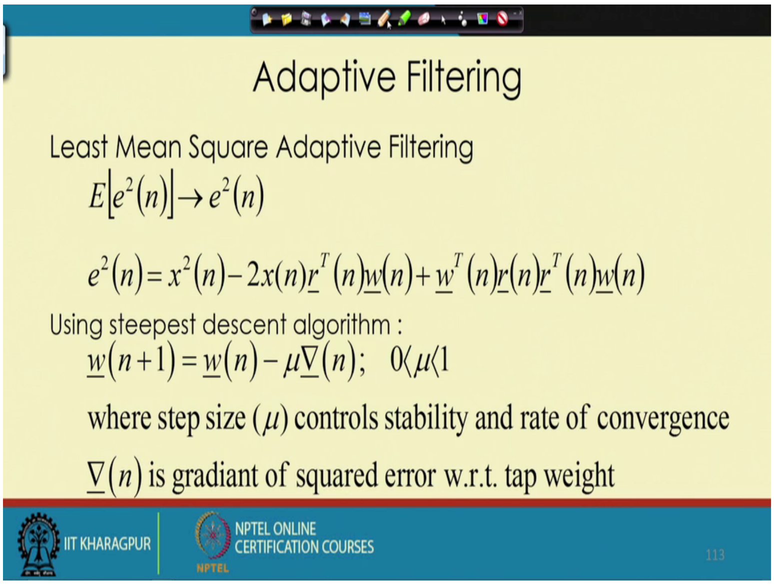
Biomedical Signal Processing
Prof. Sudipta Mukhopadhyay
Department of Electrical and Electronics Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture - 15
Artifact Removal (Contd.)

First, we look at the least mean square Adaptive Filter or LMS filter. Now the change what we do looking at that bag that optimal filter.

(Refer Slide Time: 00:34)

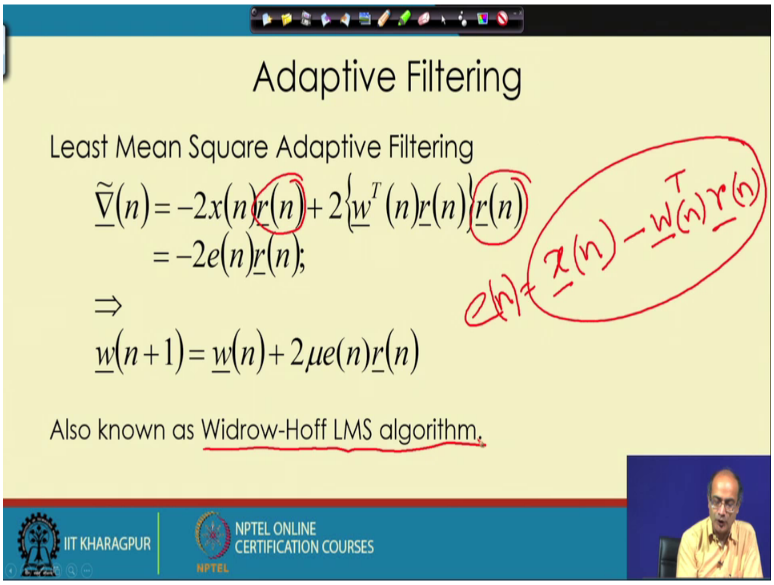


In case of optimal filter, we have actually taken that the expected value of e n square that we computed by the assumption of stationary city, rationality and the ergodicity of the signal. And we have replaced the expected value of the square of error as that sum of the error over the time. Now, as you know that thing cannot be done, because we are dealing with non stationary signal, so we cannot have something called ergodicity. The simplest thing what we could do we could drop the expectation operator and directly compute e square n.

Now, by dropping that expectation operator it gives us a huge advantage in terms of computation. Now let us see how it proceeds. So, we expand the square n we know that e n is that x n minus that the weighted the reference signal. So, that that part that when you expand we get this three terms, one is x n square and that we get cross between that x n and the that the y n, that is generated out of the reference signal and we get it terms square term out of the that reference signal here.

