Introduction To Adaptive Signal Processing

Prof. Mrityunjoy Chakraborty

Department of Electronics and Electrical Communication Engineering

Indian Institute of Technology, Kharagpur

Lecture No # 35

Formulation of the RLS Algorithm

So, in the last class we derived the weight of the equation for the affine projection algorithm APA. All these symbols were defined earlier. So, I am not going to define them again, right. There is an up posterior error vector all those were discussed last time. Now, essentially this is assumed to be invertible this matrix. This is a positive semi definite matrix you have seen earlier always, but we are assuming it to be positive definite that happens only if the columns of Xn they are linearly independent that is there is no linear relation amongst them which is a good you know assumption which is a practical assumption because the data is coming purely randomly you know.

$$\underline{w}(n+1) = \underline{w}(n) + \underline{X}(n)\left(\underline{X}^t(n)\underline{X}(n)\right)^{-1}\underline{e}(n)$$

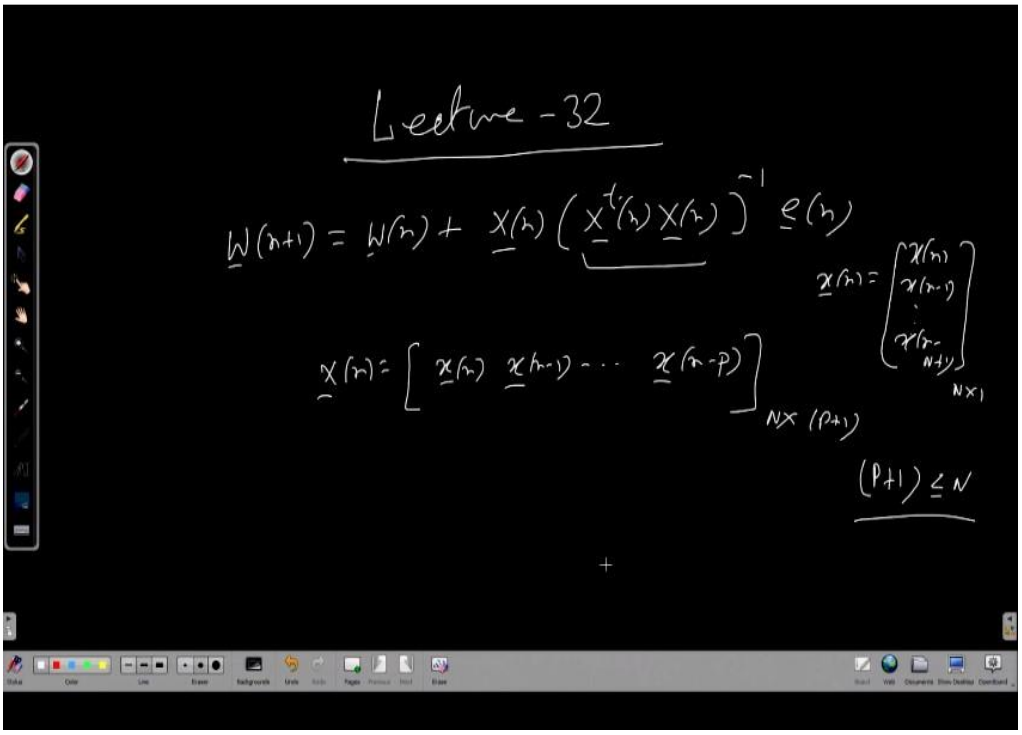
So, there is no linear equation governing the sequence of data because they are coming very independently you know randomly. So, no such linear relation governing them. So, as a result there is no linear relation involving the columns of Xn that is very practical and very very you know I mean valid assumption. In that case we have seen that X transpose Xn is positive definite and therefore, invertible, but sometimes that may not be the case like what was Xn? Xn matrix was Xn vector n minus 1 vector n minus p and we have we know any Xn is starting from that data at nth sample then n minus 1 dot dot dot dot n minus capital N plus 1.

$$\underline{X}(n) = [\underline{x}(n) \quad \underline{x}(n-1) \quad \cdots \quad \underline{x}(n-p)]$$

$$\underline{x}(n) = \begin{bmatrix} x(n) \\ x(n-1) \\ \vdots \\ x(n-N+1) \end{bmatrix}$$

So, total number of filter coefficients is capital N then W0 Xn W1 Xn minus 1 dot dot dot. So, these are n cross how many columns n minus 0 n minus 1 up to n minus p. So, p plus 1 we also have seen that if number of columns if I mean p plus 1 not if number of column cannot exceed the number of rows. Then again some columns you can write as a linear combination of the other columns this we have discussed. So, this should be less than equal to n number of columns should be less than equal to number of rows because every vector is length capital N.

So, there in a space of vectors length capital N that space has dimension capital N. So, you can have maximum n number of capital N number of basis vectors that is the Xs. If you have more column vector they should be I you should be able to write them as a linear combination of those basis vectors. In that case Xn will have linear dependence some linear relation involved with the columns and therefore, X transpose and Xn will not be positive definite all these were discussed at length last time and also in the initial part of the lecture.



But consider suppose n equal to 0 we are standing at the very initial point and also given Xn 0 for all n, n less than 0 that is we switch on the system we start the business at n equal to 0.

