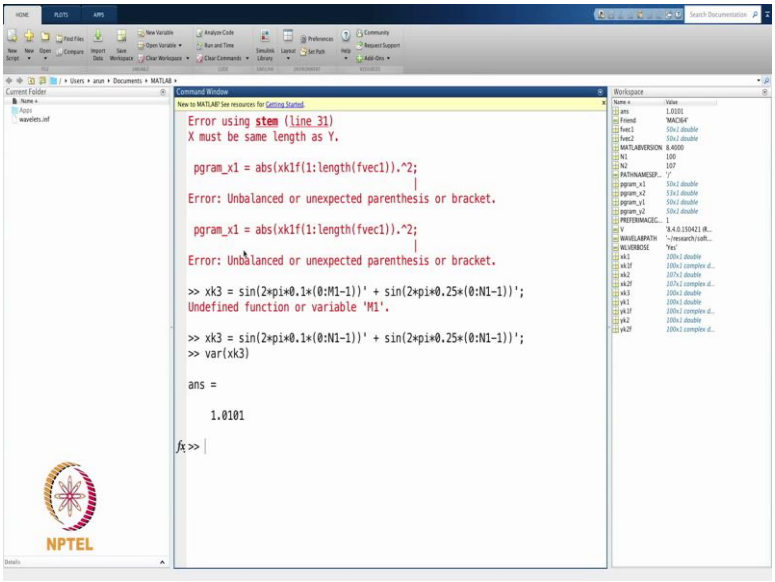
So, first what I would like to go over is the set of commands that are associated with generating the signals of interest. So, we are going to generate 4 different signals. The first one deterministic signals, the remaining two are noisy versions of the first deterministic signal that I am generating here, alright. In fact, the noisy version is slightly different; it contains 2 periodicities; so, sorry about that. So, let us look at these two deterministic signals.

What I have is, the first signal is going to be generated at a frequency of 0.2, and so, is a second signal as well. The only difference between these two deterministic signals is the length. I have 100 samples of in the first case, and 107 samples in the second case; this is the reason why I have chosen these numbers. So, these are fairly straight forward commands to generate a sine wave of this particular frequency 0.2 cycles per samples. So, this 0.2 refers to the discrete time frequency of the signal. As I said in both cases the frequency is 0.2.

Now, let us look at these two signals which are the noisy signals that I am generating. At first I generate the noise free signal which contains two frequencies of 0.1 cycles per sample and 0.25 cycles per sample. Again, there is a reason why I have chosen these 2 frequencies. And, they are going to be, I am going to generate 100 samples of this mixed sine wave. And, I generate 2 noisy versions of the deterministic signal. In the first case, I am going to add mild amounts of noise; in the second case, I am going to add large amounts of noise.

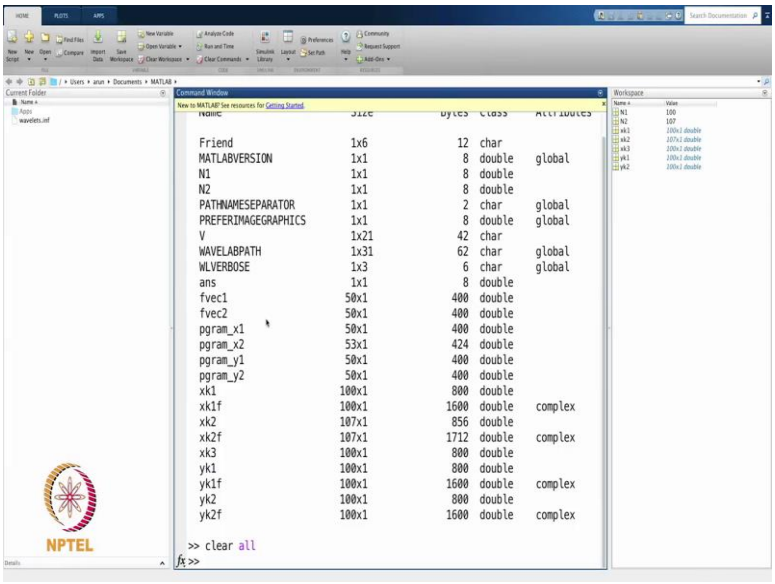
Now, the purpose of working with these 2 signals is to see how if I apply the concept of periodogram to noisy versions, what result comes out when I do that, because we have learnt periodogram purely for deterministic signals. So, let us generate the signals, as you know in MATLAB you can just divide your code into different modules. I am going to actually run this particular module itself or section itself; and, I have done that.

