Introduction To Adaptive Signal Processing

Prof. Mrityunjoy Chakraborty

Department of Electronics and Electrical Communication Engineering

Indian Institute of Technology, Kharagpur

Lecture No # 36

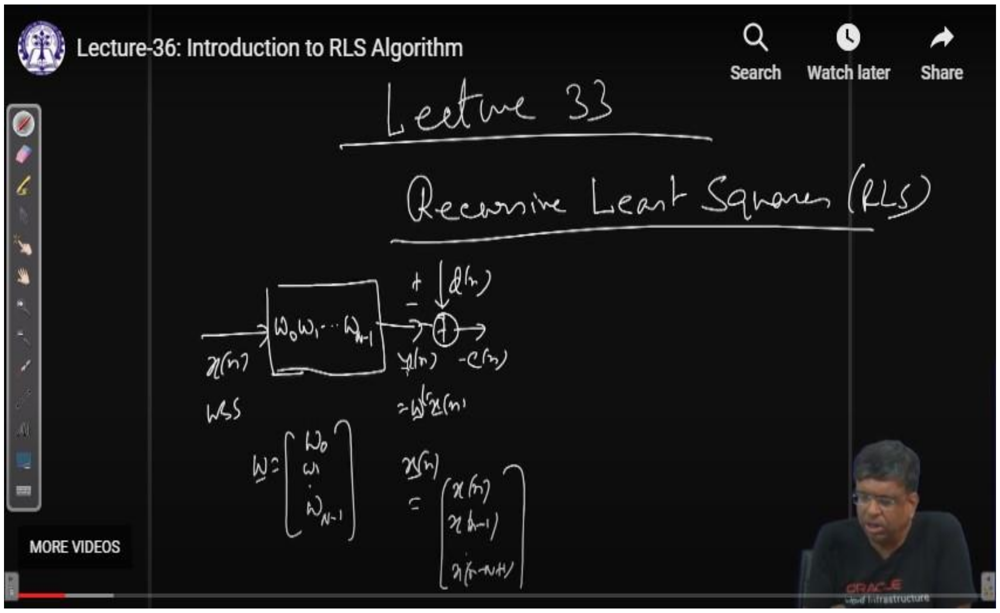
Introduction to RLS Algorithm

In the previous class we introduced you to these, but then the let me again go back and just take a recap of what we did that will make it easy for you to you know I mean connect things to you correct things properly ok. See in the case of LMS what we did there was a input RSS a WSS, these are the filter coefficients output YN you are trying to find out this filter coefficient vector W.

$$\underline{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{N-1} \end{bmatrix}$$

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

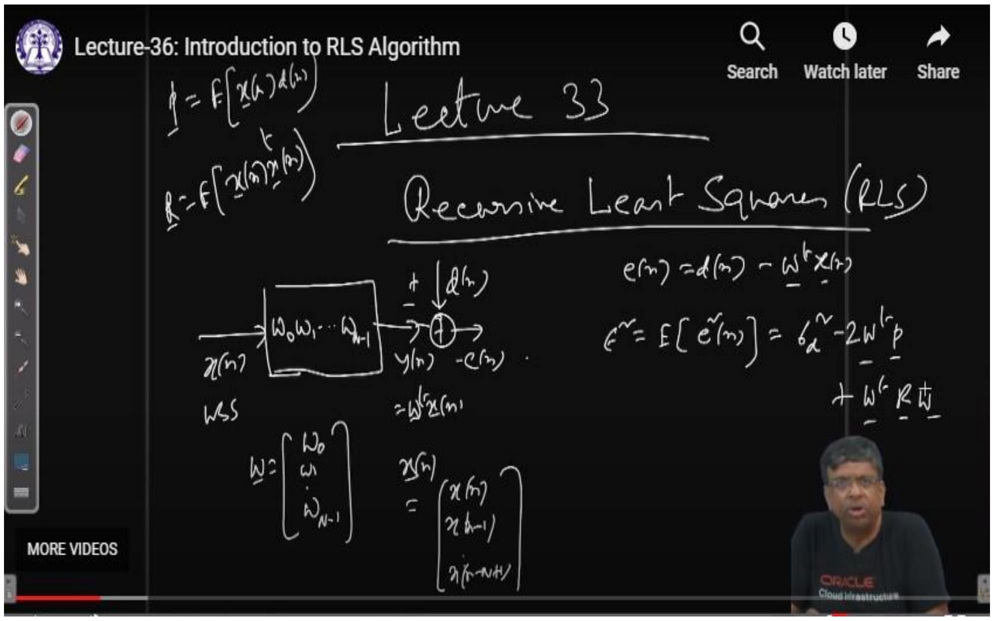
So, that YN is W transpose XN where XN these are all standard and we have done it so many times, but for the sake of completion I have to write every time I start teaching.


Lecture-36: Introduction to RLS Algorithm
Search
Watch later
Share
Lecture 33
Recursive Least Squares (RLS)
MORE VIDEOS