So, now using steepest descent technique, what we need to do the; at the next instant the weight w n plus 1 it should be w n minus mu times the gradient. And again because the process is changing over the time the gradient is also having an index of n. So, here the mu is the step size which actually controls the stability and the rate at which it will converge to the optimal value and this delta n it is a gradient of the squared error with respect to the tap weight, because tap weight is the only thing in our hand, that we can modify to get a better estimate of the noise.

(Refer Slide Time: 03:31)

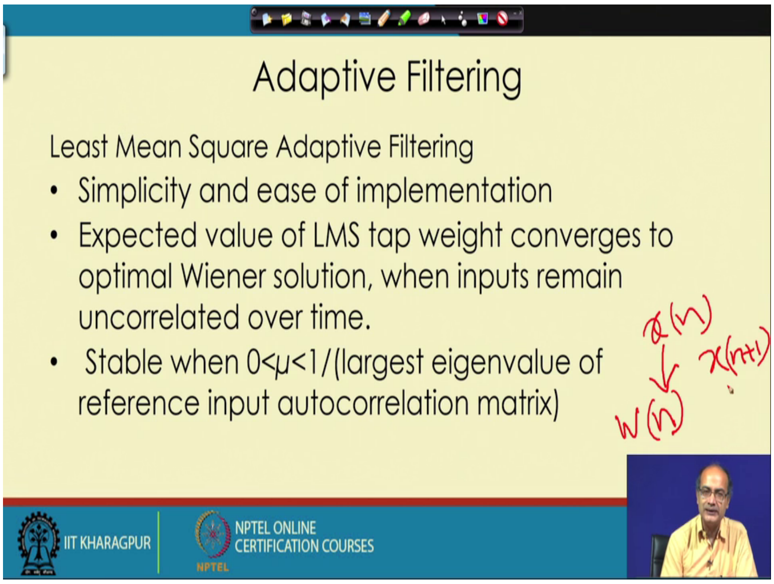


So, now that we compute the gradient of the that error with respect to the tap weight and we get it is minus 2 x n into vector r n and twice the that the dot product of the weight at the instant n, and the reference input r both of vector the dot product of these vectors into that the vector r n. Now if you take these part, these part is common out of these two what we get we get x n minus w T n into r n. So, these form you already know this is very known form this is our error, error at instinct n. So, replacing these part in this equation we simplify, it we get that the gradient is minus 2 e n into vector r n.

So, now if we replace that estimate of the gradient of the that error, we get here that the new tap weight at the instant n plus 1 it would be the previous tap weight and the minus sign along multiplied with this minus it gives us a plus, twice mu that the error at this instant that is e n into the vector r n, which is the reference signal vector. Now these equation is our update equation and this is known as Widrow Holf LMS algorithm, this simple equation actually we can use to update the weights. And as we keep on to update the weights it will actually keep on changing and (Refer Time: 06:05) to that the process and if the process is changing because the signals and nonstationary.

So, it will actually respond to that change and by that updation of the weights, it will always try to remain to be a optimal filter. So, that is the simple idea about that the Widrow Hoff LMS algorithm which is very simple yet very effective.

(Refer Slide Time: 06:42)

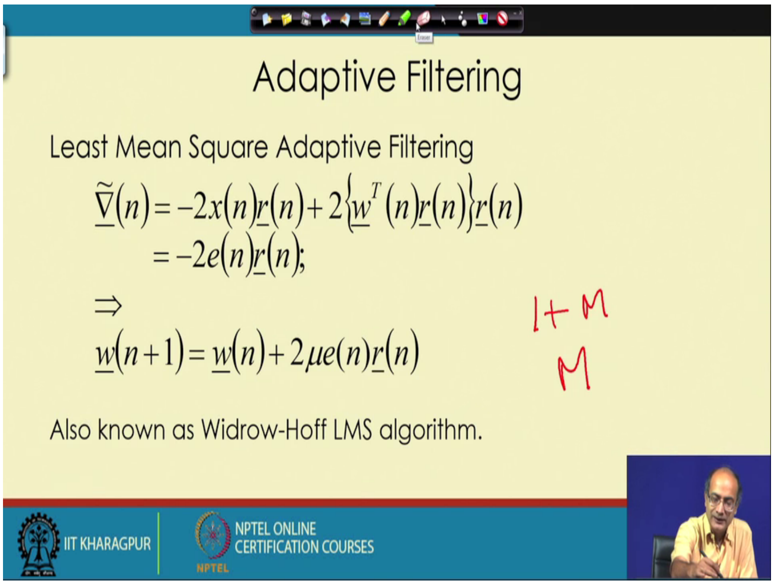


And that is the reason it is very popular in the academy as well as in the industry.

