Introduction To Adaptive Signal Processing

Prof. Mrityunjoy Chakraborty

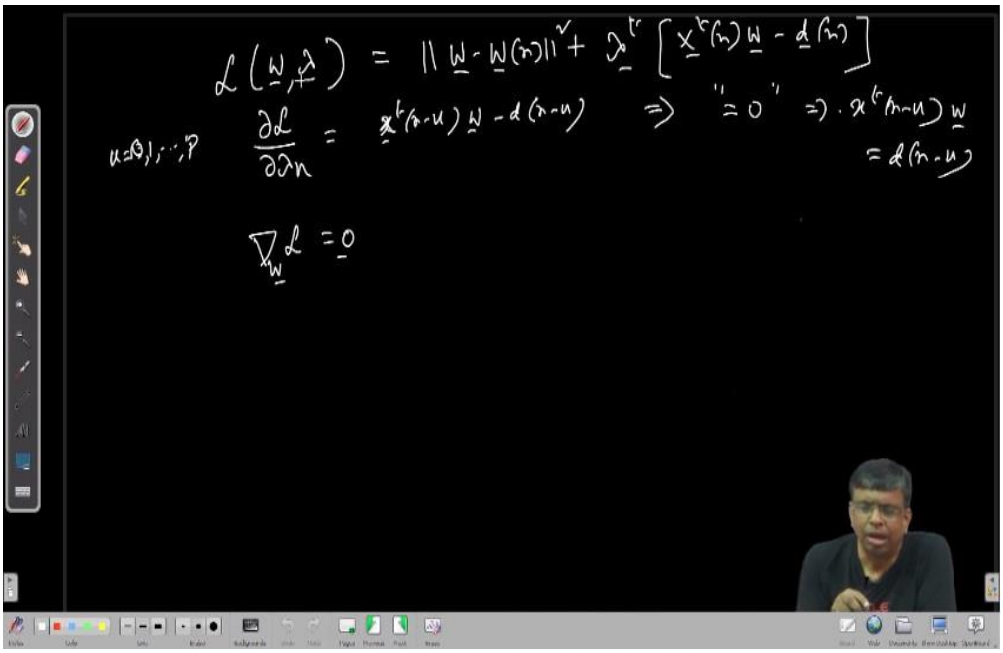
Department of Electronics and Electrical Communication Engineering

Indian Institute of Technology, Kharagpur

Lecture No # 31

Affine Projection Algorithm (APA)

So, in the last class we are carrying out the derivation of affine projection algorithm. So, this was the page we had a generalized version generalized function which was a function of all the weights w0, w1 up to w capital N minus 1 and also lambda is a vector is a function of lambdas also lambda 0, lambda 1 up to lambda p because there are p plus 1 constraints ok. So, each constraint has one Lagrange multiplier and this was the thing all these were derived last time. So, I will not be wasting time in redoing this or restating this ok. This is a function we have to derive it with respect to each lambda k and we have to differentiate it with respect to each w ok.

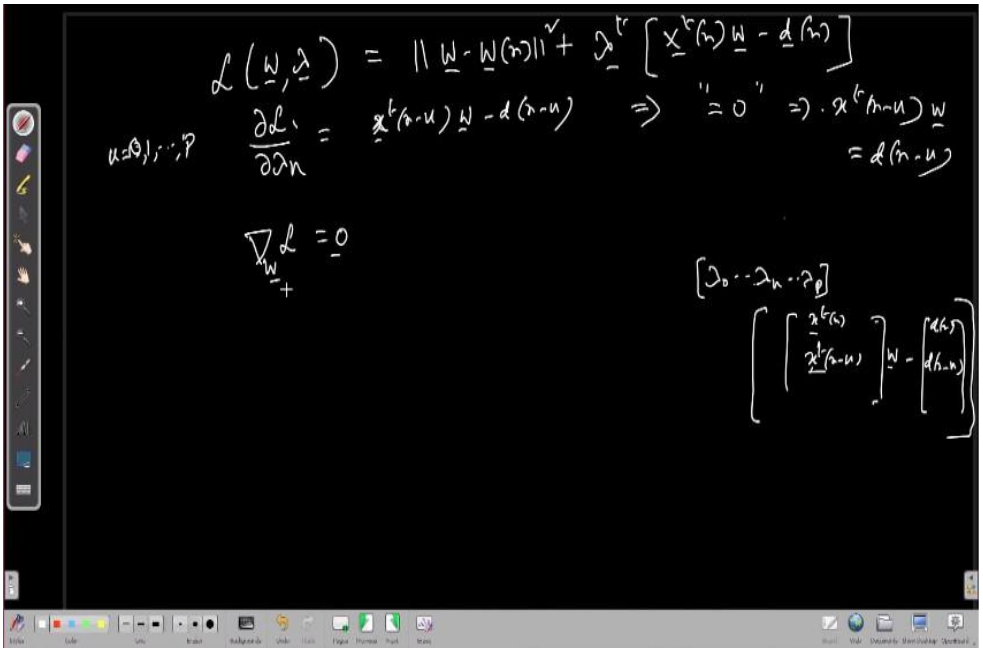


If I take any partial derivative of this with respect to lambda k lambda k figures here ok. So, actually it will be this you can see x transpose N d w ok. So, only this and lambda transpose. So, this lambda 0 dot dot dot lambda k dot dot lambda dot lambda p and this thing.

