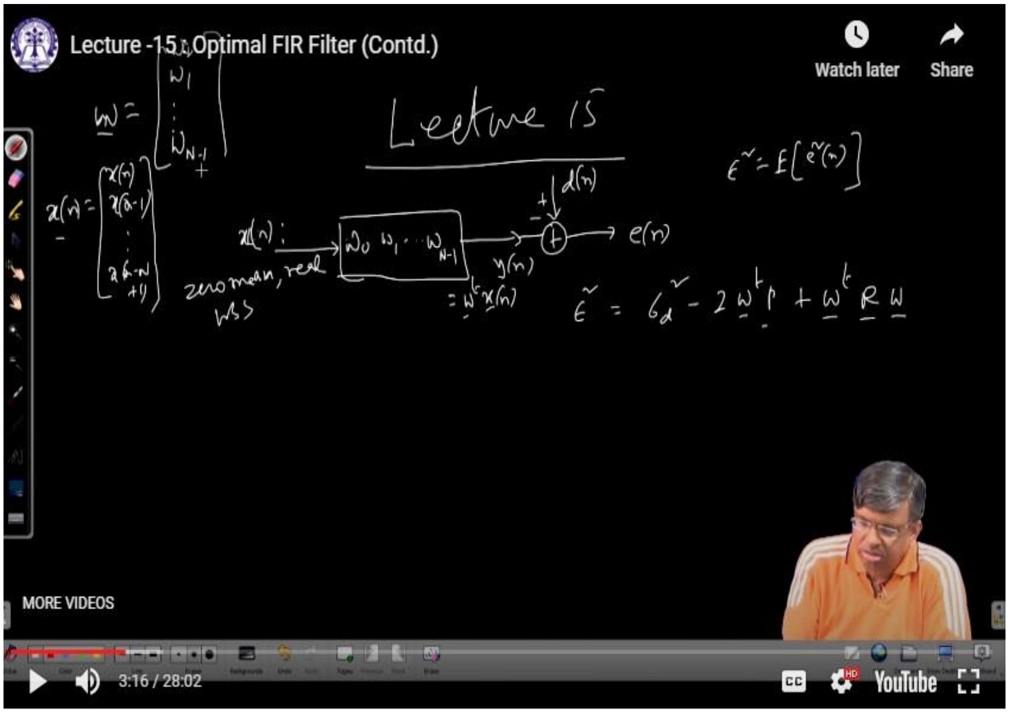
Introduction To Adaptive Signal Processing

Prof. Mrityunjoy Chakraborty
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture No # 15

Optimal FIR Filter (Contd.)

We are considering optimal filter right. We had this these are the filter coefficients there is a zero beam WSS process, real WSS, yn and it should be a good estimate of a targeted sequence called desired response so that that error its variance is minimized ok. Epsilon square which was is real so mod e square or e square they are same that is e square n expected value that is the average power because of WSS nature, this thing epsilon square is independent of n ok. And then you have shown that this is if we worked out the formula for epsilon square and we had W vector was is W0 W1 dot dot dot W n minus 1 all the filter coefficients stacked input data vector all these were defined in the last class. So, yn is W transpose xn this we have seen yesterday in the last class and we also worked out this expression that is sigma d square which is the variance of dn minus twice W transpose P which is a linear first order terms in terms of W0 W1 all that this is not second order. But there was a second order term W transpose this is rho vector R matrix W, R matrix into W so all components of W will be present in the real thing vector elements or W is a column vector in every element of the column vector all the components of W will contribute and then they will get multiplied by another term another W term from this.