So, EN is DN minus YN that is W transpose XN

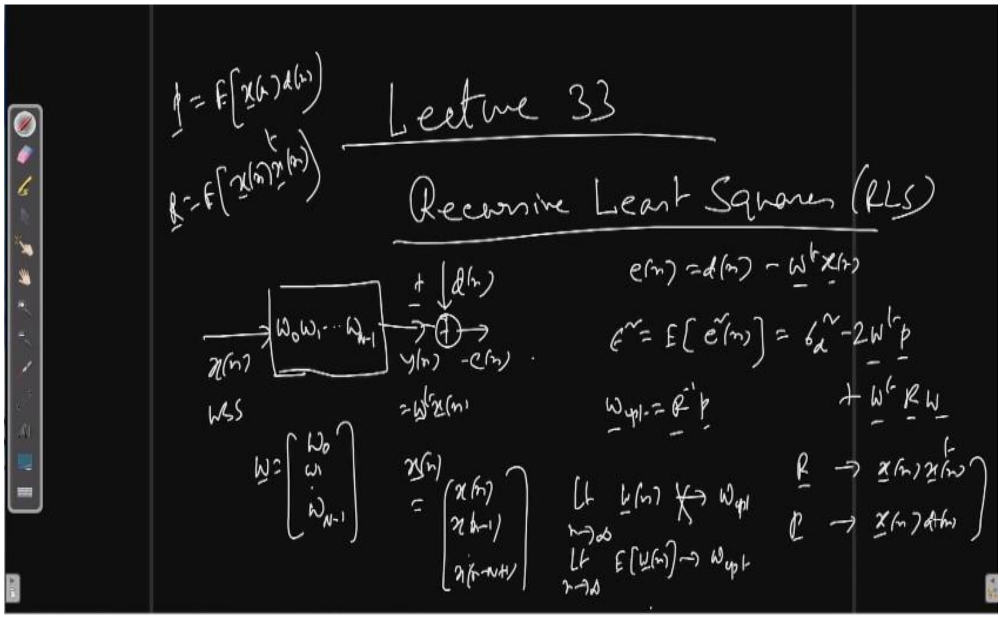
$$e(n) = d(n) - \underline{w}^t \underline{x}(n)$$

we have to find out and there is an assumption that YN also WSS and XN DN they are jointly stationary under that this variance and these are all real. So, I am not putting mod EN square just simply EN square this was we worked out this was sigma d square there is a variance of d then twice W transpose p where p is the cross-correlation vector between XN and DN because p was these are all known stuff and also there is an autocorrelation matrix ok. So, this depends on p and r for a fixed p and fixed r this is a quadratic function of the filter weights because of this term W transpose rW because rW will have elements of W present in every element of the resulting vector rW and then that is multiplied by components of W from this side.


Lecture-36: Introduction to RLS Algorithm
Search
Watch later
Share
Lecture 33
Recursive Least Squares (RLS)
$p = E[\underline{x}(n) d(n)]$
$R = E[\underline{x}(n)\underline{x}^t(n)]$
$e(n) = d(n) - \underline{w}^t \underline{x}(n)$
$\epsilon^2 = E[e^2(n)] = \sigma_d^2 - 2\underline{w}^t \underline{p} + \underline{w}^t R \underline{w}$
$x(n)$
$w_0 w_1 \cdots w_{N-1}$
$d(n)$
$y(n) = \underline{w}^t \underline{x}(n)$
$-e(n)$
WSS
$\underline{w} = [w_0, w_1, \ldots, w_{N-1}]^t$
$\underline{x}(n) = [x(n), x(n-1), \ldots, x(n-N+1)]^t$
MORE VIDEOS

So, that may be second order we will have seen this is first order term. Then the thing we do is this we try to take difference derivative of this with respect to the filter weights ok. Then we get the minima that is equate the derivatives to 0 then we get a solution which is Wopt optimal this is indeed optimal or inverse p assuming r to be invertible that is positive definite we assumed and then we try to make it adaptive because r and p can change from time to time ok or their knowledge may not be available. So, we go for a steepest descent procedure gradient descent search because this curve is a quadratic function it will have minima at a particular W that is Wopt otherwise it will go up up up we have to reach that minima point.

So, we will go for that gradient descent search and we went we get that steepest descent algorithm all right that was in terms of r and p that was in terms of r and p and then we said we have to make it adaptive. So, r we replaced by a bad estimate not so good estimate and we substituted them in the gradient descent update equation we obtain LMS and because of this we pay the price the filter weight Wn does not converge to Wopt absolutely but it converges in min that is every component is now fluctuating because they are generated from the data by the LMS subnet equation and expected value if you take the expected value at any particular nth cycle for all the weights all the. So, this vector min will converge this that is every component of Wn when we say kth meter coefficient Wkn and this Wopt k. So, Wkn will be fluctuating around Wopt k in the steady state as n tends to infinity that is n becomes very large. So, it will be fluctuating so that the min of the fluctuation is this we can get up to this and also convergence will become slow all because of this fact that r and p we could not provide correct data but we just use some approximate so called you know bad approximation.


Lecture 33
Recursive Least Squares (RLS)
$\underline{p} = E[\underline{x}(n)d(n)]$
$\underline{R} = E[\underline{x}(n)\underline{x}^t(n)]$
$e(n) = d(n) - \underline{w}^t\underline{x}(n)$
$\epsilon^2 = E[e^2(n)] = \sigma_d^2 - 2\underline{w}^t\underline{p} + \underline{w}^t\underline{R}\,\underline{w}$
$\underline{w}_{opt} = \underline{R}^{-1}\underline{p}$
$x(n)$
WSS
$w_0 w_1 \cdots w_{N-1}$
$+$
$d(n)$
$y(n)$
$-e(n)$
$= \underline{w}^t\underline{x}(n)$
$\underline{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{N-1} \end{bmatrix}$
$\underline{x}(n) = \begin{bmatrix} x(n) \\ x(n-1) \\ \vdots \\ x(n-N+1) \end{bmatrix}$
$\lim_{n\to\infty} \underline{w}(n) \not\to w_{opt}$
$\lim_{n\to\infty} E[\underline{w}(n)] \to w_{opt}$
$\underline{R} \to \underline{x}(n)\underline{x}^t(n)$
$\underline{p} \to \underline{x}(n)d(n)$

Now in the case of RLS what we do epsilon square we did not we do not use this e operator e is the root cause of the problem the moment you bring e, e of xn dn you have p, e of xn transpose R and p and R come in all our expressions and then in the end you give up you have to replace R by a some estimate. So, you take this bad estimate p also take this bad estimate and get LMS and p in those prices we have to get rid of that e operator. Therefore what we do we approximate this epsilon square by what by taking many samples of e square n, e square n, e square n minus 1, e square n minus 2, e square n minus 3 all that for the same filter weight that is I keep giving xn hold the same filter weight find out yn and en. So, take e square n for the same weight again e square n minus 1, e square n minus 2 dot dot dot take many add them and then average. So, if you take many and then average that will be very good approximation of actual epsilon square but you see there is no e operator expectation operator ok.