So, now let us look at the special the attractions of the LMS algorithm, the fasting what we get its simplicity and ease of implementation why we talked about the ease of implementation? The reason is let us look at that signal, that what we get the previous the page that update equation is so small and simple. What we need to compute we need to compute that error. In fact, that is the output so in any case we need to compute that and the reference vector of the that the reference vector, for that interfering noise that already we have taken to compute that actually that output of the adaptive filter.

So, if we take the product of these two and scale it by two mu, that gives us the that the change in weight. So, it needs actually that here one multiplication two mu is, one scalar quantity then another term is e n and with that that we are computing change actually multiplying with r n that is having dimension n into 1.

(Refer Slide Time: 08:51)



So, that the number of multiplication which needs one to compute 2 mu in two e n and then m minus 1 rather we can say that m multiplications, that because r n is m dimensional vector and then what we need that m additions because each of the component of the that tap weight it need to be modified.

So, 1 plus m multiplication and m additions so, they are actually very much lightweight and that weight the implementation is very easy and simple. Now why we are so much actually worried about the simplicity and ease of implementation? The reason is in case of adaptive filter what we have proposed that we are going to update w n. So, at each instant that before the new value of say x n comes; that means, once we get the x n we need to compute that w n, before the arrival of n plus x n plus 1.

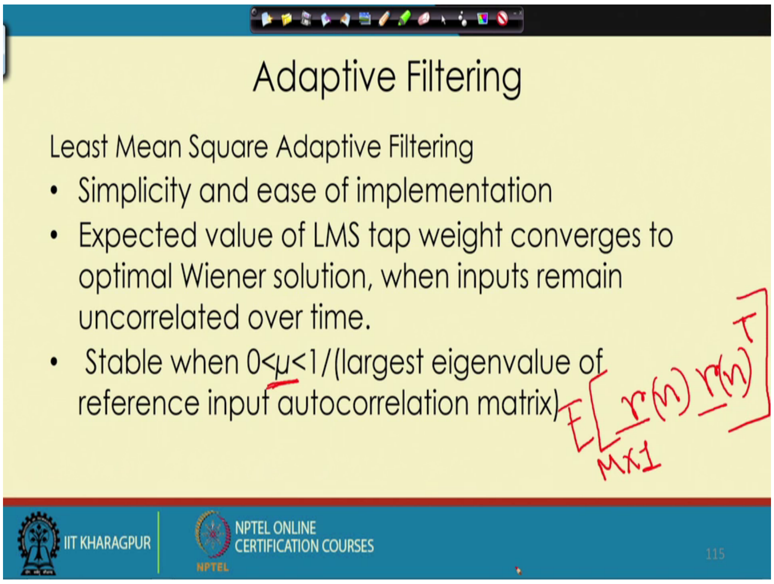
So, we have actually that one sampling period that is the maximum time to compute the weight, now let us look at the signal if the signal is having a frequency maximum frequency is one hertz, then we have one second time. If it is 1 kilo hertz then we have one mille second time. So, you see that more and more the sampling frequency that our competition also need to be more and more fast and there it becomes very important that the algorithm what we choose it is something that which we can implement in real time.

So, that is actually one of the very big bonus for the withdrawal of LMS algorithm. Now let us look at the that the next the point, the expected value of the LMS tap weights converges to the optimal wiener solution, when the inputs remains uncorrelated over time. That means, what it is suggesting that if we have the input that is something which becomes stationary at least over sometime it is not changing with the time, then we could have used the optimal wiener filter.

Now these LMS algorithm will act as a that optimal wiener filter, in the sense that we will get the same set of actually the tap weights or the tap weights will converts to that ideal tap weights given by the optimal wiener filter. The only condition what we to be maintained that the input that is the desired signal and the additive interfering signal, they should remain uncorrelated with each other over the time. So, this is the only condition on that signals, it does not demand that they need to be stationary or any other mode condition.