So, before n equal to 0 all that are 0. So, now, they started n equal to 0 then we will have Xn what will be the matrix it will be Z first Xn vector Xn vector will be n is 0. So, X of 0 which may be a non 0 data then X of 0 minus 1 X minus 1 that is 0 because all past values are 0 before time equal to 0. So, 0 0 dot dot 0 then next is Xn minus 1 if n is 0 it is X minus 1 column. So, X minus 1 minus 2 minus 3 all are 0s and by the same process all X if n is 2 n is 0 n minus 2 next column.

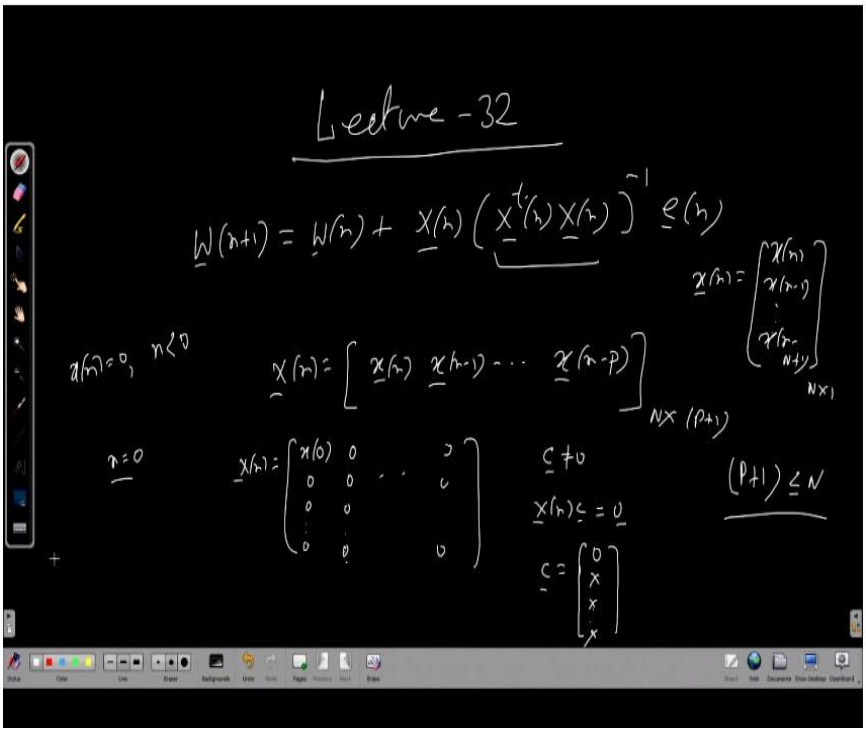
So, that will be 0 minus 2 there is X minus 2 minus 1 minus 2 minus 3 there is n is minus I mean Xn minus 2 the next column. So, n is 0 here. So, that column starts at n minus 2 there is minus 2 X within bracket minus 2 vector X within bracket minus 2 vectors. So, it will be X minus 2 minus 3 minus 4 like that all are 0. So, it will be 0s and so on and so forth.

Here the columns they are linearly dependent because you can easily see I can find a non 0 I can find a non 0 vector C. So, that if I combine so that Xn into C will still be 0.

$$\underline{X}(n)\underline{C} = \underline{0}$$

How that is if C suppose you take as first guy 0, but all other guys are non 0s you know I am putting cross cross means some non 0 values. So, that C is a non 0 vector because 0 vector means all the elements must be 0 here, I am taking only the top guy to be 0 all other guys are non 0. But if you do Xn into C with this C what will happen this 0 times first columns that will make it 0 then a non 0 times second column by second column is already 0.

So, it will again be 0 then another non 0 times third column third column is again 0 column vector 0 vector. So, that into any non 0 element still will be 0 and then so on and so forth when you add them you will get 0. So, here we say Xn is rank deficient columns are linearly dependent there is a linear relation that is you can find some non 0 vector C. So, that if I combine the elements of the column combine the columns by the elements of this C vector you can still make it 0 and therefore, this will not be positive definite because for it to be positive definite for any non 0 vector C Xn into C should not be 0. For any non 0 vector C you know for positive definite this we have derived this Xn into C must be a non 0 vector, but that is not happening here I can cleverly choose C like this.