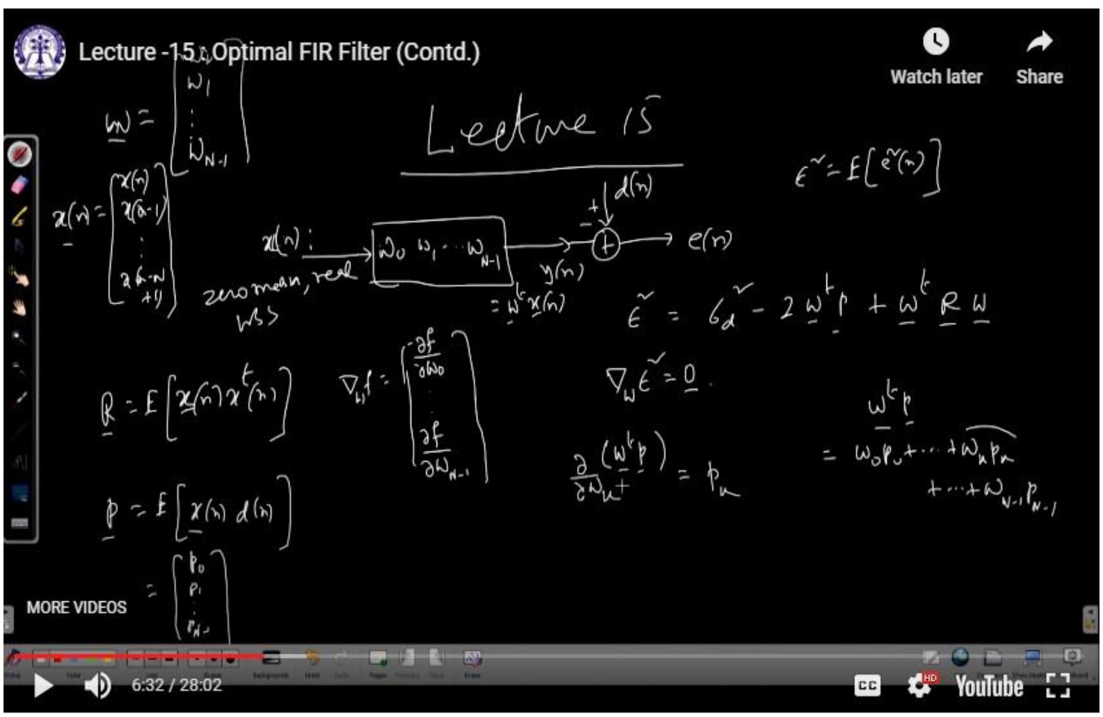


So, essentially it will be a second order thing either W0 square W1 square W2 square kind of terms or W0 W1, W1 W2, W0 W2 like that ok. Together this is a second order thing and if we we can then minimize if we take partial derivative equal to 0 with respect to each of the weights ok. Then we get n equations because n derivatives and second order stuff when derived it will give us to first order equation. So, we get first order equations total n capital N in number we solve them and find out the optimal one this is what we are doing.

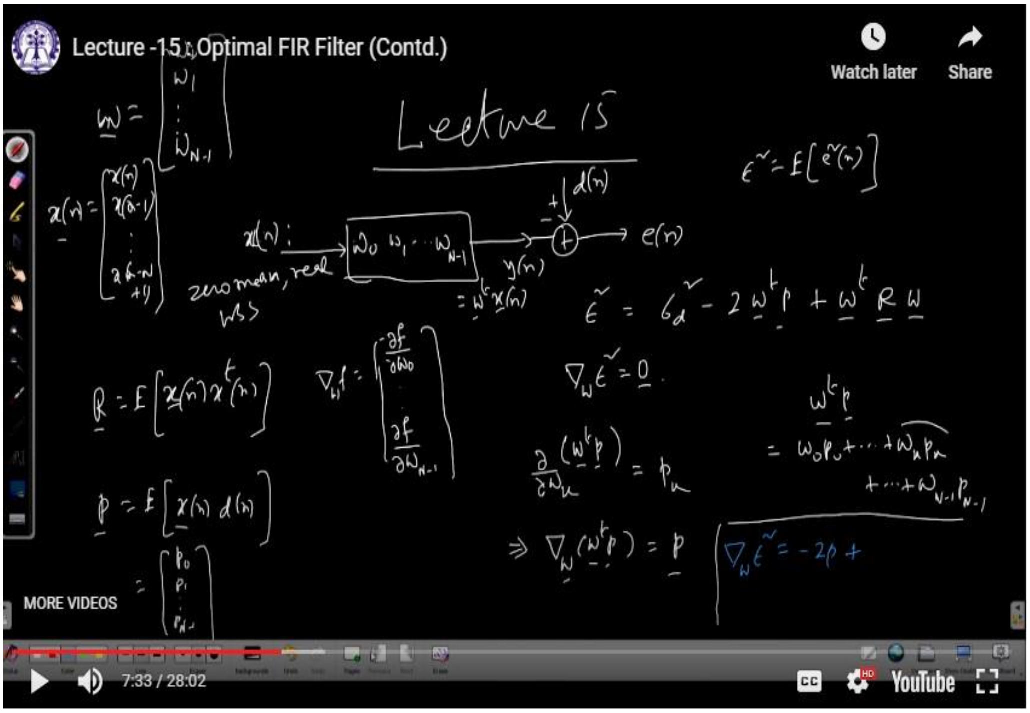
And r was though defined yesterday it is the input autocorrelation matrix it follows that toy fluid structure symmetric toy fluid structure and p was the cross-correlation vector E of this is xn, xn above this, xn will be here all the components of xn the xn vector times the scalar dN. So, it is xn and dN they are jointly stationary that is why this vector also independent of N like, capital R is independent of N because xn is jointly stationary xn is stationary Wss ok. So, then I defined this total derivative operator ok which is what that if you have a function f it is nothing, but you are taking the partial derivatives and stacking them in a column del f with respect to del W0 dot dot del f with respect to this ok. So, then each of the partial derivative of F standard square with respect to W0 Wn minus 1, Wn minus 1 will be 0. That means, if I have instead of f this equal to 0.

So, basically it will mean capital N number of partial derivatives which equal to 0. So, I will get and second order stuff derived will give us first order stuff will give rise to first order stuff. So, I will have capital N number of first order equations linear equations I solve them that is my game plan. Now to do this, first see if this sigma d square is differentiated with respect to any of the weights, I get 0 because sigma d square is independent of weights. So, this term I do not have to bother about W transpose P, W transpose P is what W0 P0 W1 P1 like that ok.