But the value of that will be almost equal to epsilon square if you have many samples. Therefore, on one hand if you can plot this epsilon square against the filter weights it will be a quadratic surface. So, if you take the sample average that is e square n added over no I mean averaged sample wise if you plot that also versus w you will get more or less the same curve same identical curve because at any for any w the sample average will be following this epsilon square very much that is in the case of RLS let me elaborate this further. Suppose we maintain the same weights xn yn is w transpose xn which is also x transpose w all right.

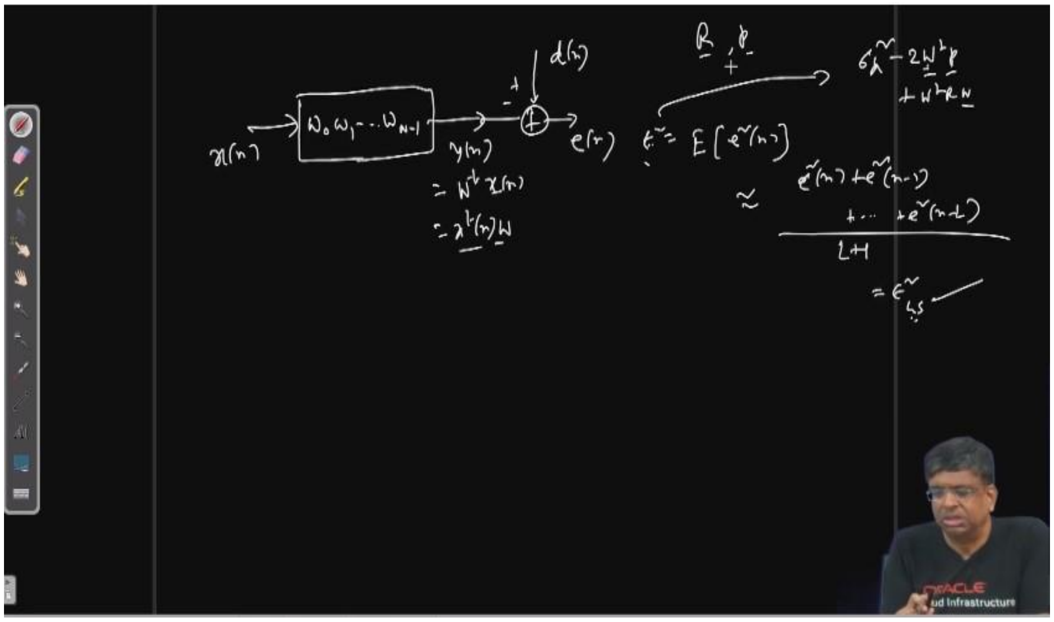
$$y(n) = \underline{w}^t \underline{x}(n)$$

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

So, we keep the same weights but keep sending in new and new input and find out new and new en ok.

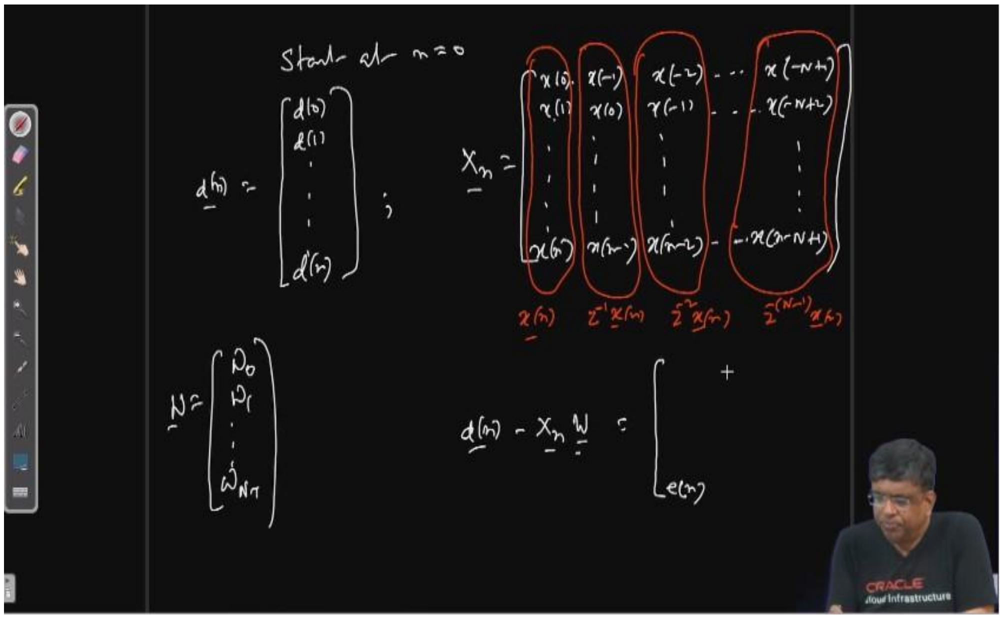
So, for the same weight we find out this as I mean for this, we take a good I mean sample estimate that is e square n e square n minus 1 dot dot may be e square n minus something may be L and divide by how many samples L plus 1 L plus 1. So, if L is large this function this function, I can call epsilon square Ls e squares this and epsilon square they will be very close to each other ok. This is the actual mean square error but has that e operator and therefore, if you we have already given the expression in the previous page that will have R and P that is coming because of that e operator. But suppose I replace this by a good estimate that is when that is this that when I was giving you epsilon square, I kept the filter weights as it is ok. And for that particular chosen filter weight found out e square n take expected value and we got that expression some sigma d square we already did in the previous page minus twice w transpose P plus w transpose R w ok that was this.

And just P and R that became because of this e operator. So, for the same w if I keep the same w here and now keep sending data and finding out e n, e n minus 1 e n minus 2 and all those then for large L if I take this average e square n e square I mean so many cases that we just square up the output error add them divide by the number of terms and take length will be large. So, this will definitely which is called epsilon square L s, this will be a very good estimate of epsilon square. So, epsilon square versus w plot and epsilon square L s versus w plot they will be almost identical because the two for any w the two values are very close almost same because I have taken many samples is not it. So, therefore, instead of reaching the minima of this epsilon square if I use the minima of epsilon square L epsilon L s square I will still be happy I will be very close to the minima of epsilon square ok.