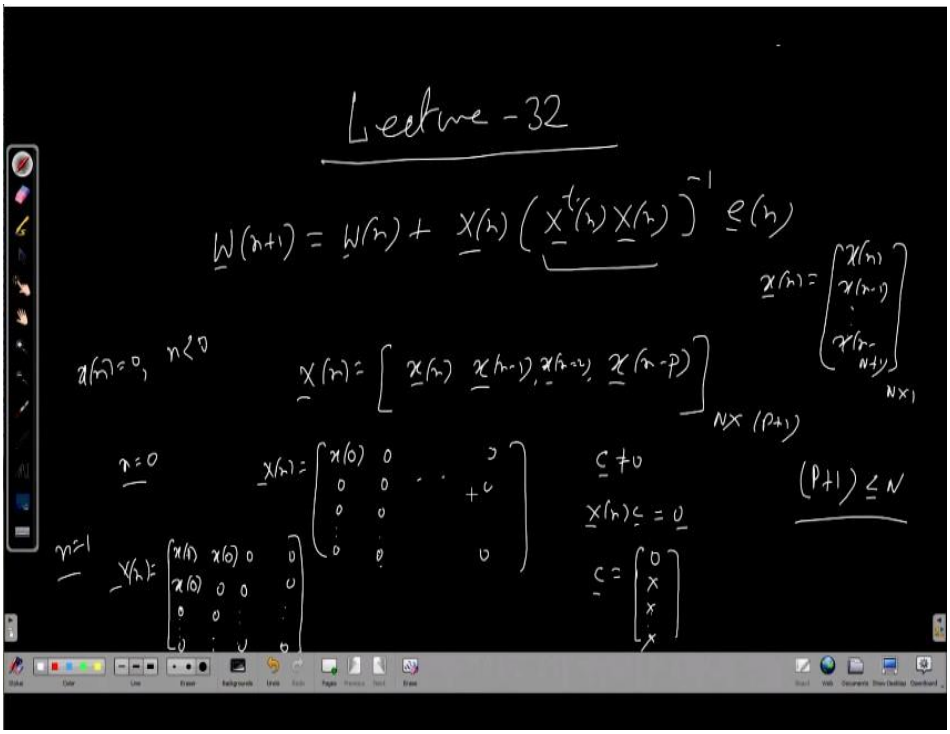


So, that Xn into C is a 0 vector. So, it is not positive definite same thing we will continue if you add now n equal to 1 that n equal to 1 then Xn will be first is X1 vector means X of 1 X of 0 and then 0. So, they may be non 0 X of 1 X of 1 X of 0 then X minus 1 minus 2 minus 3 they are all 0s. Next column will be Xn minus 1 n is taken to be 1 Xn minus 1. So, n is taken to be 1 now say X0 vector.

So, this fellow all right X0 and 0s then the X1 which is Xn minus 2 if you put n equal to 1 it is X minus 1 vector X minus 1 means X minus 1 minus 2 minus all are 0s and we will continue. So, here I should choose C to be 0 0 and then any non 0 elements. So, one 0 will take care of this first column 0 times this is 0 another 0 will take care of the second column 0 times this is 0 and non 0 elements will multiply this columns which are already 0 columns. So, we will get 0s 0 columns when added Xn will be C Xn C will be 0 all right. So, again Xn transpose Xn will not be positive definite and Xn will be rank deficient and this will continue for some time, but you see next time it will be X1 sorry X2 X1 X0 then X1 X0 then X0 and so on and so forth.

Finally, you will have a you know you will have something like this X0 dot dot dot dot dot if it is 0 at total length is N. So, it will be X then again next one will be X0 here 0 here and non 0 values X1 and then finally, X0. So, up to this up to this you will have you know this will continue like this I mean a situation will like this will happen when you have non 0 elements ok. I mean I think no need to draw this you can understand that as we go along N this column will have more and more non 0 elements more and more non 0 elements ok. Finally, situation will come when all the columns are non 0 columns maybe some elements may be 0s that is what I was showing here, but at least other elements will be non 0.

So, there will be non 0 columns and then we assume there is no linear relation between them and X transpose and Xn that can be assumed to be positive definite, but as long as that does not happen I mean the initial phase that is at equal to 0 or N equal to 1 N equal to 2 I will have 0 columns some 0 columns and presumably 0 columns makes this Xn rank deficient because there will be a linear relation amongst the columns like Xn into C will be 0 for a non 0 C like here I took C like this one element only 0 first guy others can be any non 0 element ok. So, if you linearly combine the columns by this you will get 0 ok. And in that case we have seen that if C is a non 0 vector and still X transpose and Xn is C, Xn C is 0 alright in that case this cannot be positive definite and it is not invariable. So, this will continue beyond the point of time when all columns are filled up with non 0 elements at least partially if not all the elements are partially then you can safely assume there is no linear relation now between the columns among the columns and this will be positive definite. So, to take care of this case where this is not positive definite for the rank is you know less it is not full rank.



So, not invariable to take care of that what we do we know this is a Hermitian matrix positive definite positive semi definite matrix at least always. So, X transpose n Xn ok, this is a Hermitian matrix positive which includes Hermitian. So, this you should be able to write as some T is Hermitian matrix call it R n Nth index.

$$\underline{R}_n = \underline{X}^t(n)\underline{X}(n)$$

So, it will be any R Hermitian matrix will be TDTH I am just putting the index n here because we are standing at nth clock this consists of all the orthonormal mutually orthonormal eigenvectors of R n this is a diagonal matrix consisting of the eigenvalues which are real in fact, non 0 because it is positive semi definite at least these are thing.

$$\Longrightarrow T_n D_n T_n^H$$

Now, if along with this R n we replace by R n plus some epsilon i, i is identity matrix epsilon is a very small positive constant.