So, P is like you can name them P0 1 element P1 dot dot dot P N minus 1. So, W transpose P we know it is W0 P0 dot dot dot dot general term maybe Wk Pk and lastly W N minus 1 P N minus 1. So, if I take the partial derivative of this with respect to any of the weights say del del Wk of this W transpose P. So, right hand side if I find I find that only here I got Wk present no other term has Wk. So, other terms will disappear Wk Pk when differentiated with respect to Wk will give rise to Pk.



So, if I now stack the derivatives like del del W0 that will be P0 del del W1 that will be P1 this P0 P1. That means if I stack that is del W this term W transpose P it will be all this P0 P1 P2 like this which is P vector itself ok. So, one term will be minus 2 P. So, one term is minus 2 P, minus 2 W transpose P differentiated give rise to P so minus 2 P and then another term will come from here, but that requires more involved derivation which I will do now W transpose R W.



We have got this is nothing, but W transpose R W I take it together.

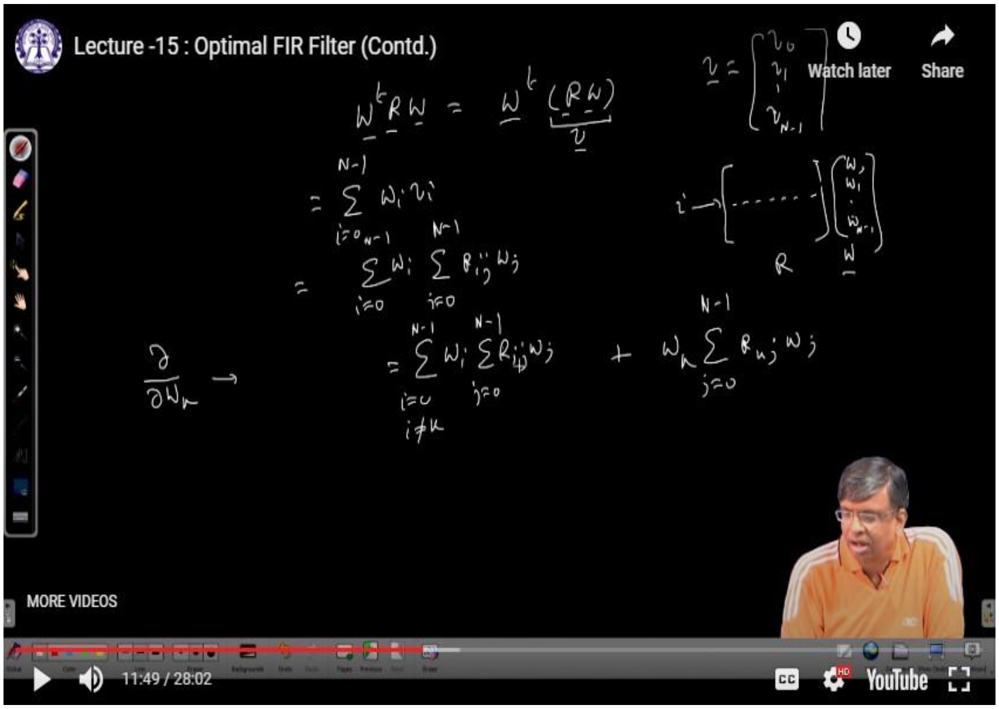
So, it is a matrix into a column vector. So, it is a column vector ok you can call it V. So, this is nothing, but W0 V0, W1 V1, W2 V2 and their summation. So, it is Wi Vi ok V is all the terms are indexed like this V is a column vector R W transpose V. So, we will have W transpose V row vector column vector if you do this multiplication, we will have W0 V0 W1 V1 and dot dot dot this what you have, but Vi now consider V alone Vi means what this is your R matrix and this is your W vector W vector means W0 W1 dot dot dot W N minus 1.

So, R into W will be a V vector ith component of that vector means if I is here that means you have to take this row ith row times this that will be the ith component Vi this ith row will be Ri 0, Ri 1, Ri 2, dot dot dot Ri N minus 1 ok. So, Ri 0 W0 plus Ri 1 W1 plus Ri 2

W2 like that. So, this product Ri you bring another term j and Wj Ri j will be from 0 to N minus 1. So, Ri 0 W0, Ri 1 W1, Ri 2 W2 like that that it and this Wi will be here this is what it is. So, suppose I want to apply just this partial derivative del del Wk for some k, a general weight I have taken Wk, k could be 0 1 up to N minus 1 on this this what I have to apply on this.

So, once Wk has been fixed by you my next step is first in this summation i goes from 0 to N minus 1. So, 0 1 2 dot dot dot N minus 1. So, it will become i equal to k also once that case I separate out separate out on this side Wk. Wk that is i equal to k and Ri equal to k. So, Rk j Wj you understand i equal to k case. So, Wk where i is k and then again put k here i is k and so k is a Wj the same summation that is i equal to k case I separately write.