So, we try to minimize this this is the philosophy, but by that we are getting the advantage that we are no longer using expectation operator e operator and therefore, R and P they will not curve explicitly alright and this is the philosophy of RLS. So, what we do now let me teach more formally suppose we start at n equal to 0 ok we have a vector we have a dot of matrix. So, x0 to xn very long vector maybe starting from 0 to n ok. The next column will be one cycle delayed version. So, if it is x0 it will be x minus 1 if it is x1 it will be x0 if it is xn it will be xn minus 1.

Next column another cycle delayed version since x minus 1 it will be x minus 2 x minus if it is 0 so 1 0 minus 1 0 minus 1 minus 2 n n minus 1 n minus 2 dot dot dot dot maybe we take up to some n minus n plus 1 ok. So, if n is 0 it is x minus n plus 1 if n is 1 dot dot dot dot dot this column for the time being only this column for the time being we will call it xn vector later xn will be used to indicate some other vector but for the time being xn. This vector it is one cycle delayed version right when I have x0 that time this fellow is x minus 1 when I have x1 then x0 when I have xn xn minus 1. So, use a notation gen inverse xn just a notation only as this vector is delayed by one cycle this will be and lastly alright and let W be alright then tell me what this is if I take the Vn minus this xn into W. Dn start with the last component D of small n minus last row times W last row times W means W0 into xn plus W1 into xn minus 1 plus W2 xn minus 2 dot dot so it is a filter output at nth clock alright.

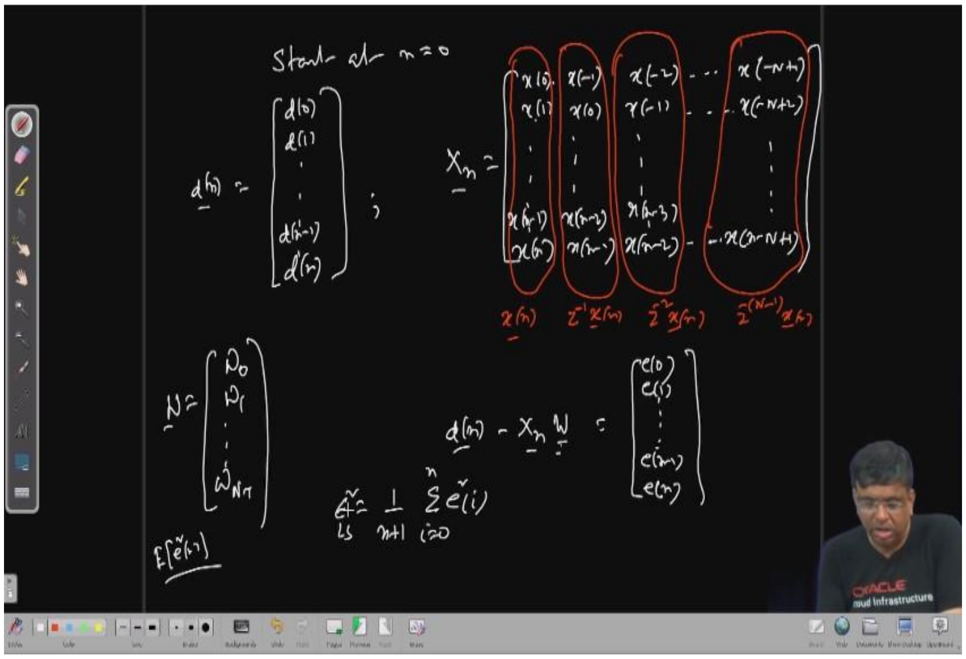


Start at n=0
d(n) =
d(0)
d(1)
d(n)
X_n =
x(0) x(-1) x(-2) ... x(-N+1)
x(1) x(0) x(-1) ... x(-N+2)
x(n) x(n-1) x(n-2) ... x(n-N+1)
x(n) z^-1 x(n) z^-2 x(n) z^-(N-1) x(n)
W =
w_0
w_1
w_N-1
d(n) - X_n W =
e(n)

So, D of n minus the filter output at nth clock for this choice of n vector okay that will be En nth clock D of n minus when you are multiplying xn into W then the last row times W. So, W0 xn W1 xn minus 1 Wn minus 1 into xn minus capital N plus 1 that summation that the filter output at nth clock that you are subtracting from Dn. So, what you are getting is En then you can say Dn minus 1 it will be xn minus 1 n minus 2 like that dot dot dot. So, next term will be Dn minus 1 minus this row last but one row times W. So, W0 into xn minus 1 plus W1 into xn minus 2 plus W2 into xn minus 3 dot dot dot which is the filter output at n minus 1th clock that subtracted from Dn minus 1.

So, it will be En minus 1 and like that dot dot dot x1 so n is 1 here at 1th clock I have D1 minus this row second row times W, x W0 x1 W1 x0 dot dot dot. So, it is the filter output at clock 1 subtracted from D1 so it will be E1 by the same logic this will be E0 all right. So, this our purpose is to minimize how many rows here 0 1 up to small n so n plus 1 n plus 1 terms right. So, if I take square up all the errors, I have kept the weight vector fixed just running the experiment and various indices clock indices and finding out various errors squaring them up squaring them up adding dividing by the number of terms. So, this is what is my epsilon square Ls if n is large or n plus 1 is large this will be a very good estimate of mean square error ok.