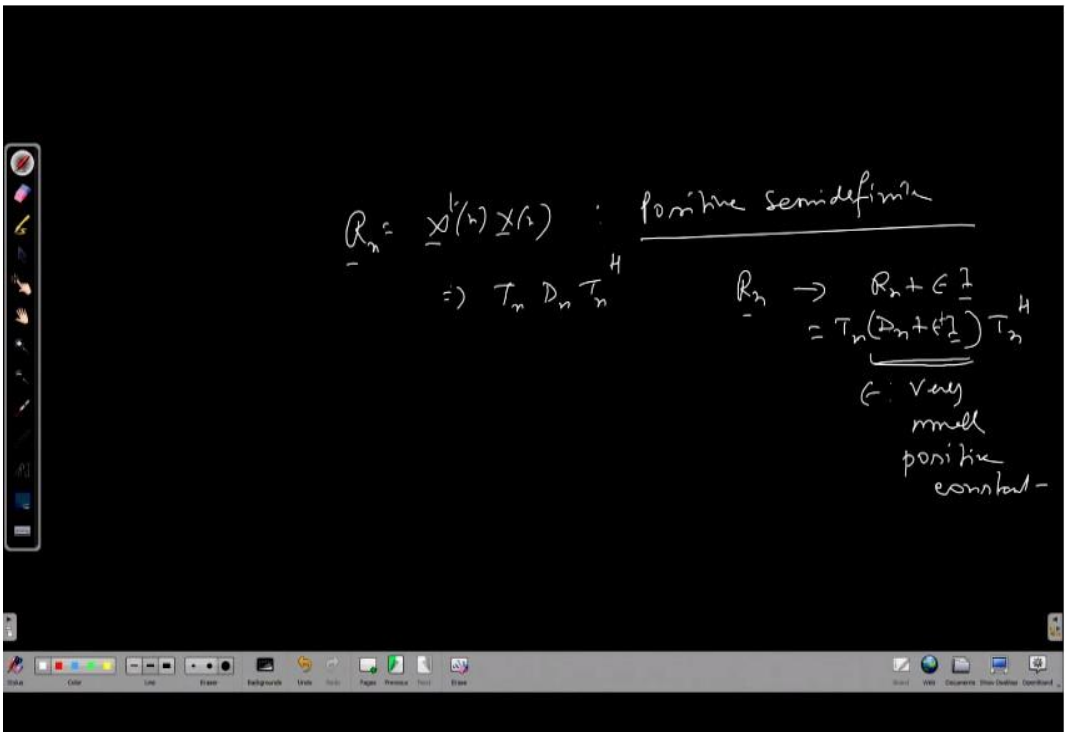
$$\underline{R}_n \longrightarrow R_n + \epsilon \underline{I}$$

So, it will be diagonal matrix with elements epsilon epsilon epsilon epsilon like that. So, this now you can write as T n dn plus epsilon i T nh

$$= T_n(D_n + \epsilon \underline{I})T_n^H$$

because T n dn T nh there is R n and T n epsilon i epsilon is a scalar. So, it can be taken out. So, T n into i is T n T n and T nh that is identity because T n is unitary. So, you get epsilon into i, but now this matrix is such even if R n was not invertible R n was not positive definite meaning some of the eigenvalues were 0.

Now, the real eigenvalue of the overall matrix will be even if the eigenvalue is 0 here that will now the new eigenvalue will be 0 plus epsilon. So, this addition of epsilon will make all eigenvalues non-zero positive because eigenvalues here could be either 0 or positive because it is positive semi definite, but even with any 0 eigenvalue I am adding an epsilon right. It is a diagonal matrix, diagonal matrix. So, suppose i th diagonal element is 0 here, 0 eigenvalue, but corresponding diagonal element is epsilon here because all the diagonal elements here are epsilon epsilon to i. So, real thing will be 0 plus epsilon.

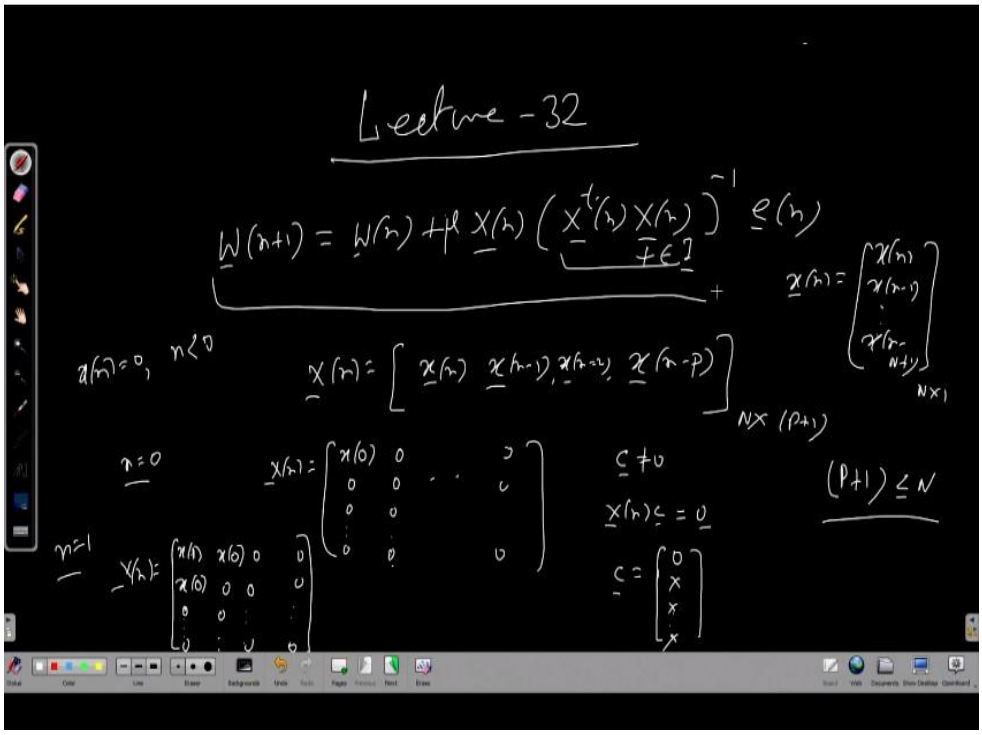


So, corresponding eigenvalue now will not be 0 will be positive epsilon which will make R n invertible. So, this is the thing we do ok. So, here we add 1 epsilon i x transpose n x n plus epsilon i ok. This is one modification and one more thing we have so far considered no noise right. So, suppose W 0 x n there was one x n sorry, let us consider one hyper plane in a 3-dimensional space equal to this is what originally, we thought dn.