And then here I write i equal to 0 to N minus 1, but I also write i not equal to k. So, it goes from 0 to N minus 1, but it skips kth case and Wi inside there is no change whatever i you choose that i comes here. Now if I differentiate this with respect to Wk you remember outside I have W0 W1 up to WN minus 1, but never I have Wk. So, this is like a constant because it is not Wk inside this summation for every Wi inside the summation I have Ri 0 W0, Ri 1 W1, dot dot dot Ri k Wk. So, there is one Wk inside Ri k Wk and other terms.

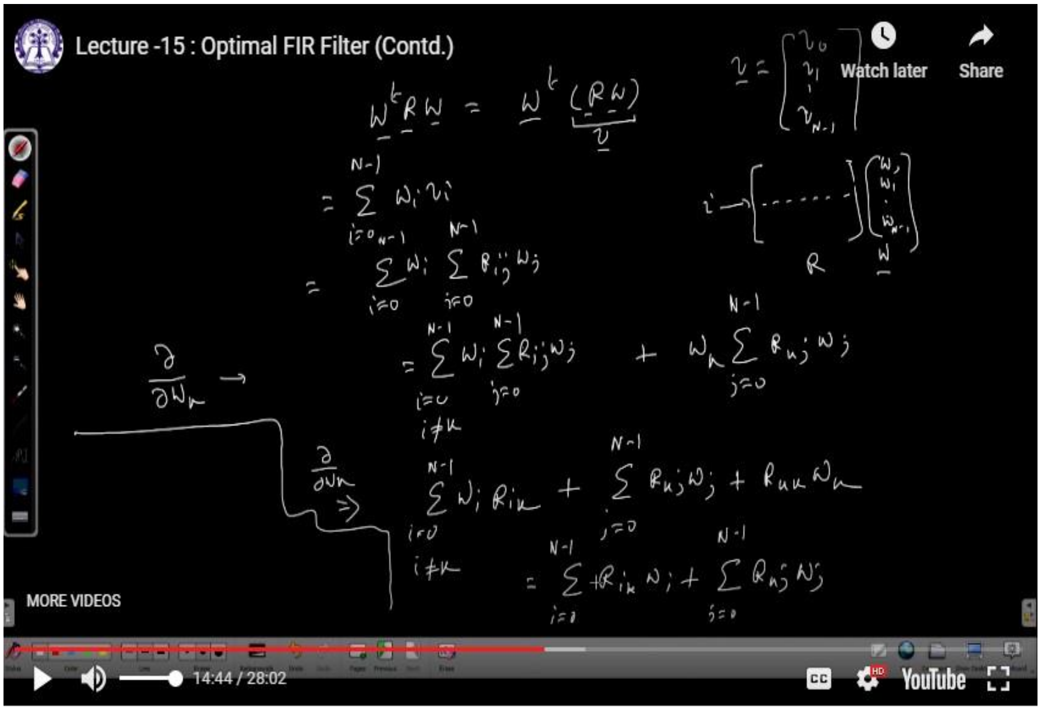


So, if I differentiate these with respect to Wk I will get Ri k others go they disappear because they are not, I mean no Wk present there. So, Ri k Wk when I differentiate that particular term in this summation with respect to Wk I get Ri k. So, this summation give rise to upper derivative that is if you do this it gives rise to summation Ri k, i equal to 0 to N minus 1, but i not equal to k. And now here I have got Wk here also and inside the summation j goes from 0 to N minus 1. So, j will become k once that time I have Wk or kk.

So, Wk and this. So, it is a product of 2 this one and this one. So, you know the differentiation rule I will first differentiate with respect to this without changing this and then differentiate this without changing this ok. So, if I differentiate this with respect to Wk I get this term. And now if I hold this fix Wk if I differentiate this with respect to Wk I get Rkk Wk differentiated Wk goes Rkk. So, Rkk and Wk fixed here.

So, Rkk Wk. Now you see i equal to k is not present here. Suppose i equal to k was to be present then that would have been a term Wk Rkk and that is what it is Wk Rkk. So, I can as well absorb this in the summation and remove this i not equal to k ok. So, it will be R ik

Wi and i from 0 to n minus 1 no longer i not equal to k that is absorbed in summation and another term R kj Wj and j equal to 0 to n minus 1 ok. Then R is symmetric matrix Hermitian real Hermitian so symmetric.