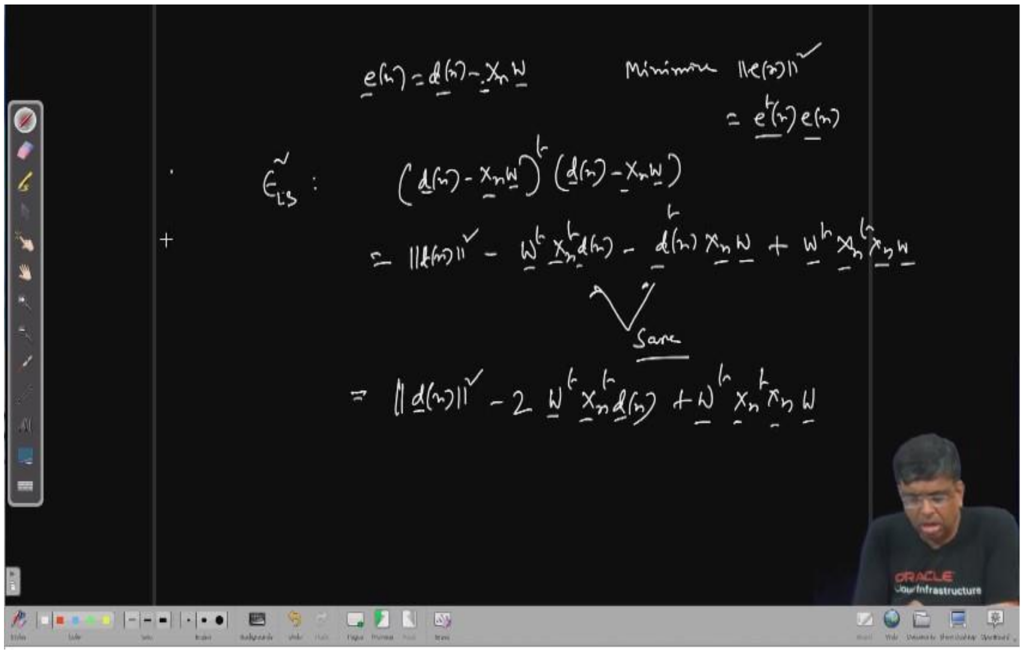
It is a very good estimate of mean square error, but E operator is no longer present right. So, we minimize this with respect to weight after all this vector is Dn minus xn W. So, this vector every term depends on W ok. So, now we will minimize it with respect to W that will be the best estimate optimal one and that will be close to the optimal one which we get there because this epsilon square Ls or this epsilon square they are almost same. So, minimize here and minimize here they are also same ok.


Start at n = 0
d(n) =
d(0)
d(1)
d(n-1)
d(n)
X_n =
x(0) x(-1) x(-2) ... x(-N+1)
x(1) x(0) x(-1) ... x(-N+2)
x(n-1) x(n-2) x(n-3)
x(n) x(n-1) x(n-2) ... x(n-N+1)
x(n) z^{-1}x(n) z^{-2}x(n) z^{-(N-1)}x(n)
W =
w_0
w_1
w_{N-1}
d(n) - X_n W =
e(0)
e(1)
e(n-1)
e(n)
ε̃²_Ls = 1/(n+1) Σ_{i=0}^{n} ẽ(i)
E[ẽ²(n)]

If you plot epsilon square versus filter weights and plot epsilon square Ls versus filter weights the two plots will be almost identical to each other because the values are same almost same is very good estimate with the number of terms is less large alright. So, our now purpose will be to minimize this now is now denominator is n plus 1 that does not depend on W. So, we will be minimizing only this much ok. So, our task is to minimize this sum of square of errors is called sum of squared error. Every error is squared and then sum sum of squared error we will minimize this with respect to filter weights, but this is equal to this is same as you can see if I take this vector as en vector this is nothing but norm square of this that is en transpose en is it not.

If you take en transpose en e0 e0 e0 square e1 e1 e1 square e2 square all that right. So, essentially we will be minimizing the norm square of this error vector with respect to filter weights alright. So, go to the next page we had en I rewrite then we have to minimize the e transpose and en replace en by this that means. So, is a column vector after transposition row vector. So, dn transpose dn which is norm square of dn where it is independent of W then xn W transpose dn.

So, this is W transpose xn transpose dn and another one dn transpose this is a column vector now it is a row vector dn transpose xn W and then xn W transpose xn W. So, W transpose xn transpose xn W this is what you have. Now these two terms are same what is the transpose of the other because it is a scalar all the terms are scalar here is it not because this overall thing is norm square of en which is a scalar you can see here also W transpose is a row vector xn transpose is a matrix. So, row vector into matrix is a row vector row vector column vector is a scalar like that these two are same because if you take the transpose of this you get this is it not. If you take transpose dn transpose comes first then transpose of this W transpose xn transpose which is xn W.



$\underline{e}(n) = \underline{d}(n) - \underline{X}_n \underline{W}$ Minimize $\|\underline{e}(n)\|^2$ ✓

$= \underline{e}^t(n)\underline{e}(n)$

$\tilde{\epsilon}_{LS}$: $(\underline{d}(n) - \underline{X}_n\underline{W})^t(\underline{d}(n) - \underline{X}_n\underline{W})$

$= \|\underline{d}(n)\|^2$ ✓ $- \underline{W}^t\underline{X}_n^t\underline{d}(n) - \underline{d}^t(n)\underline{X}_n\underline{W} + \underline{W}^t\underline{X}_n^t\underline{X}_n\underline{W}$