So, this was a plane this is this plane, this is W 0 axis, W 1 axis, W 2 axis this is a plane. But now when there is noise this is just a filter output that plus noise is dn ok. So, this is this left-hand side is dn minus some zn where zn is a noise. So, these are actual hyper plane, hyper plane because x n x n minus 1 x n minus 2 have not changed the gradient the slope does not change only it just gets shifted ok. So, wherever it was cutting maybe one plane maybe you can say W 1 W 2 that entire thing will get shifted ok.

So, it will get shifted, so it will move up it might move up like this. Similarly, another plane was like this, so it was cutting it here, but this also will now chase to some other place. So, it will cut somewhere else which means that intersection line earlier was here now it will move somewhere else. But I was projecting on this assuming no noise foolishly I was projecting on this which is intersection between all the hyper planes assuming no noise, but I should actually have projected it here. So, therefore, I multiply it by mu and now in a three dimension by this line may not hit it directly may be up or down, but mu can be taken to be close such that it takes me closes to this ok.

Now, again this position of this is random. So, mu should be chosen randomly you know in every iteration of the algorithm, but that is not possible. So, you just make a provision of a mu. So, mu times this in a statistical sense in an average sense we will make this line we will take this line closest to the actual the desirable actual line of intersection ok.



So, there is a provision of mu. So, you have a mu here. So, this is the affine projection algorithm. You understand we will converge very fast because I am not like unlike the N-LMS algorithm I am not projecting current WN on a particular line or particular hyper plane, but rather on the intersection space intersection hyper plane intersection between several hyper planes that take make it takes my projection closer to the desired or true system vector coefficient vector alright. So, this is for affine projection.

So, so far we considered LMS category ok, LMS category least mean square category, but as I told in the very beginning there is another one more category of adaptation which is equally powerful very equally I mean equally well known and mathematically very exciting that is called recursive RLS.

We have seen one thing in LMS algorithm how did we proceed there was a again we are considering called real value data no complex thing to make life simple. So, suppose W vector these are all standard notation we all know, but still for the sake of completeness I

am rewriting them. This is the filter output YN which is either W transpose XN or equivalently X transpose N W .

$$y(n) = \underline{w}^t \underline{x}(n)$$

$$= \underline{x}^t(n)\underline{w}$$

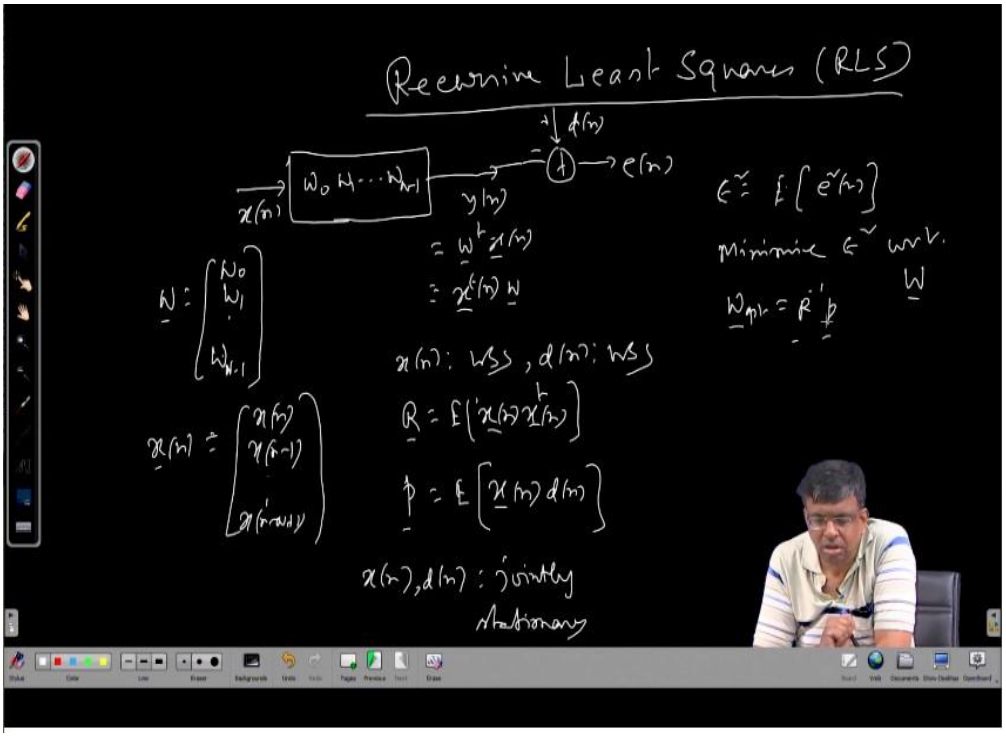
And there was a desired response this EN this was assumed to be XN WSS autocorrelation matrix R was assumed to be known say to start with and cross correlation vector P was it was assumed input is WSS and XN DN also WSS and XN DN jointly stationary.