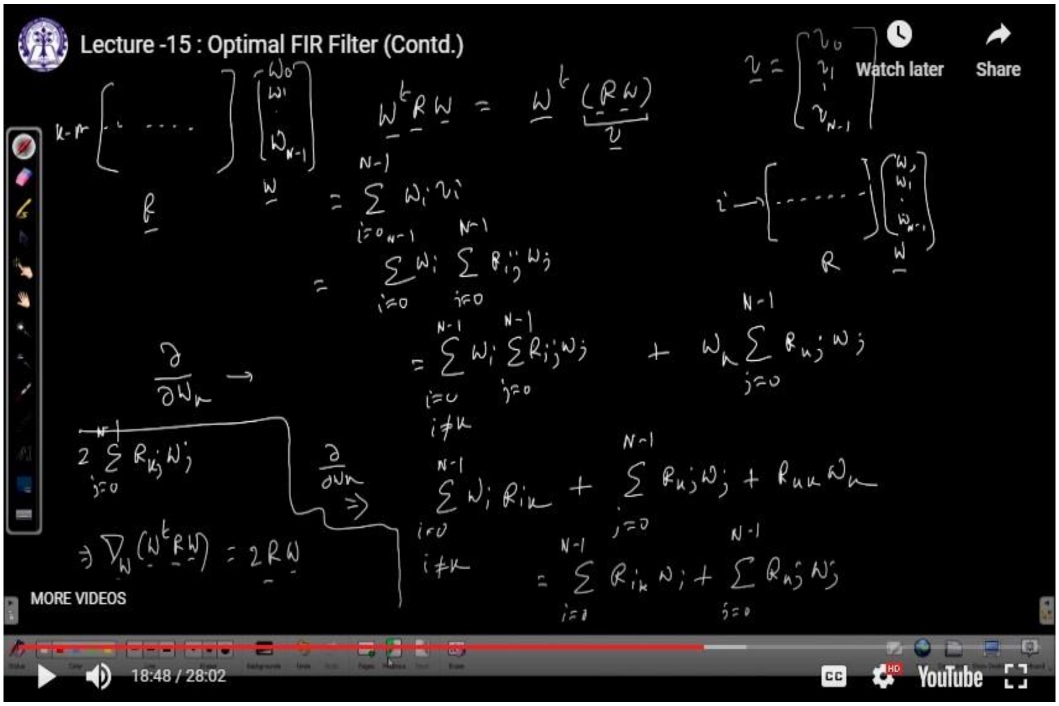
So, R ik is same as Rki. So, Rki Wi and Rkk Wj both j and k j and i are from 0 to n minus 1. So, the two sums will be same whether you put j here you can change the index from j to i you still have the same number of term same terms Rk 0 W0, Rk 1 W like that. So, if you change the index from j to i here in the summation it does not alter anything you are covering the same range. So, Rk 0 W0 when i equal to 0, Rk 1 W1 and like that.

So, no change whether you are using j as the index or i to replace it. So, I replace j by i here and R ik is same as Rki. So, I will have Rki Wi again Rki Wi same summation. So, it will be it will give me i should twice Rki or maybe instead of replacing I, j by i I replace i by j ok. I am just fonder of that replace i by j.

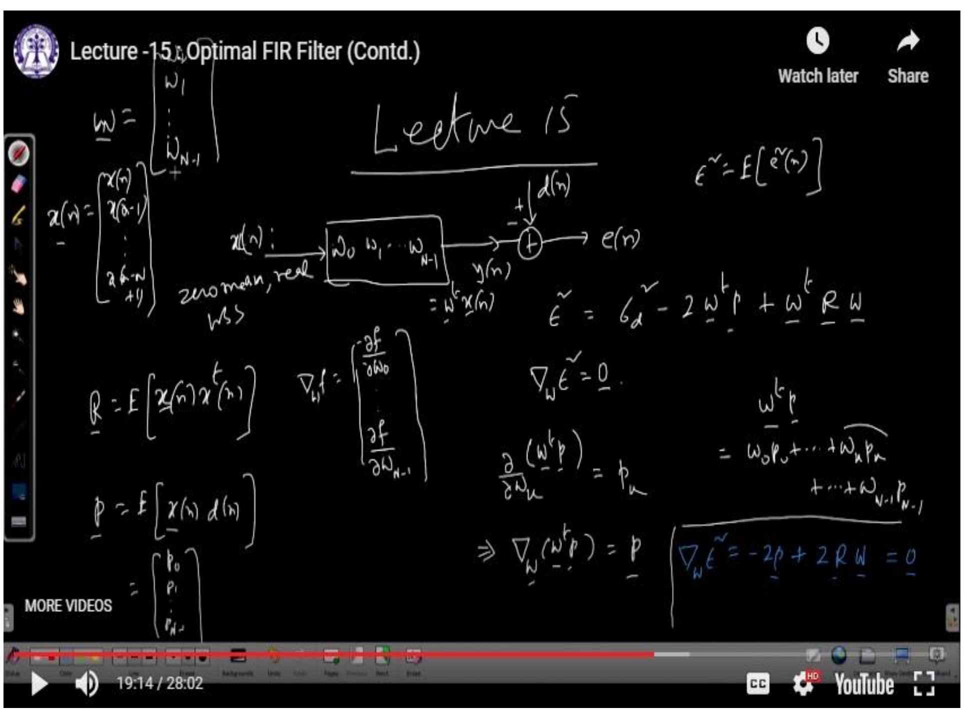
So, it will be Rk j Rk j Wj R ik is Rki and then replace i by j. So, Rk j Wj same summation. So, j equal to 0 to N minus 1 ok. Now, look at this summation Rk 0 W0 Rk 1 W1 like that what does it mean that if I have the R matrix and kth row and then W vector. So, I have got kth row Rk 0 W0, Rk 0 W0, Rk 1 W1, Rk 1 W1 and so on and so forth.