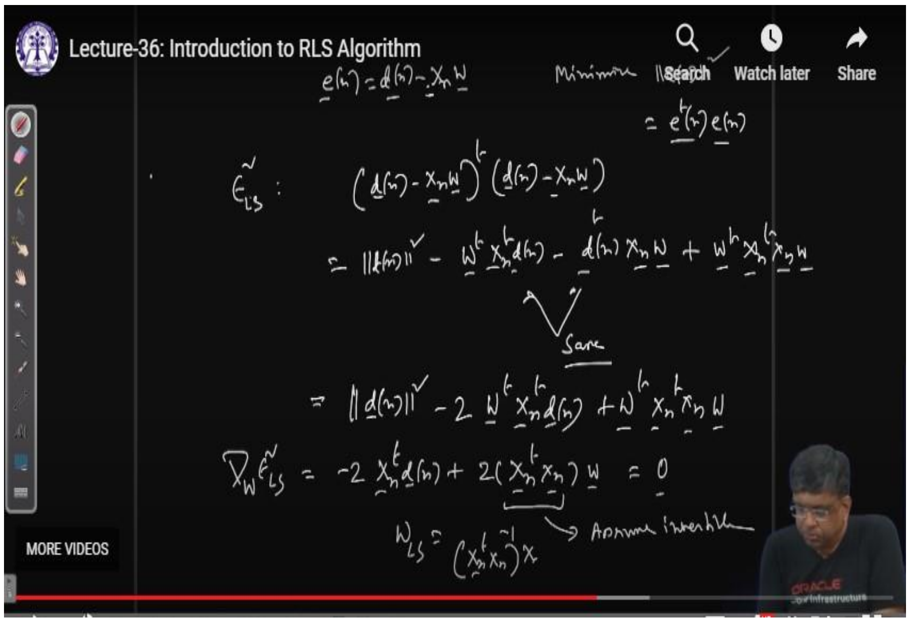
Same

$= \|\underline{d}(n)\|^2$ ✓ $- 2\,\underline{W}^t\underline{X}_n^t\underline{d}(n) + \underline{W}^t\underline{X}_n^t\underline{X}_n\underline{W}$

So, one is the transpose of the other, but both are scalar. So, scalar and its transpose they are same is it not scalar and its transpose they are same. So, if they are transpose of each other then they are same because each is a scalar. So, this is equal to twice this is by epsilon square Ls you can say I mean I am not taking that 1 by n plus 1 because I am taking on the denominator a numerator. So, therefore, if I take the gradient of this I have to minimize it right.

So, we all know W transpose this is a column vector xn transpose is a matrix times dn is a column vector. So, W transpose column vector when differentiated with respect to W and put in a stack it will be just that column vector. So, it will be minus twice and this kind of things we have done many times in the past. Latest case being that case of normalized LMS algorithm even affine projection that time also we did this kind of differentiation. And again W transpose some matrix which the Hermitian matrix into W xn transpose xn is the Hermitian matrix you take transpose of it xn transpose xn transpose transpose that is xn you get the same thing.

So, here again we know it is twice the Hermitian matrix times W if it to 0 vector that will give me W opt. So, here W Ls this squares if this is invertible and we will show that this is indeed invertible in practical cases.



$$\underline{e}(n) = \underline{d}(n) - X_n \underline{W}$$

Minimum $\|$ $= \underline{e}^t(n)\underline{e}(n)$

$$\tilde{\epsilon}_{LS} : \left(\underline{d}(n) - X_n\underline{W}\right)^t \left(\underline{d}(n) - X_n\underline{W}\right)$$

$$= \|\underline{d}(n)\|^2 - \underline{W}^t X_n^t \underline{d}(n) - \underline{d}^t(n) X_n \underline{W} + \underline{W}^t X_n^t X_n \underline{W}$$

Same

$$= \|\underline{d}(n)\|^2 - 2\,\underline{W}^t X_n^t \underline{d}(n) + \underline{W}^t X_n^t X_n \underline{W}$$

$$\nabla_W \tilde{\epsilon}_{LS} = -2 X_n^t \underline{d}(n) + 2\underbrace{(X_n^t X_n)}\underline{W} = \underline{0}$$

$\rightarrow$ Assume invertible

$$W_{LS} = (X_n^t X_n)^{-1} X$$

So, this will be okay. So, this is xn there are how many columns? N number of columns there are N number of columns and number of row small n plus 1 assume number of row is much more than number of column. Now, these columns we can assume them to be linearly independent easily because they randomly generated data there is no linear relation involving the columns relating the columns.

So, as such we have seen xn transpose xn Hermitian if a positive semi definite if you take any c nonzero any vector c and c is length n plus 1 cross 1 any nonzero vector c transpose this matrix xn transpose xn c is nothing but if you take xn c and this is xn c transpose because if you take xn c if you take xn c take the transpose of it c transpose comes first xn transpose there. So, that is what it is. So, xn c if you call it a vector d this is d transpose. So, it is nothing but d transpose d which is norm square this is always greater than equal to 0 if this is to be 0 greater than equal to 0 that makes it positive semi definite. So, let us see why it becomes 0 if it becomes 0 xn c if the norm square is 0 every element of the vector has to be 0 because norm square of a vector if you call it d vector d xn into c is a d vector.

$$\underline{X}_n^t \underline{X}_n\text{: Hermitian}$$

$$\underline{C} \neq 0$$

$$\underline{C}: (n+1)X$$

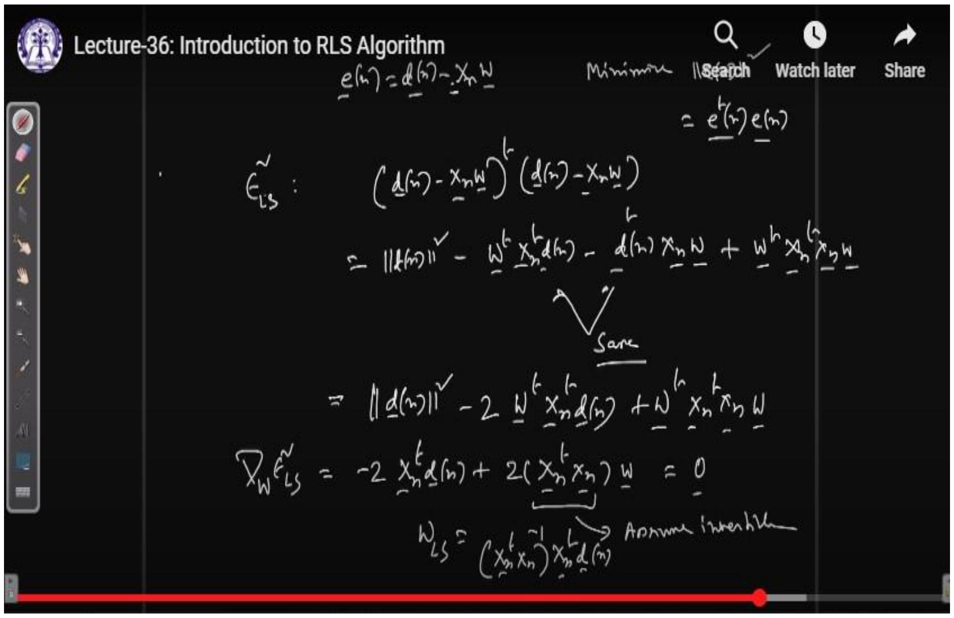
$$C^t X_n^t \underline{X}_n C = \left\|\underline{X}_n \underline{C}\right\|^2 \geq 0$$