$$\underline{R} = E\left[\underline{x}(n)\underline{x}^t(n)\right]$$

$$\underline{P} = E[\underline{x}(n)d(n)]$$

That is why when you take the XN vector so, this multiply anybody with DN and take the expected there is a cross correlation between any sample of XN and DN you know N disappears what matters is only the gap. So, this is the thing what we to derive the optimal filter first we started minimizing this we try to minimize with respect to W all right directly we minimize, but then that was giving rise to W opt in terms of this R and P.

$$\underline{w}_{opt} = \underline{R}^{-1}\underline{p}$$

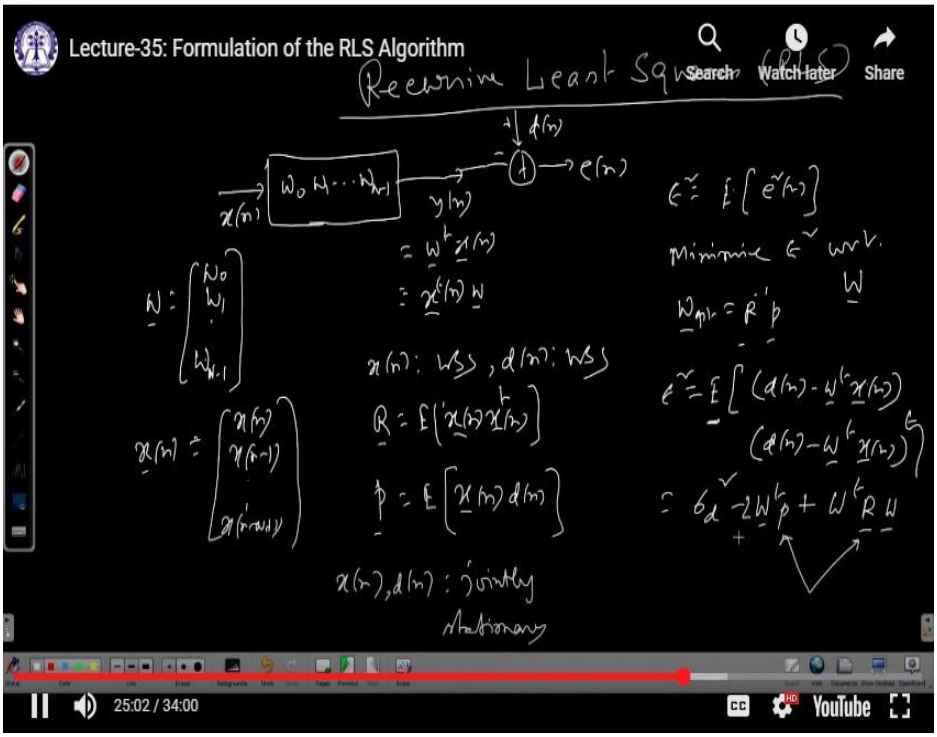


Why R and P are coming there is most important that is coming because I am using this expectation operator. So, EN if you write as DN minus W transpose XN square upon all those finally, R will show up P will show up. We have seen earlier you can see again because of this E business epsilon square is expected value e square N, e square N means DN minus say W transpose I am redoing it to emphasize some point that is why I am doing it only. And again, same thing, but this is a scalar this is a scalar. So, I can you know put a transpose on one of them may be on this.

So, DN, DN transpose and DN transpose is DN only say expected value of DN square which is variance of DN sigma d square minus W transpose XN DN expected value W is constant. So, expected value of expected expectation will go on XN into DN which is XN vector into DN which is P. So, it will be W transpose P other cross term also DN W transpose XN transpose W transpose XN is a scalar. So, it is transpose is itself W transpose XN XN vector DN is a scalar DN you can write to the right. So, W transpose XN DN then e will go inside because W is constant.

So, you will work on XN into DN like here you will again get back W transpose P. Why P is coming because of this expectation operator. I am getting a product XN into DN XN vector into DN like this, but because I am using e, e on that is giving me P. Remember that because of this e failure because this e is causing me some problem which we will see and again W transpose XN transpose of this X transpose NW. So, e will work on XN XN transpose the W transpose will be outside XN XN transpose because this will come here XN transpose this will come as W transpose transpose cancels and e working on XN XN transpose gives me R.

So, these two things we are getting because of this e failure because I am using expected value exact bit square mean square error minimization.