So, basically it is nothing, but kth row times this vector that will be the derivative of this W transpose Rw with respect to Wk. So, if I want to differentiate W transpose Rw with respect to Wk I have to target the kth row and then this product. So, when it is W0 I have to target the top row when it is W1 I have to target the second row. So, top row times this that will give me derivative of this with respect to W0 next row times this that will give me derivative of this with respect to W1 and dot dot dot. So, if I want to put all the derivatives and stack them it will be just this resulting vector R times W this row times this will be if you carry out the product Rw this is the top element, but what is the top element this times this will be the derivative of W transpose Rw with respect to W0.

So, del of this del W0 will come in the first guy then next row times this that is same as from this theory derivative of this with respect to W1. So, this matrix into vector if I carry out the product next guy from top will be derivative of this with respect to W1 and dot dot dot. So, derivatives are stacked that means, if I put if I now take all the derivatives and put them in the stack of this quantity W transpose Rw it will be nothing, but there is a 2 and R and W all right. So, 2Rw.

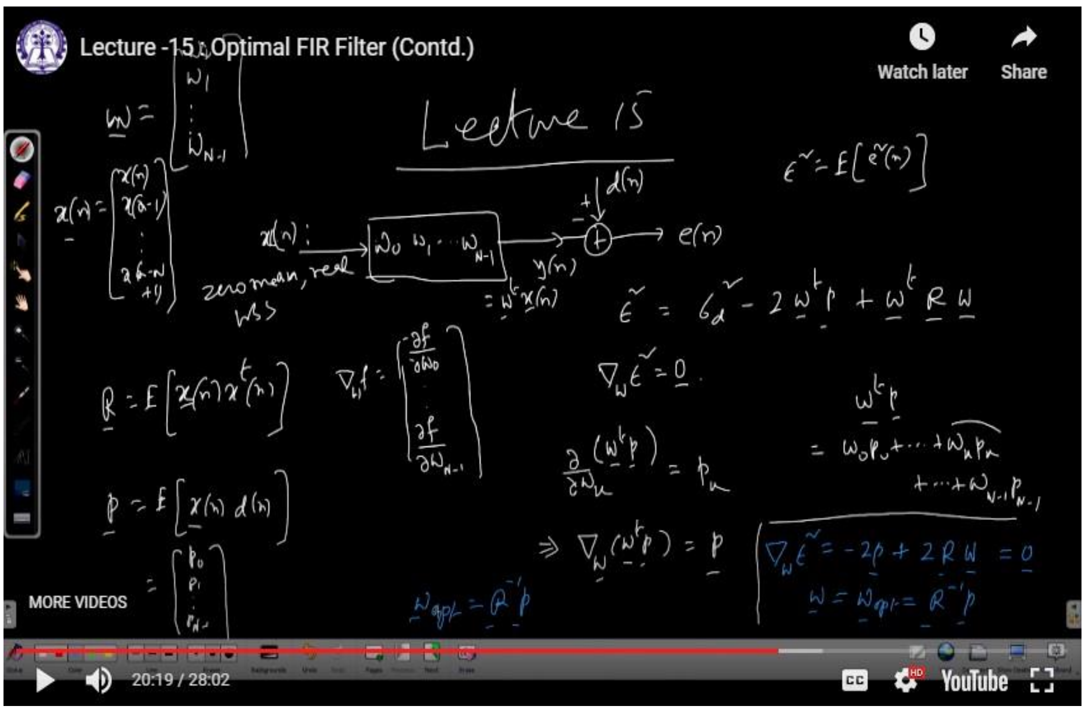


so, going to the previous page the other term is 2Rw and for minima every derivative should be 0. So, this should be equal to a 0 vector. This is a stack of the derivatives partial derivatives each partial derivative of epsilon square with respect to W0 W1 and this should be 0.



So, all the derivatives are stacked here this is the expression of the derivative. So, that should be equal to 0 vector. So, now then that means, 2 2 cancels you can take minus p to the right-hand side Rw is p which means W which is the optimal weight down is R inverse p. R is invertible because we are assuming R to be positive definite not just Hermitian or symmetric it is positive definite, which is a very standard assumption and this we discussed at length earlier. So, this is that optimal filter expression R inverse p ok.