(Refer Slide Time: 03:27)



And, let me take you to MATLAB here.

(Refer Slide Time: 03:33)



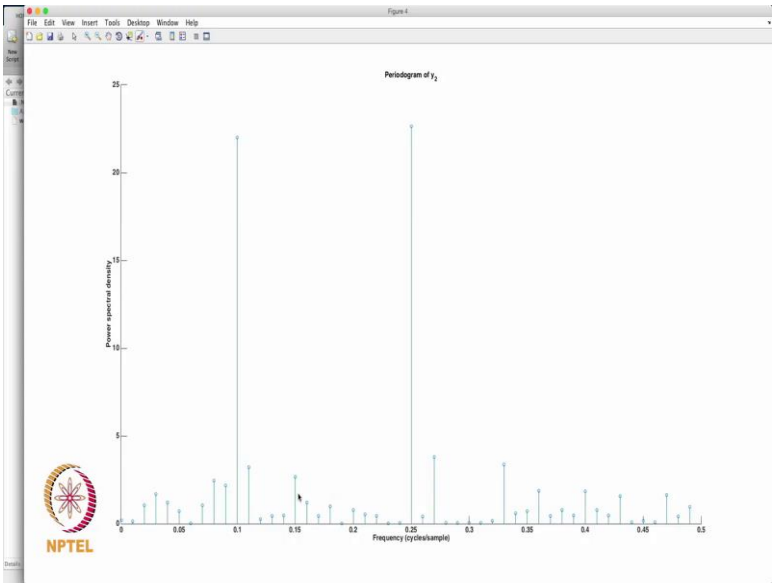
It clearly shows, if I ask the list of variables it shows that it has generated x 1. Let me clear everything for you and then rerun, so that things are very clear to you; these are generated. So now, I have the signals x 1, x 2, x 3, which is the noise free version of the noisy signals; y 1 and y 2 are the noisy versions of x 3. So, this fact is verified. Now, let us compute the Fourier transform or the DFT using the routine FFT, and then compute the periodogram.

So, FFT as I said is the routine that computes the DFT for you. You should look up the help on FFT to understand its syntax and the algorithm, and so on. So, I have computed the FFT of x k 1, x 1 here. Now, here, in the second line here, you should pay attention on line 18 in fact. I have, what I am doing here is, I am creating the set of frequency points as a result of computing this DFT via FFT. So, what I am doing is, the first frequency point is 0, and the second the interval is 1 over n 1; we have already seen that DFT computes with the frequency resolution of 1 over the length of the signal.

The last frequency in my set is 0.5 minus 1 over n ideally, this should have been 1 minus 1 over n, but because of the conjugate symmetry of the DFT, I only need to walk upto the half and look at the power spectrum. The power spectrum is going to be symmetric for, with respect to negative and positive frequencies. So, I am only walking upto the half, not exactly half, but 1 over n short of 0.5, and that is why I create the frequency vector this way.

Then I compute the periodogram that is fairly straight forward. I take the magnitudes square of the DFT coefficients, divide by n 1. And then, I plot a stem plot. Stem plot is a correct one to do because the DFT is only computed at the grip points. The rest is all decoration of the plots, and prepare underscore figure is a routine that I have written to decorate it further, annotate it further, and so on. So, likewise, for the second signal as well. It is the same story. Let us generate these 2 plots and see what I get. So, this is a second plot that I have.