Now the next important input that we need to choose actually this mu what is a that that is called the tap weight very carefully, it should be bigger than 0, because if you take is 0 then there will be no updation of the weight. And again we told that it should be less than one in the actually that it can be shown that it should be less than 1 by the 1 divided by the largest eigenvalue of the reference input autocorrelation matrix.

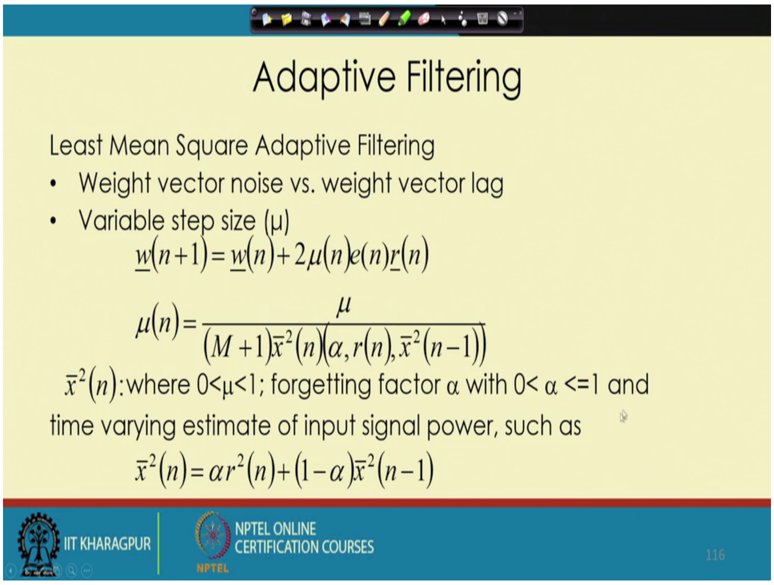
(Refer Slide Time: 13:58)



The reference input r n that with the help of that its say m by 1 vector. So, we can get the autocorrelation matrix of it, if we take the cross with the same vector or multiply with 1 by m vector that is transport or transpose form of rn vector. So, it can give us the autocorrelation and to get that autocorrelation, we need to take the expectation of this and if we compute the eigenvalue of this matrix that mu should be less than 1 by that the largest eigenvalue of this matrix. And that is necessary for the stability of that LMS algorithm otherwise what can happen this LMS algorithm will never converge.

So, we need to choose mu very carefully and in practice that as you may not be able to compute the eigenvalues. Usually, it is taken a very small value and we try to keep it small so that the stability does not come under any question.

(Refer Slide Time: 15:35)



Now, let us see that what could be the impact of choosing the mu. That we get actually two scenarios one is weight vector noise another we call weight vector lag. Let us start with the second one the weight vector lag. We told that we should be very careful in choosing the mu and it should be very small so that the stability does not come into question. So, if we choose mu to be very small then what will happen the updation of the rate of updation also to be very small because whatever the change was suggesting we are reducing it by mu by multiplication.

So, we are taking that part of the change suggested. So, if you take mu to be very small value, as the updates will be small if the process is changing, will see the compensation is lagging behind and the optimal that filter that is our LMS algorithm is taking time to adjust to the situation and the area will actually come down after a lot of time. Whenever there would be a change in the process that LMS algorithm will change, but because of its actually lack of speed in change it will lag behind, and because of that till it can catch up there would be high error. So, that is something unwanted.

On the other hand if we increase the value of mu, what will happen the updation would be faster, but in that case that when we go near the convergence and let us assume that we have chosen it in such a way, that it does not cause any problem in the that stability of that filter. But as we have increase that mu the updation is happening all the time even after the convergence. So, there will always be a change in the error and after the convergence we see that the noise is more.

So, that is called actually that situation is called weight vector noise; that means, there is a noisy estimate we are taking for the weight vector, and that is getting reflected in the output as noise. And the main source of it is that that the beginning of the LMS algorithm that there was an expectation operator on the error we have actually drop that and this is the result of actually removing that the expectation operator or that smoothing operation there.