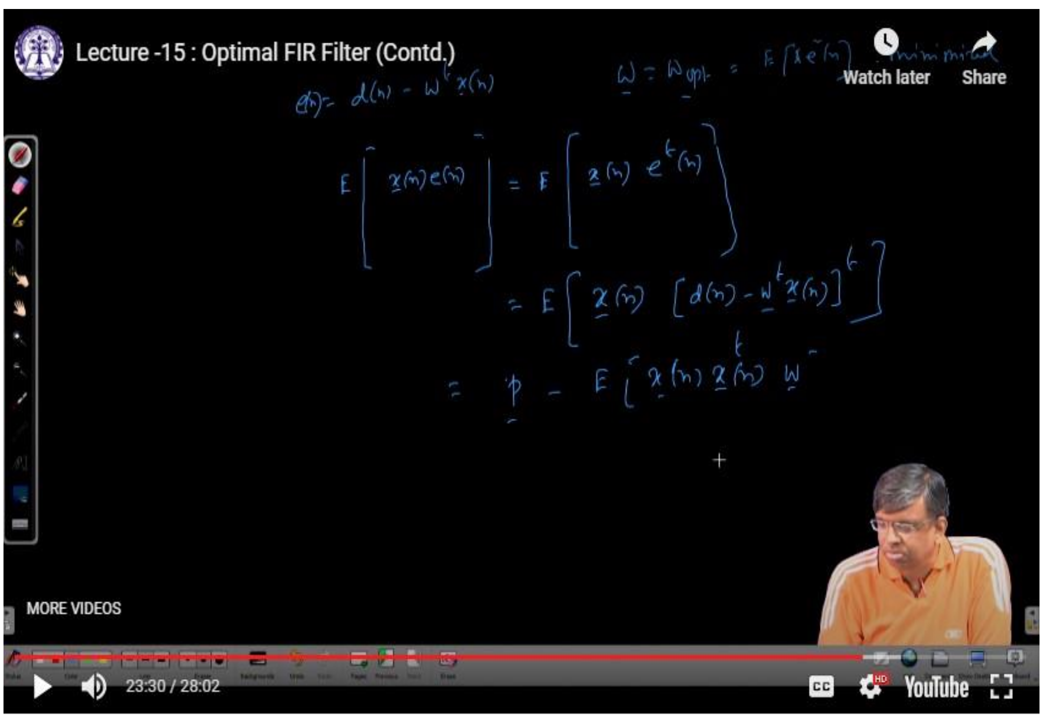
So, W opt I write separately is R inverse p ok. then only this error power error variance expected power is minimized. If I choose this coefficient of this vector W opt here W0 W1 and Wn minus 1 if I choose from this vector all right.



One thing you see if I know that error e is dN sorry en is dN minus W transpose xN all right. At W if you choose W opt then this is minimized, e square N minimized ok W opt. Now, if I have to find out correlation between every component of xN vector that is x of N, x of N minus 1, x of N minus 2 dot dot dot with en ok then what is the result. So, I take expected value of xN vector into en.

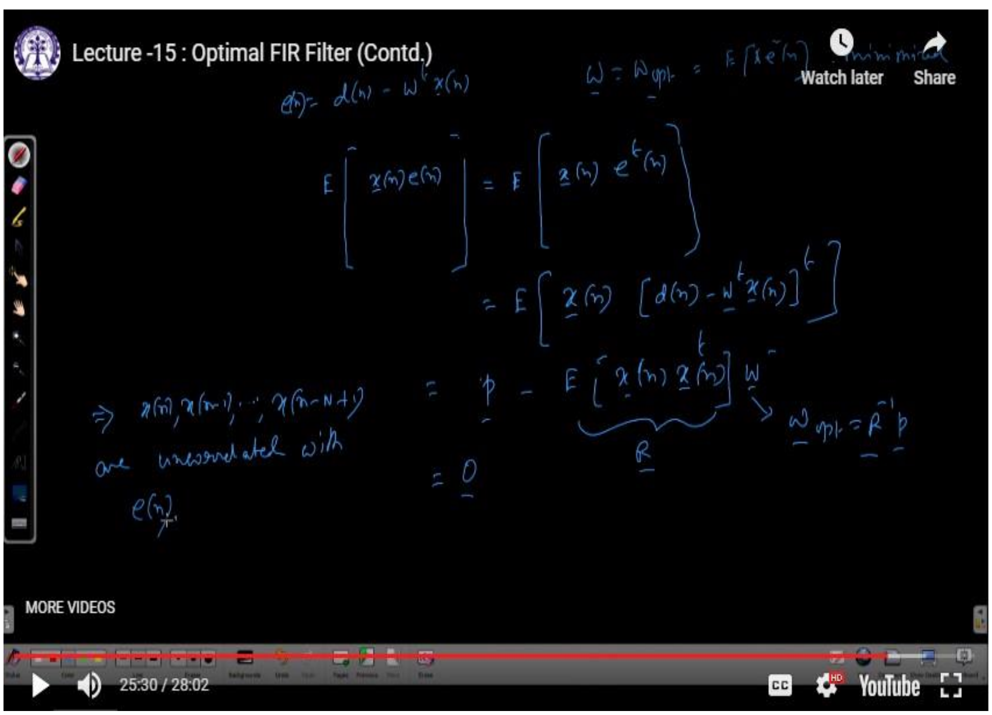
So, if you write the xN vector fully here then first term will be x of N into en. So, correlation between x of N into en expected value then xN minus 1 en another correlation between xN minus 1 to en, then xN minus 2 into en correlation between xN minus 2 to en and like that. So, let us try to calculate this then en is a scalar. So, scalar is like a 1 by 1 matrix. I can write it also as e transpose N because for a scalar which I can view as 1 by 1 matrix its transposition is itself only one row one column. So, en is same as e transpose N if I substitute that e transpose N here that is e e transpose N that is transpose.

So, xN dN, dN transpose is same as dN. So, E of xN dN transpose that is x of dN, xN dN that is my p vector minus E of xN and transpose of this will be like A matrix B matrix AB transpose is B transpose A transpose. So, it will be xN and then xN transpose W transpose transpose that is W.