(Refer Slide Time: 06:04)



Let us look at the first plot. Let us magnify this. Now, this is the periodogram of the first signal. Remember, the first signal has a frequency of 0.2, and I had generated 100 samples of that signal. The periodogram correctly detects the frequency component of the signal. It says, exactly at 0.2 there is a p l square the DFT is 0 or the periodogram, power spectral density is 0; or, you can say, the power is 0. So, that clearly tells me that there is a single frequency component in x 1 of frequency 0.2 cycles per sample. So, you have to be very clear of the units.

What about the other signal, x 2, which is also of the same frequency, but I generated 107 samples of it. Now, it says, that apart from, in fact, the first thing that you should observe is that it does not even correctly detect the frequency there, which is 0.2. It actually shows that there are many frequencies, of course, within the vicinity of 0.2. It does not hit 0.2. And, this phenomenon that you see here is called spectral leakage.

What has occurred here, keeping aside all the map, a simple way of understanding spectral leakage is, in the given data record the signal of interest has not completed integer number of cycles; it has completed fractional cycles. And therefore, not only you do not hit the exact frequency, but this spectrum at the frequency 0.2 is actually, it actually spreads around to neighboring frequencies; it leaks and that is called the spectral leakage.

So, this has occurred primarily because the data record does not contain a full integer number of cycles of the sine wave. Of course, in practice, what do I do? I do not know the frequency, so, how do I collect the record such that integer cycles are record then. In fact, if I know the frequency I do not even need to do a Fourier analysis. Well, in practice what happens is you will have to use some window functions which I will talk about in the case of short time, when we talk about short time Fourier transforms which will mitigate the spectral leakage. It will not completely eliminate it.

However, the good news is that if you collect large samples then does not matter really whether the signal has completed fractional cycles or integer cycles because the number of full cycles will be so large that the fractional, effect of the fractional cycle will be minimal. So, as n becomes larger and larger, and I leave that as an exercise to you; you can choose for example, 1007 samples and see how this spectral aspect periodogram looks like; you will see that the spectral leakage would come down. So, we will talk about spectral leakage more in, more formally when we talk of windows in short time fourier transforms.

Now, let us move on to the noisy versions of the signal. Again, the purpose is to show you how, when you apply the concept of periodogram to noisy signals things change; whether do they change or not, is what we will see. After this I will take an ideal random signal where there is no sine wave and compute the periodogram, and show you what it is.

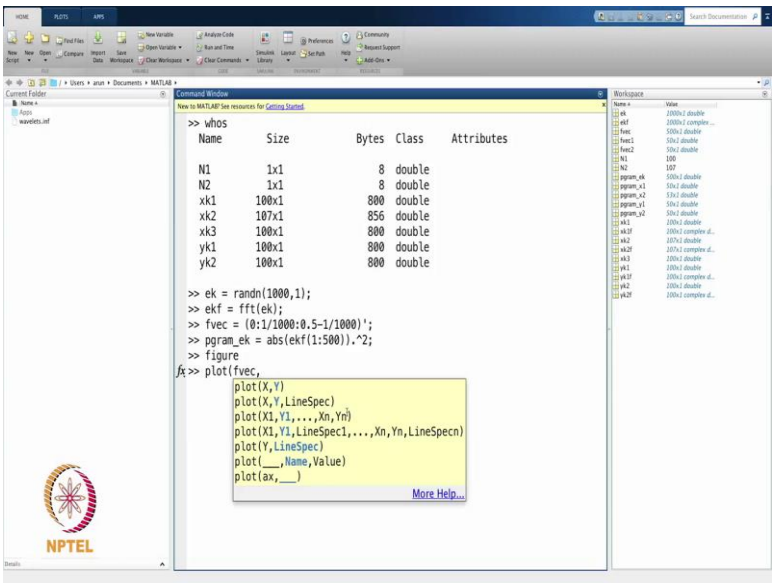
Let us look at the periodogram of the first noisy signal. Despite the addition of noise I am able to detect those 2 frequencies, 0.1 and 0.25. Here there is no spectral leakage because both these frequency components have completed integer number of cycles in the length of the signal. That is the reason I have chosen these 2 frequencies.