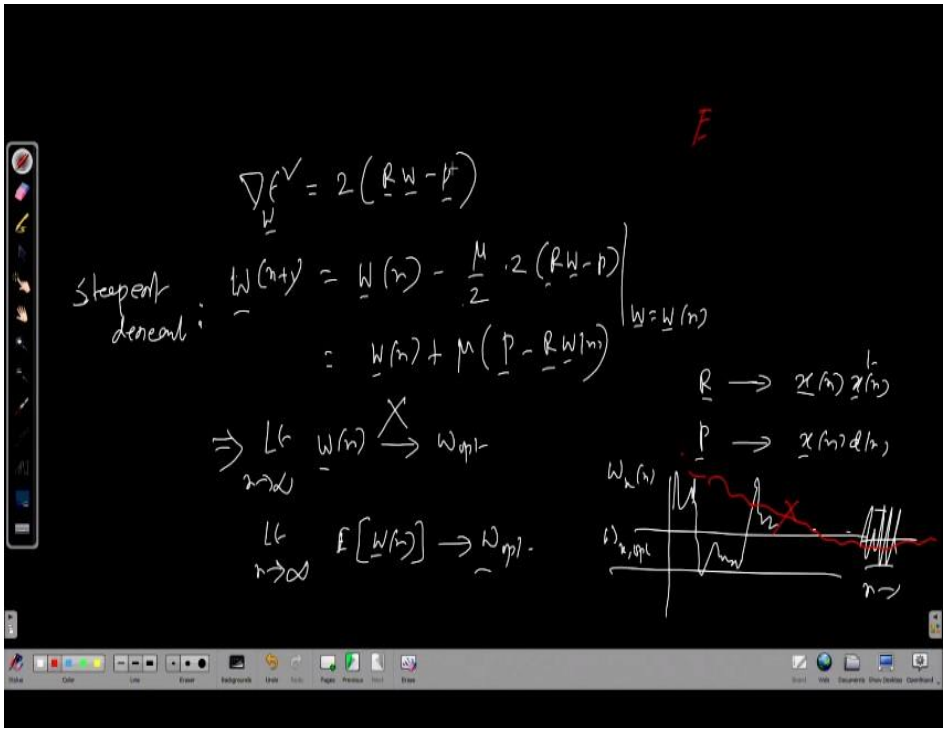
Then what I did I used a steepest descent method I found out the gradient I found out the gradient of this and that gradient was twice R, R is continuing from that expression of epsilon square R W minus P both P and R continuing. So, steepest descent was this minus mu by 2 gradient the gradient evaluated at this point this value of W at W equal to WN.

This is repetition of the old story and then u22 cancels minus you can push inside. So, it is WN plus mu times P minus R WN this was steepest descent then I said that I will replace capital R by that if I know exact R exact P this is very good this will converge exactly to W opt, but exact R exact P is not known because there is time varying input statistics may change from time to time that is what the need of adaptation comes ok.

So, I said let me replace R by a bad estimate just XN into XN transpose I am not really averaging over many that XN into XN transpose plus XN minus 1 into XN minus 1 transpose and dot dot dot over many may be 100 such cases sum and then divide by 100 no I took just one case one XN vector into X transpose I took that to be enough for R. So, it is a bad estimate similarly P I replaced by just XN into DN whereas, ideally, I should have had XN into DN one vector plus XN minus 1 into DN another vector plus dot dot dot 100 such cases add get a new vector divide all the elements by 100 then that would be a good estimate that I am not doing because of these because of these. So, that takes me to LMS algorithm ok, LMS algorithm we all have seen. So, I am not writing the expression again, but what happens we do not have any more this does not happen anymore this was happening if you could give the current value of R current value of P WN would have converge exactly WN, but because of this approximation what is happening is this it is converging in mean alright. So, this is not something very good because what is happening if suppose you take a particular weight WKN and this is a corresponding opt value.

So, it will be fluctuate and this index N it will fluctuating, fluctuating, fluctuating again fluctuating finally, so it will take lot more time and finally, it will be fluctuating around the optimal value ok. So, it will only converge in mean it will not converge on this it will not it will not be like this it will not be like this ok. That is why this error comes ok, weight error, weight error variance all those things come. So, this is a problem see it takes more because of this approximation we inject we basically do not use a correct gradient, but use what is called an incorrect which we call noisy gradient ok, because there is error that is going in as a result we do not get actual convergence we call the only convergence in mean. So, at best the filter weight estimate WKN will be only oscillating around the at best

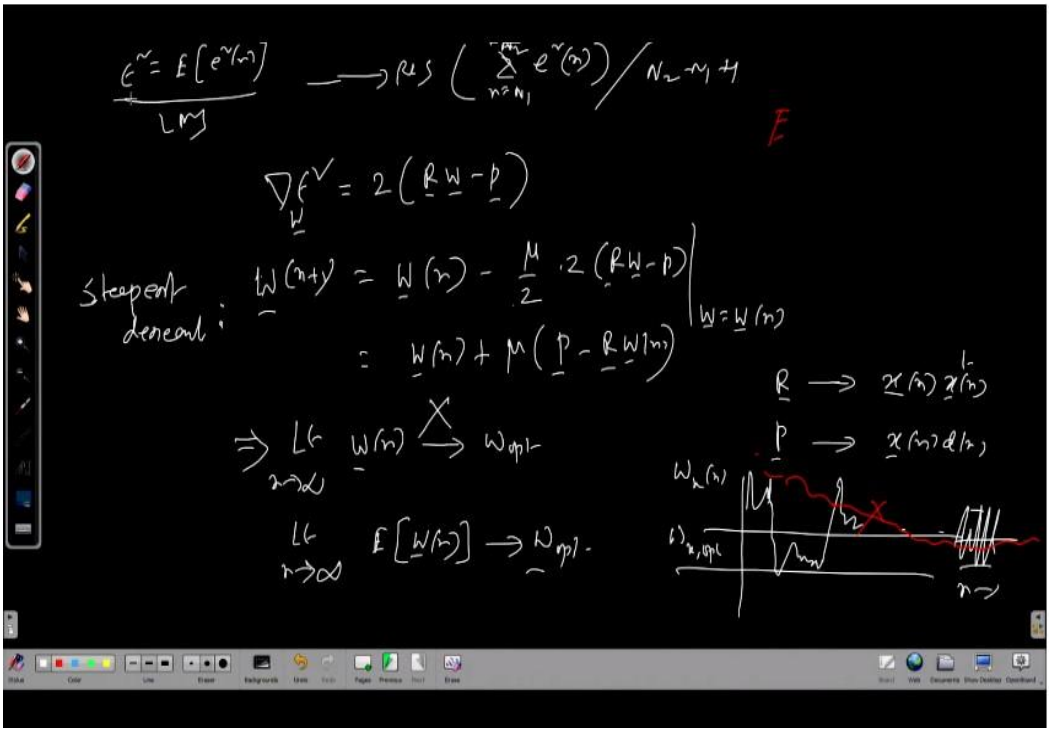
oscillate around the optimal value, but it will not hit the actual optimal value, but of course, if the ring is small I am happy.