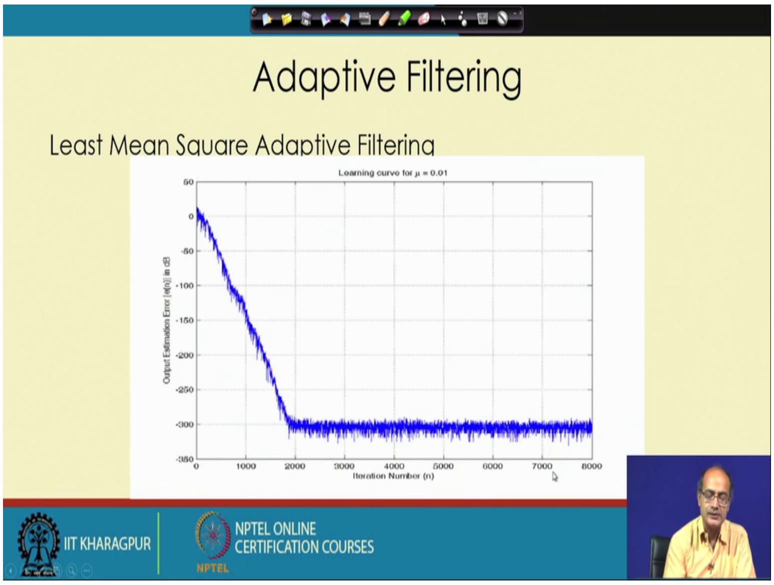
Now, what could be the ideal situation then? You would like actually to have mu to change over time initially if the mu is little high then we can update it quickly, and we can have actually low weight vector length and ones it converge then if you can reduce it then we can get actually that less weight vector noise. So, both we can minimise usually for a if you keep it a foxed value then we need to get one of them we cannot minimise both at the same time, but if we can update the value of mu change the value of the mu appropriately; that means, at the beginning if we can have little high value, then we can reduce the weight vector lag and then once it goes to the convergence if we you reduce it we can again reduce the weight vector noise.

So, with that expectation we come with a new update equation where the mu is changing over the time and how it is changing the constant mu divided by m plus one you know the m is the dimension of the that vector that tap weight vector and we have a function x bar square and it is actually a function of different thing there is something called alpha a forgetting factor then we have the reference the signal y n sorry r n and we have the previous value of that x bar square and how it works let us look at that x bar square at the instant n it is nothing, but that the alpha the part what we have taken as the forgetting factor it is varies between 0 to 1.

It is alpha times the instantaneous energy of the reference signal and one minus alpha times the previous estimate of that x, but square. So, if we start with the situation where r n was0before the start. So, in that case what is happening, if we make alpha equal to one it will change directly with the that reference signal because there is will be no contribution of the previous value of the x bar square. And if we make alpha equal to 0 then that there would not be any updation because there is no contribution of r n square, it will go only with the previous value of the x bar square.

Now if you choose in between a value which would be the case. It will take some part of the reference input as well as it is actually using the part of the previous estimate. That means, it will smooth out the change in the energy instantaneous energy of the reference, and with the help of that we are updating the mu. And thereby, we are able to actually optimise both the thing, that we would like to have when the system is changing less weight vector lag and when it goes was the convergence that less weight vector noise.

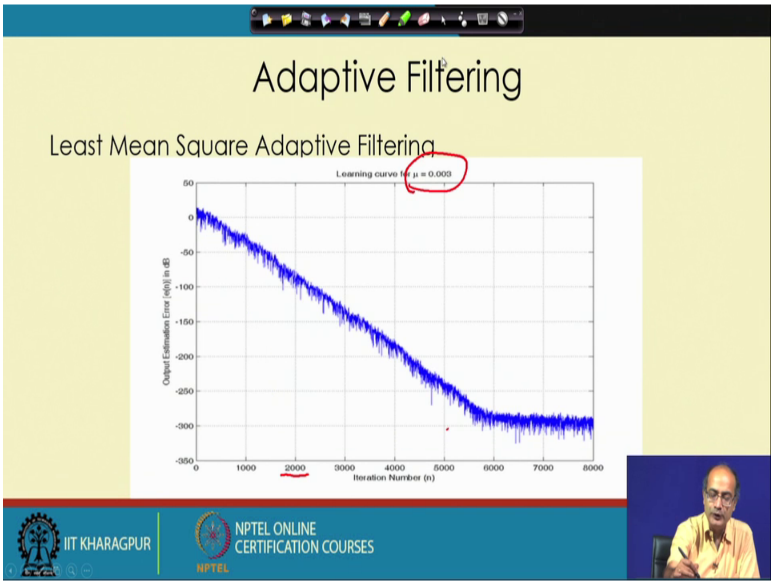
(Refer Slide Time: 23:40)



