# Real-Time Digital Signal Processing

**Prof. Rathna G N**
**Department of Electrical Engineering**
**Indian Institute of Science - Bengaluru**

## Lecture - 38
## Application of Adaptive filter in MATLAB

Namaste welcome back to real time digital signal processing lab. So whatever we have studied in theory regarding the adaptive filter, we will be seeing the demo first in MATLAB. In the next class we will be seeing on the board using the Code Composer Studio.

**(Video Starts: 00:44)**

So today we will present the MATLAB. So, we will see the autocorrelation function with an example using MATLAB. So what is it? Number of samples is chosen as 1024 and $f_1$ is the frequency of sine wave what we will consider and $f_2$ is the other frequency component which is 200 hertz, $f_1$ is 1 hertz, which is the sampling frequency both of them and then we will first we have to generate the sine wave using the sine function in this case what we are using it.

So, as we have discussed in the sine generation, you can use any of the methods either you can use the table, you can use the IIR filter has an oscillatory function or you can use the sine function basically, in this case, we have taken sine function and then we can do the sub plot and then see how it is going to look like then this is the frequency what it is generated with sampling frequency of this thing, what is it 8 kilohertz.

So, one depicts that as the 1000 hertz and 200 will be 2 kilohertz what it is going to be generated. So then you will be calling the autocorrelation function in MATLAB Rxx, which is not nothing but x correlation of x basically, and then we will plot it and then we can now Rxx is getting plot and then we will see how it is going to look like. So, this is the function in MATLAB what will be running it, so we will check the thing.

So, it will be asking me from which folder it has to take a default it may be there, since I have opened it or you have to go to that particular directory and open the file. So, all the files are here as you can see, so it is running. Now, as you will be seeing the thing this is the sine wave which is correlated with the other sine wave. So, you will be seeing the peak in the centre and then which is going to be dying down in this case.

So this is how it works. And we will see that if we change the adder noise to it. So, we have generated that is between the correlation between the 2 sine waves. Now, if the sine wave is burried in random noise, what will be generating it? So that is x plus the noise added and see whether we can see in that particular noise, the correlation between whatever input what we want to take that. So again, we will be checking y is our noisy signal along with the original signal which has got embedded.

And then with the original signal we are trying to correlate and what will be our, we call this as the cross correlation. So, in this case, because we are taking the correlation between 2 different signals, so if it is with the same signal, we know that it is autocorrelation here it is between 2 signals. So we call it as the cross correlation basically, and we will see how the output is going to look like. So, this is the sine wave which is burried or noise signal what you are seeing that thing.

So this is y of n. So you are unable to make out the thing when I take the cross correlation. So you will be seeing that your signal is present. That is what the cross correlation is going to show in this case. So now what is the next one we will see cross correlation. So this is between sine wave and then cosine wave first example what we will be seeing it. So you will be seeing that these are the t and t2 are the 2 frequencies, what we will be taking it wave periods what you will be multiplying with and then sample period what you will take it.

Here over sample is 1024 in this case, and frequency is 1 kilohertz. And then you will be this is a sample period with the 2 by oversample what you will be taking it. So, you will be doing the generation of sine wave here and cosine wave and then you can see the correlation between sine wave, cos wave and then the sine wave, and then see how it is going to look like. So we will run the code and you will be seeing that this is what our code looks like, what we will do is one thing what I have to do is for you to look at the thing correctly.

So I will try to reduce the display settings so that you can view the thing correctly. So I will make it 100% and then only for the letters, we may have to have it a little bigger. So I can make it this is a little bigger for you to view the signal. So, you will be seeing that these are the 2 signals so when you do the subtraction of it, there will be something correlation coefficient is going to be more in this case.

So that is what we will look at it and then we will see for I will be passing sine wave, and then I will be passing the other one is the exponent. So how does it look like we will see it. So I will be commenting the cos wave, and then I am calling with the same name because I need not have to change at this place also, but this is now the exponential decaying signal. So minus 0.1 into t whatever t is getting generated here in the same way.

Then we can see it in t2, also, what is the thing is going to look like so we will run this again. So you are seeing that this is what you are, that is cross correlation between your exponentially decay signal and then the sine wave, when you add them out, you will be seeing that it becomes 0. So that shows that they are not correlated. So, as we discussed in the theory, if it is +1, it is positively correlated if it is negative, negatively correlated, or if it is 0, it is uncorrelated.