Now in RLAs we try to get rid of that E operator because we know where from R and PK because I am using the expectation. So, expectation of XN vector into DN whenever I come whenever I come across I replace it by P, expectation of XN vector into X transpose N whenever I come across I replace it by R and then I am into this trap and finally, I am required to replace R and P by these things and I get into this trouble. So, in RLAs we will rather we will not bring in E operator instead that is here we had expected value of this was LMS. Here in RLAs we will take an we will instead of using this instead of having an epsilon square directly given by E operator we will rather replace this by a good estimate. What good estimate we will take various cases at various points of time maybe E square just a minute you know it is not only at one point we will take it over a long window N equal to maybe capital N1 to N2.

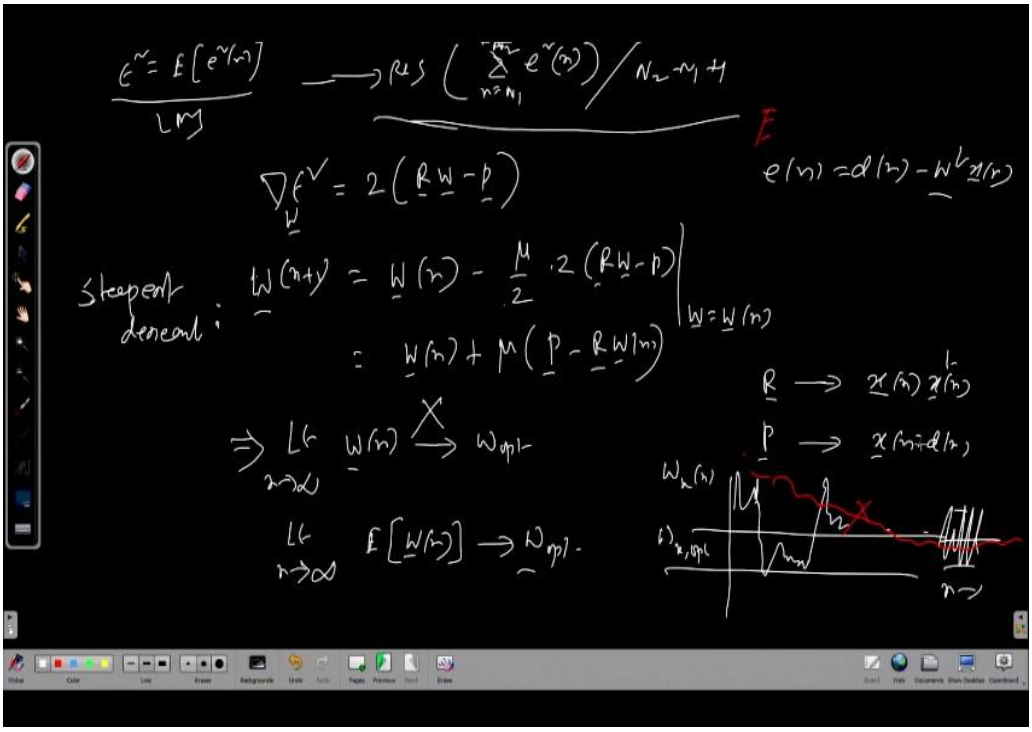
So, we have E square at capital N1 then E square at capital N1 plus 1 and dot dot dot ok.

So, various such value of I mean E square will be considered for the same filter coefficient ok. They give data XN and the input find the output take the error from DN of square up the error again next clock and the data goes in again filter find the output subtract from the new DN square of the error and so and so. So, you are basically taking various samples various experimental outcomes or observations for that output error EN squaring them up for many times how many times N2 minus N1 plus 1 and then averaging.



So, that will be a good estimate of this. So, if your window is large its value will be almost equal to epsilon this actual epsilon square. So, this actual epsilon square if you plot it will be like a quadratic function we have seen this also will follow it almost identically. Because for large window value of this obtained you know by the statistical average will be actually very close to actual epsilon square. So, if you plot this as a function of W because EN consist of the EN is a function of W you know at any EN is DN minus W transpose XN. So, whether here or here this is the function of W if you plot it as a function of W here or here the two curves will look almost identical this will follow this.

But the beauty is here I do not have E operator expectation operator. So, that explicit expression of capital R or P will not come if I minimize this with respect to filter weights. Since the two plots will be almost identical the minima also minima locations of the two also will be almost identical or very close to each other. So, if I minimize this with respect to W I will reach either the exact minima point here or very close there and I will be happy. But I would not have this problem of you know replacing R and P by this bad estimate this is what will be the starting point of recursively square algorithm.



So, I stop here today and we will continue from here in the next class. Thank you very much.