So, let us see some actually graphs which will help us to understand this concept. So, first we look at the scenario that we have taken in LMS filter, where the value of mu is not that 1. So, 0.01 for these value we get for this process that as it is moving that there is a noise all the time in the error the error is fluctuating, but it is converging steadily and before 2000 iterations it has converge.

So, roughly we can tell that within 2000 iterations it has converge and after that we see that minus 300 dB has become that noise power. And there is a small amount of fluctuations is there which is that weight vector noise and the lag in getting to that position that is a 2000 iterations that is the weight vector lag.

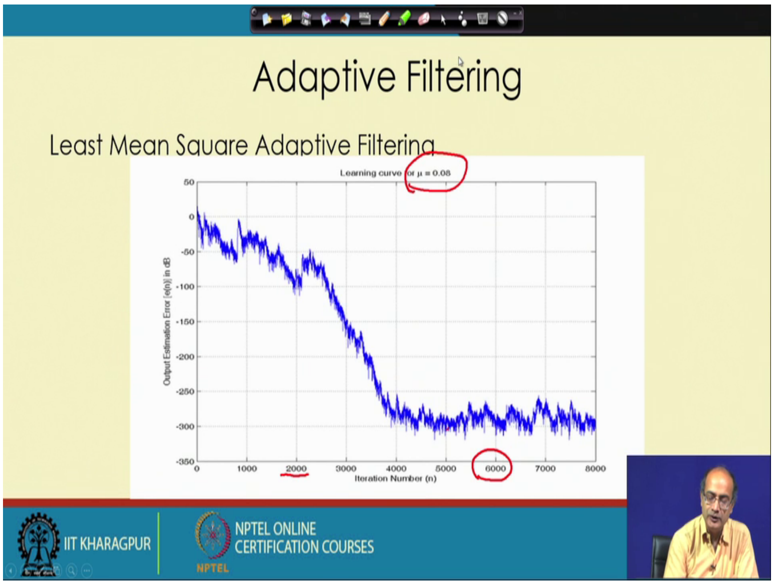
(Refer Slide Time: 24:55)



Now, if you change this value that if we actually make it even smaller, that point naught naught 3. So, what is happening in that case instead of 2000 now it is taking almost 6000 iterations to converge, and it is again converging to the same value minus 3000 and again there is some amount of noise which was there at the time of adaptation as well as after the convergence. So, what we get the weight vector lag has increased by reducing the value of mu even smaller.

So, we know that what we get the message here that if you keep on reducing the mu it may not actually help us in a new way.

(Refer Slide Time: 26:02)



Now let us look at the other scenario if we try to reduce the weight vector actually lag and increase the value from 0.01 to 0.8. So, in that case what is happening see that it is not in the previous cases if we look at that the previous cases that compare to that see in the previous two example it was directly going towards the convergence the character was smooth.

Now, in this case the trajectory is no more smooth and it is a jerky kind of movement, it is going down again going up. So, moving away from the that minima again it is coming down again going up and because of that nature it is taking more time though it is not going up to 6000, but it is not actually that coming down by 2000 it is requiring more time it is coming towards the convergence at 4000 iterations.

So, weight vector lag in this case has not reduced and associated with that we see that the error after the convergence that is also have increased you see there is a movement above the minus three hundred degree line over all error. If you look at here the changes are more here. So, we see that by increasing the mu we are having more weight vector noise. So, that these three examples that helps to get the importance of the choice of mu and that if we can get that mu in a proper way then we can actually have much faster response and lower noise.

Thank you.