So, norm square will be d 0 square plus d 1 square plus d 2 square dot dot dot all are non-negative and they are adding only no subtraction no minus sign. So, nothing like you know this cancels with that that does not happen d 0 square plus d 1 square plus d 2 square dot dot dot added. So, if that is has to be equal to 0 d 0 has to be 0 d 1 has to be 0 d 2 has to be all simultaneously. So, equal to 0 occurs only when the d vector that is xn c is 0, but xn c is 0 and c non-zero means what if you have some elements of c non-zero and when you are doing xn into c you are linearly combining those columns by the corresponding non-zero elements of c and equating to 0. So, that is leading to a linear relation because then you can keep one column on one side others on the right-hand side.

For instance, all the elements of c are non-zero. So, c 0 c 1 c 2 up to c capital N a capital N minus 1. So, xn into c will be c 0 into first column xn plus c 1 into second column c 2 into third column and dot dot equal to 0 and none of the coefficient is 0 neither c 0 0 dot c 1 0 suppose that is the case then you can happily keep one on one side others will be on the other side ok. And this will be divide both by c 0. So, this column will be expressed as a linear combination of the other columns and so on and so forth which means the columns have some linear relation we say they are linearly independent linear relation among them.

But that is that does not occur in practice because data is coming purely randomly there is no linear binding relation on them ok. That is why it is safe and logical to assume the columns are linearly independent of course your number of rows should be more than the number of columns if there is not suppose small n plus 1 there is a number of rows there is less the number of columns. Now if every column is in a space of dimension small n plus 1 because like this n plus 1. So, you need small n plus 1 number of axis or basis vectors to express anybody. If you have more than small n plus 1 the extra vectors, they will be expressible as a linear combination of those small n plus 1 number of basis vectors or axis vectors ok.

That should not happen because then again there is a linear relation you can ask me, but we started n equal to 0 1 2. So, we have to go through these stages right then only come to very large n yeah, those initial stages I will take care later for the time being assume we are at a very large n. So, number of rows is higher than number of columns ok. So, that is why it is definite so invertible and what it means see this ok you can do one thing now bring a n plus 1 here ok inverse inverse inverse cancel so n plus 1 and to cancel it as a 1 by n plus 1.



Lecture-36: Introduction to RLS Algorithm

$$\underline{e}(n) = \underline{d}(n) - X_n \underline{W}$$

Minimum $\|\underline{e}(n)\|^2 = \underline{e}^t(n)\underline{e}(n)$

$$\tilde{\epsilon}_{LS} : (\underline{d}(n) - X_n\underline{W})^t(\underline{d}(n) - X_n\underline{W})$$

$$= \|\underline{d}(n)\|^2 - \underline{W}^t X_n^t \underline{d}(n) - \underline{d}^t(n) X_n \underline{W} + \underline{W}^t X_n^t X_n \underline{W}$$

Same

$$= \|\underline{d}(n)\|^2 - 2\,\underline{W}^t X_n^t \underline{d}(n) + \underline{W}^t X_n^t X_n \underline{W}$$

$$\nabla_W \tilde{\epsilon}_{LS} = -2 X_n^t \underline{d}(n) + 2(\underbrace{X_n^t X_n}) \underline{W} = \underline{0}$$

$$W_{LS} = (X_n^t X_n)^{-1} X_n^t \underline{d}(n)$$ Assume invertible

This I write as it is that 1 by n plus 1 I write here they are scalars so I can write them anywhere so this I write here this is nothing but n plus 1 inverse inverse and xn transpose xn inverse.

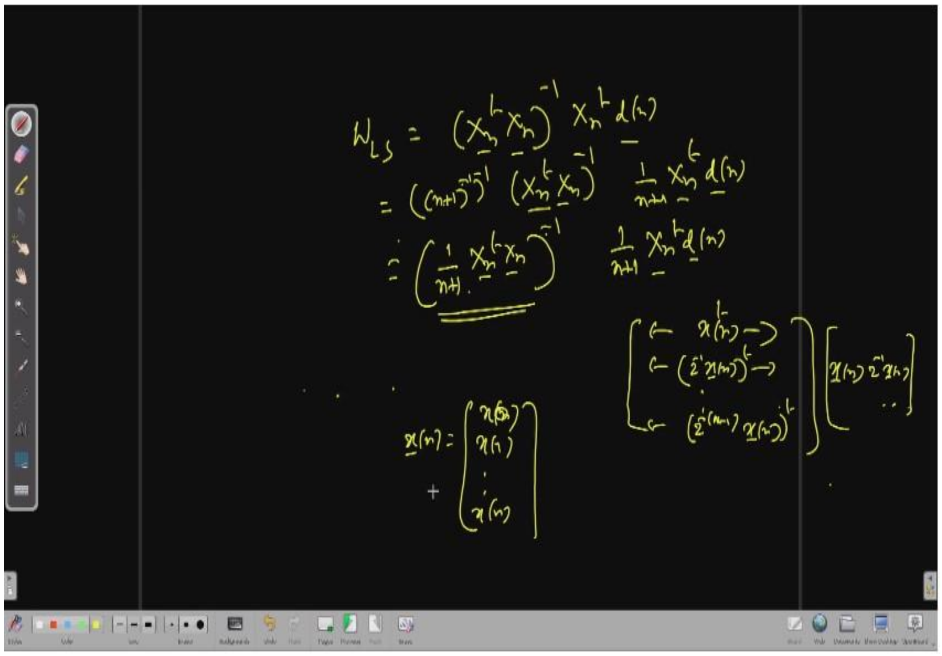
$$W_{LS} = \left(\underline{X}_n^t \underline{X}_n\right)^{-1} X_n^t \underline{d}(n)$$

$$= ((n+1)^{-1})^{-1}\left(\underline{X}_n^t \underline{X}_n\right)^{-1} \frac{1}{n+1} \underline{X}_n^t \underline{d}(n)$$