However, if you look at the periodogram of the second signal where we have added large amounts of noise; what we mean by large is the signal to noise ratio is 1 here, and you

should verify why it is 1. Earlier, the signal to noise ratio was 25. Now, still the periodogram is able to detect the frequencies quite vividly, but then there are other computing neighbours here which brings, which bring in a lot of ambiguity; you are not sure whether these frequency, these peaks here are due to noise or due to the signal itself. So, that is the problem of having large amounts of noise.

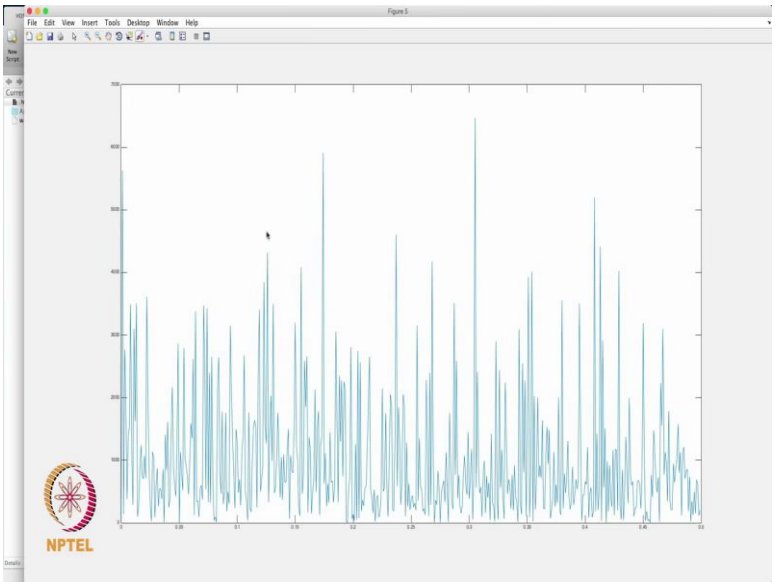
Of course, then there are algorithms which detect the levels of noise and then you can filter the noise from this data and rerun the periodogram, and so on. But, the purpose of this example was to show you that you can still use the periodogram to detect the frequencies when the signal to noise ratios are fairly high; ideally, you should not be using periodogram if the signal is purely noisy.

(Refer Slide Time: 10:38)



And, that is the last example that I am going to work for you here. So, let us generate here 1000 samples of a random signal, ideal random signal known as the white noise; its mean 0, variants 1. And, I am going to compute the fourier transform of this white noise realization, 1000 length realization. And, we will create here the corresponding frequency vector. So, what we are going to do here now is, we are going to evaluate the periodogram of the signal over the first half as we did earlier, and then obtain a stem plot. In fact, because you have 1000 observations it may be better to plot the regular plot itself.

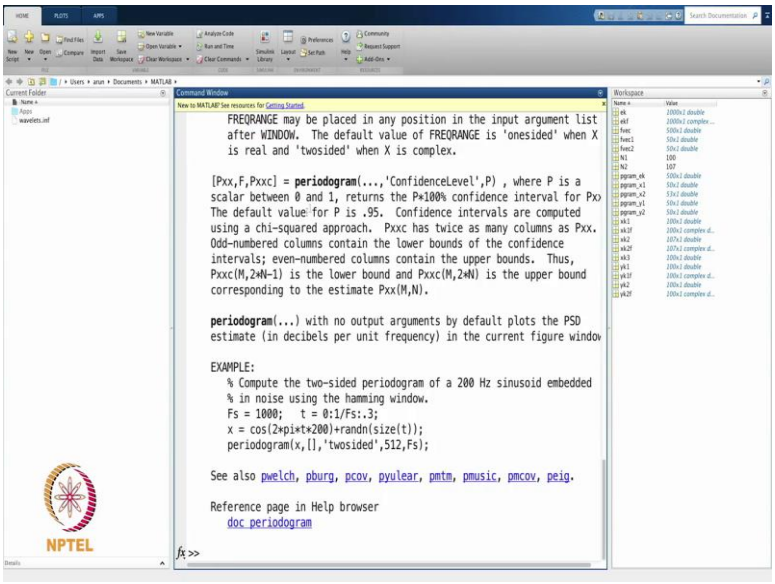
(Refer Slide Time: 11:38)