So this is what, what it will show. So we will see that with respect instead of t we will put it as t2 and see whether it is going to make any difference here, t2 was the sample period which was getting generated with respect to our cos wave signal so we will run the thing. So, you will be seeing that it is how the distances when you add both of them, you will be seeing that it is going to become 0 respective of whatever changes you have done.

So, this shows that how our correlation is going to look like some other functions are there. So, you will be seeing that how to generate if you see the thing this is generating 64 samples of this thing sine wave with frequency 1 kilohertz. So how you can run that? And then the other one will be how to generate your exponential signal. So, you will be seeing that this is what how the exponential looks like which we this thing run the cross correlation with this exponent and then the sine wave generated.

And then the other one, you will be seeing it is generating what is it? 64 with the this thing unit impulse signal how to generate it, what it is going to show some of the signal generation what you will be looking at it. So this is your unit response, so at 0 it is going to be 1 and rest of the places it is going to be zeros as you know how to generate it using MATLAB, these functions show. Now with this correlation, so what we will do is we will go to adaptive filter.

So we have already seen this adaptive filter in the last class. So what we will do is how to generate our LMS and then now an NLMS algorithm and then see the difference application what we discussed in the class we will look at it noise cancellation and other things and then how to generate our echo and then scrambler and equalizer with respect to MATLAB what will see it and then the same thing we can look at it in the Code Composer Studio also.

And now we will be seeing these are the assignments done by as I have been mentioning by different students, so how they do that and then how your code also can be what we will look at it. So, this will be m file what it is going to run it, I will be closing because we have finished the thing. So, we can open this so here it is doing a GUI so LMS algorithm demo what we have seen that thing. So we will see NLMS and then RLS those who are interested can look into the thing and then see that how it is going to work, want to work.

Because it is a very complex equations one has to write and then it will take time for most of the applications either will be running LMS algorithm which has faster because we know that time consumption for inversion of a matrix and other things will take longer time in our hardware. So we will be trying to avoid so, you will be seeing that this is what the how the creation of our LMS algorithm. So, the order is defined with 20 if you want to change the order you can do the thing.

And then the weight functions are initially set to 0 and then mu is set to point 006. So, we have seen that varying the mu how it is going to get affected. So you will be seeing that up to the length of x minus order. So you will be putting it to the buffer so our y out of i is going to be, what I will do is now before running the thing, I can increase the font size so that you would be able to see them properly. I will make it as 125 and then when I am running the thing, if the graphs are coming, then we might have to reduce the thing.

So, now, you will be seeing the length of x minus the order of your filter, you will be putting that what is it buffer is initialized and then y out of i is buffer of into your y into your this thing wave function w into 2 and then error function is calculated of yi minus buffer into w and this is the weight function which is getting updated w + buffer dot star mu into our error of i. So update the weights and then we will be computing it.

So, the next one is NLMS algorithm so, here we will be defining alpha = 1 we said it is equal into LMS algorithm. So here alpha is chosen as 0.005 and then your constant what you will be calling it as 0.7 so, you will be doing the same thing with your mu is calculated on the go as you can see it here it is 2 times alpha divided by because we said some small constant what we have to assume so that divided by 0 is not going to happen.

And then you will be doing the buffer squared in this case and then your weight function is going to be updated. So as you can see that this division also has to happen which is a costly affair as we; have seen in the architecture class. So, multiplication is easier to do it we saw Braun multiplier, but division successive subtraction what we have to do it which is costly. So, that is the reason why most of the cases will be assuming mu and then we will be running the LMS algorithm.

In the worst case, if it is not okay then we may have to go for this. So, the other one is RLS algorithm you will be seeing that your this thing will be calling it as a random function and then you have to length of the w is going to be 0 and then some constant what you will providing in this case gamma is one of the constants one has to use it, which is almost nearer to 1 what is shown in the thing.

And then the equation you will be seeing that how we are calculating your temporary value and then error function what you are calculating and then alpha is going to be calculated in this case using gamma plus our buffer basically, and then G is the constant p into buffer divided by alpha what we will take and then the weight is updated based on these constants basically G value.

And then you will be calculating the; that is p value power of the thing by gamma and then your y out is n is going to be your y temporary what you have calculated here earlier. And this is how you will be updating your this thing, what is it mu value, which is going to be calculated and then as you will be seeing that a lot of division and then multiplication are involved in the RLS algorithm.

So we will see, but it is much more precise than the both of the algorithm. So we will run the algorithm as you will be, I was able to see the thing, otherwise I had to reduce my size of the what I will put it as screen to in any case, to have a better clarity, what we will do is put it in the 100% our display size so that you will be able to see them clearly. So as you can see here, I will make maximise, so I will be loading the noisy signal.