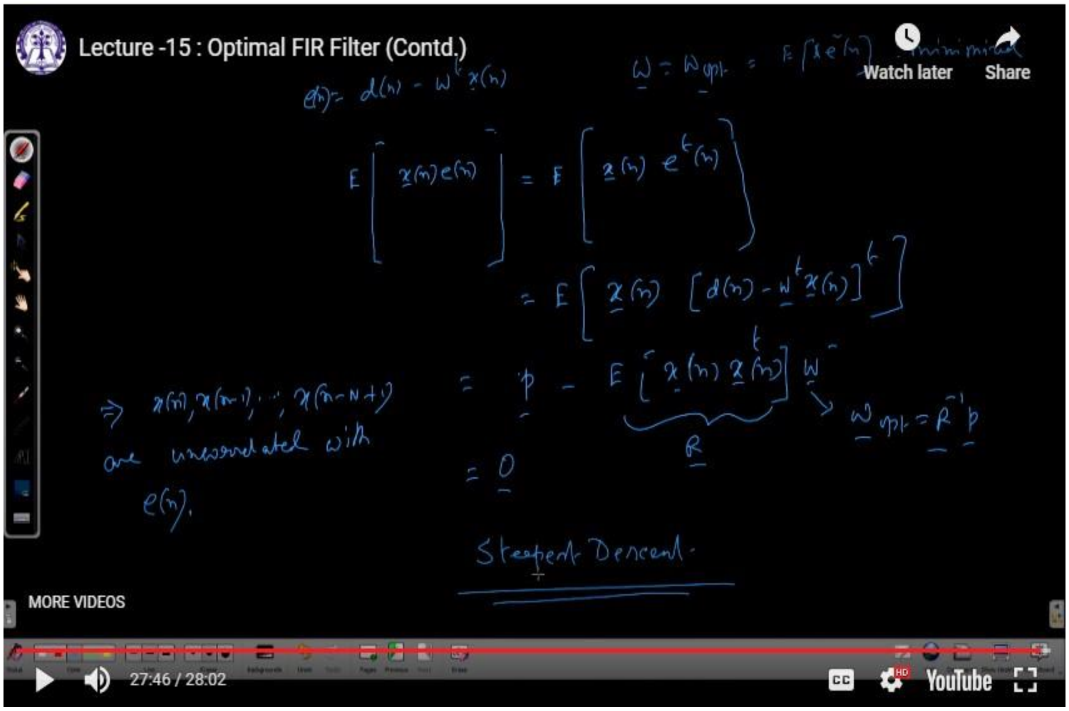
Another day I told you since this is the random part and this is constant you will get the same thing if you take W out and apply the E on this matrix xN into x transpose is a matrix column vector row vector. So, if you apply E on every element of the matrix and then carry out the product you will get the same thing if you do not do that if you do this xN into x transpose in this matrix multiply this vector by the matrix then apply E. E will not work on the elements of this you will still work on the elements of this product matrix xN into x transpose will get the same thing.

So, I can put it here and this is my R and this W is W opt that is when I use W opt which is R inverse p. So, R R inverse p, R and R inverse cancels identity matrix identity matrix times p is p and p minus p is 0 which means every element xN, xN minus 1 dot dot dot are uncorrelated with En this is an important result all right. Now, we have seen that optimal filter depends on two things R and p, R matrix and p vector, R matrix is a input autocorrelation matrix. Now, that may not be fixed always because input statistics can change depending on circumstance like say one person is speaking his voice has some kind of I mean autocorrelation and the person changes or even the words spoken by the person changes from you know time to time ok. So, autocorrelation matrix changes p vector changes like that.



So, if they keep changing then I have a problem because if they do not change I know R matrix I know p vector R inverse p I calculate offline. So, I get the if IR filter coefficients W opt vector in that vector. So, constructed if IR filter forever construct the hardware forever using those coefficients and carry on filtering, I will be very happy. But if R matrix and p vector keeps changing from time to time then right now maybe I have an optimal filter for the corrector correct p, but after sometime this will not be the optimal filter because R has changed p has changed point is can I like a mad guy every 10 seconds or every 10 minute or whatever you know from time to time recalculate W opt then break up destroy I mean break up my hardware and reassemble my hardware again running for some time I cannot do that nobody can do that. Rather better will be if the filter has an learning mechanism or training mechanism by which it can learn about the changing statistics of input and cross correlation and you know I mean autocorrelation things from the data itself and it adjusts the filter coefficients you know clock to clock to clock nth clock to n plus 1th clock to n plus 2th clock.

So, that finally, filter weights again converse to new value of R inverse p because R has changed p has changed. So, W opt has changed. So, that error pressure mechanism will make sure that the coefficients now again change more adapting clock after clock, but finally, they go to again this optimal one which is a new optimal because R is now new, p is new and it will continue. Such a filter is a filter of our interest and that is called adaptive filter. Now to go to adaptive filter there is an intermediate step where we still obtain this optimal filter only, but not by this method of R inverse p by an iterative method called steepest descent.



We will move to this first this is an intermediate step to get at the adaptive filter belonging to least mean square LMS family. So, we will cover that in the next class. Thank you very much.