So, look at how it looks like. This is how the periodogram of a white noise looks like. Now, how do I know if this is correct or wrong? Well, theory says, that the spectral density of a white noise process should be flat; that is it should be a horizontal line running across frequencies, running over frequencies. But, what we have here is a very erratic display of the spectrum. There is no way I can see straight away that there is the underling theoretical one is a flat one. So, this is the problem of using periodogram on a purely noisy signal or a signal which has large amounts of noise.

Of course, there are improvements to this method known as modified periodogram methods and so on, for random signals. And, we are not going to discuss this further; just to give you a feel of when to use a periodogram and when to use modified versions of it. So, I would like quickly go over the documentation on periodogram; I am not going to show you how to use it, but I leave that as an excise to you. Verify the results that we have obtained manually; that is by manually computing the periodogram with the results that you get from the MATLAB routine.

(Refer Slide Time: 12:53)



So, let us just quickly walk through the documentation on periodogram. In fact, I am not sure how well you can actually see here. So, let me actually go through the help in the screen here. So, what does the periodogram do? Let me take you to the start of the help here; it returns the power spectral density, but it uses the, it returns the power spectral density in terms of angular frequency. So, that is the expression that you should be using.

What we have calculated is power spectral density in terms of cyclic frequency. So, when you are comparing, please be careful; compare the right things. And, there is an option called window which we have not discussed in detail, but I have mentioned this that windows can be used to mitigate the effects of spectral leakage; by default there is no windowing that is performed, if you do not specify the window.

Now, there is a something called here NFFT which is a number of frequencies at which you are evaluating. And, you should check what are the default values of the NFFT; each is of, each routine uses a different number for this. We have used NFFT equals the number of observations itself, but periodogram I used something else.

And finally, if you go down here, it can also return the power spectral density in terms of the cyclic frequency, but then you have to specify the sampling frequency and so on. What you could do is you could obtain the periodogram, the power spectral density in terms of angular frequencies, simply multiplied by 2 pi that will actually give you the periodogram in terms of cyclic frequency.

And, there is something called a 1 sided spectral density, 2 sided spectral density and a centered one. Now, this is the last point that I want to mention. The power spectral density is symmetric with respect to frequencies, therefore many a times what is reported is a 1 sided one. That is, what is done is, the power spectral density on the negative frequency is actually added to the positive one, except at a 0 frequency. And, that is called a 1 sided power spectral density; so that the area under 1 sided power spectral density still turns out to be the total power. So, the area is preserved.

Two sided is nothing is done that is the default power spectral density for both negative and positive frequencies are returned. And, center is where the p s d itself is written as a vector where the center frequency is 0. So, if you look at the vector of p s d as it returns, the center value corresponds to 0 frequency. So, these are the 3 different options. You should see what is the default option? I believe the default option is the 1 sided power spectral density, the periodogram returns. So, you have to be careful. We did not compute 1 sided power spectral density; we computed 2 sided power spectral density.

So, if you want to compare the results that we obtained with what periodogram is, then you should use NFFT equals number of observations; you should not window and you should ask for a 2 sided power spectral density, and you should multiply what you obtained with 2 pi. So, these are the 4 different things that you have to do to compare.

Thank you.