So, this can go inside n plus 1 inverse is 1 by n plus 1 this inverse very easily 1 by n plus 1 if it is the inverse of it it gives you n plus 1 n plus 1 and 1 by n plus 1 cancels.

$$= \left(\frac{1}{n+1} \underline{X}_n^t \underline{X}_n\right)^{-1} \frac{1}{n+1} \underline{X}_n^t \underline{d}(n)$$

Now let us see what is this xn you remember this was xn so if it is xn transpose first column will become first row second column second one like that ok. So, this will be first column was xn now it is xn transpose then it was gen inverse xn transpose this is xn transpose and xn is dot dot dot dot dot now when you multiply xn transpose by xn what you are getting xn, xn transpose xn means x0 x0 square x1 x1 square dot dot dot up to xn xn square added and then divided by 1 by divided by n plus 1. So, it will be a good estimate of what variance there are 0 mean the variance that is you have to find out the variance of the data you just square up every data add and average and that is what is happening xn transpose into xn means your x0 this is a row vector of this. So, it becomes x0 into x0 x1 into x1 x2 into x2 like that it takes 0 square plus x1 square plus x2 square dot dot dot up to xn square ok sum and they divide by the number of terms n plus 1.



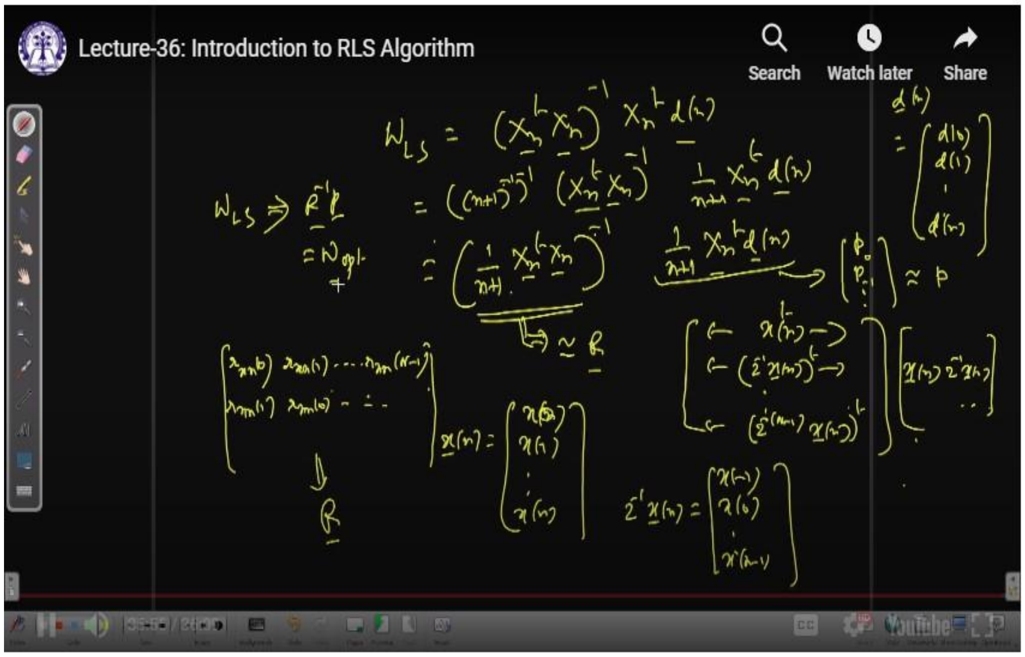
So, that will be a good estimate of so this matrix first row first column will give you and then 1 by n plus 1 this will be close to rxx 0 then xn transpose next guy next guy is here, x0 x minus 1, x1 x0 gap is 1, 0 minus 1, 1 0, n n minus 1. So, multiplying 2 samples at a gap of 1 and then adding them and averaging. So, that will be a good estimate of correlation with gap 1 xnth data n minus 1th data multiply xn minus 1th n minus 2th multiply 1th 0th 0th minus 1th you multiply and add and average it will become next guy this first row times next column rxx 1 and like that like that you can see it will be lastly rxx n minus 1. Similarly, this transpose xn is same as x transpose n this ok z inverse xn transpose xn or xn transpose z inverse xn they are same. So, this also will be rxx 1 the z inverse xn transpose with itself means this vector with itself.

So, x minus 1 square x0 square dot dot dot up to xn minus 1 whole square add all of them divide by n plus 1 again for large n it will be a good estimate of rxx 0. So, it will be rxx 0 and it will continue. So, this will become at r matrix. So, this is getting this matrix is becoming approximately r and here xn transpose dn. So, if you have got dn xn transpose dn vector dn was so xn transpose.

So, take a transpose of it then dn so x0 d0, x1 d1, xn dn. So, multiplying samples of both at a gap of 0. So, this is a cross correlation with gap 0 and then dividing by n plus 1. So, you are averaging so cross correlation with gap 0. So, this will be actually a matrix first guy is that cross correlation vector 0.

Then next is delayed vector these times dn. So, xn minus 1 dn, xn minus 2 dn minus 1, x0 d1 x minus 1 d0. So, it will be again cross correlation with gap 1. So, it was p in fact, it is p minus 1 gap minus 1 n minus 1 and n. So, difference is minus 1 and dot dot dot.

So, this is actually p. So, as a result if n is large and these averages are very good averages the w Ls directly goes to R inverse p which is w opt.



So, this sum of square minimization gives you the same optimal weight if you have large number of samples, but again you do not need any R and p in this calculation it is all data data data right. Now, what purpose will be this that suppose we have taken data up to nth clock you found out this estimate. If I have new data xn plus 1 coming in how will that be updated that is the recursive least squares adaptive filter that is what we will do in the next class. Thank you very much.