So, how to select the thing this is the noisy signal what I have it and then I had to load the reference signal both has to be fed into our algorithm. So this is the desired signal. So now, we have run it already in the previous class LMS algorithm. But to have the better understanding of it, I will run the LMS algorithm and we are seeing that this is the noise added signal what it is shown this is the original signal, when you do the LMS algorithm we are getting back.

So we will play the thing it is the same speech signal what we will be doing it. So you will be seeing with the noise. Here also single tone as we demonstrated in the filters, same thing has been used for our adaptive algorithm. So you will be hearing the output. So you can see initially there was a dip so we will run the same thing with NLMS algorithm.

So you have seen that the thing is little bit shifted and we will hear it how it is different from the LMS algorithm. So now we will run the RLS algorithm, which is much more precise, as you will be seeing that this is your input signal and this is with the noise and you are seeing here output almost your complete noise is removed so we will play this.

So one has to pay for computation, if you want to have much more clarity, so we will get back to our 125% so that we can see the thing and then when we are running the code will go back again to our 100%. So, this shows our NLMS and this thing algorithm. So, what we will do is again, as I said, different students do different way here everything put together. So, the other students what they do is some of them, they do it in a different way it is directly LMS algorithm here it is going to run.

So, that is what is it you will be algorithm remains the same thing your order of the filter may vary and what tone you will be taking it is going to be different. So, this is going to start from 120 this thing x and y arrays basically. So, what you are storing it. So, this will be your LMS algorithm, i equal to L you will be starting from l value down to 2 actually that is why minus 1. So, you will be updating your x i in the reverse direction. So, this consumes less time when you are doing from the other end to this end.

Basically circular convolution should happen so, that is what, what is implemented here. So, you will be seeing that x l filtered through the thing. So, that is up to the filter length, you are calculating your y x of i into your w of i because n - i what it is the thing what it is taking x of i minus n you can take it and then you will be multiplying with w of i and then this is your error function, which is given by that is desired signal big of n - y you will be knowing even you may name it in a different way and then update of w is going to happen.

So, that is mu error is calculated as mu into e n error and you will be updating the wi you may be wondering why this has been done earlier why not? In the other student what he had done was it is inside this loop I will be updating my error and then I will be using it so, can you guess what is the thing is going to happen? Because I am putting in the loop every time this constant has to be multiplied. Otherwise I can there are going to be how many multiplications along with it 2 more multiplications I have to provide here I have reduced it by 1.

So that which is pre computed this is not going to change according in the loop and then you can multiply and get the result. So you will be getting your error basically shown this way so let me run the thing and then. So one of the thing is what is it? This has not been calculated so what we will do is I have to hopefully I will be able to uncomment it and then we will save it and then try to run the thing.

Still it is giving an error in the, can you guess what is the thing? It is unable to get voice and then tone dot wav file actually I have missed to put the thing. What we will do is I can bring the thing from the other one and then put it here both have been I will put the copy it here and then we will see and then change the name with this name it is noisy sorry you should use capital save it and then I can run. There is this thing because it is you will be seeing that index in position 2 exceeds array bounds this should not exceed what it says.

So, the problem with this is because it is a different signal what I have taken because they would have set it to their requirement as you will be seeing it I had to get this file here and then run the code. So that, we can see it in the hopefully I have the; that thing it has not been copied for the thing, I will get that signal. So, that otherwise we have to whatever error is coming, what it shows is it is 2 channels what it has to take it.

So, here it is now having only 1 channel, so unable to read this file. So, when they have combined the 2 thing together, we have to read them. So, what I will do either we have to correct this or because it may give some more error in the thing the way they have implemented it, because it is 2 channel what it has taken the thing this is they are putting it as a first channel and this is going to take from the second channel as you know it is a stereo input.

So, stereo input the first channel what you can input is your desired signal in the second this thing that is left channel will be desired signal right channel you can put the noise basically so that you are depicting whatever we discussed in the class that is the signals are coming from 2 sources one is the desired signal the other one is your what is it? Noise separately captured from 2 places which has been combined and then put it as audio basically in.

So this has to be merged using the MATLAB code and then we must be taking it in here it is only 1 channel what it present that is why it is giving me error. So thank you for listening in the next class we will take up the scrambling and then what I will put it is an echo generation reverberation and then scrambler together. Thank you

**(Video Ends: 28:25)**