So, lambda k will multiply this x transpose N minus k vector. So, it is like this x transpose N w. So, rho vector into w column vector minus dN this is k that will be multiplied by lambda 0, but there is no lambda k there. Lambda k will multiply this term x transpose N minus k this rho times w ok. That is there is a filter output at N minus kth time index with the filter coefficient vector w.

So, this output minus dN minus k alright this is the thing this will be multiplied by lambda k. So, when you take a partial derivative with respect to lambda k only this will result that is what we had obtained earlier ok and you equate that to 0. So, you get this. So, by this partial derivatives with respect to lambdas I get this alright this equation. So, this will be satisfied.

So, therefore, because whatever be the wN whatever is the w that must satisfy those p constants by forming this generalized function and introducing the Lagrange multiplier we see when you differentiate this L with respect to lambda k's those conditions get satisfied right. That is if you want to find out minima of this function you have to take minima with respect to both lambdas and w's. So, there it shows that that minima will will automatically satisfy those constants these are the constant equations ok. For k equal to 0, k equal to 1, up to k equal to p. So, those will be satisfied.



Now comes the other derivative. Here w minus now L is if you take this w minus wN norm square we have seen in the context of LMS this is minus the two cross terms are same. So, twice w transpose wN plus norm I mean you can write either way either in terms of norm or just this w transpose w is norm square of w. Similarly, w transpose N wN is norm square of wN I am writing like this you could write as norm square also. So, this is the part and then you have got that extra part right.

$$L(\underline{w}, \underline{\lambda}) = \underline{w}^t\underline{w} - 2\underline{w}^t\underline{w}(n) + \underline{w}^t(n)\underline{w}(n) + \lambda^t[\underline{X}^t(n)\underline{w} - \underline{d}(n)]$$

So, therefore, if I take this derivative this function w transpose w we have already seen in the context of N LMS algorithm. What is w transpose w? W0 square plus w1 square plus dot dot wk square plus dot dot like that. Any partial derivative with respect to wk will be twice wk. So, if you put all the partial derivatives in a stack twice w0 twice w1 twice w2 dot dot this is 2 if you take common it is 2w. So, this term will give rise to 2w here twice w transpose wN that means, w0 w0 N, w1 w1 N, dot dot dot dot.

So, if you differentiate with respect to the particular weight say wk. So, it will be this twice

the term is twice wk wk N. So, wk goes because of differentiation will be twice wk N. So, if you now stack then this derivative. So, twice w0 N twice w1 N twice w2 N dot dot dot which will be twice wN this has no w.

So, after differentiation it goes here lambda transpose x transpose w minus lambda transpose dN lambda transpose dN has no w. So, that will go away because of derivation lambda transpose x transpose N w it is nothing, but xN lambda transpose w. Because you know from basic matrix theory any matrix if you have A into B transpose is B transpose A transpose lambda may be a vector, but it is also matrix is not it P plus 1 cross 1. So, xN lambda transpose which is lambda transpose x transpose N ok. So, that times w.

$$\underline{\lambda}^t \underline{X}^t(n)\underline{w} = \left(\underline{X}(n)\underline{\lambda}\right)^t \underline{w}$$

So, if you call it if you give it any name say may be alpha. So, alpha transpose w and now if you differentiate it with respect to each component of w and put those results in a stack you will get nothing, but alpha vector like we get wN vector here all right alpha vector w transpose I am sorry I wrote wrongly here no it is ok this is fine this is fine. This term I am talking of twice w transpose wN and that is same as twice w transpose N w twice w transpose N w. So, w transpose N w when differentiated with respect to w get wN instead of wN I have alpha. So, alpha transpose w when differentiated with respect to w put in a stack all the derivatives it will be alpha like it was wN there.

So, it will give rise to alpha, alpha is this which means this derivative will be lambda transpose sorry alpha is xN lambda. So, it will be xN lambda. Now, remember it is a vector equation not a scalar equation xN is a matrix lambda is a vector lambda 0 lambda 1 up to lambda p and this is also a gradient vector all right. This is the situation this I have to equate to 0 vector for minimization for minima all right. That means, if I take this to the right hand side what I have is twice this xN lambda lambda vector ok.

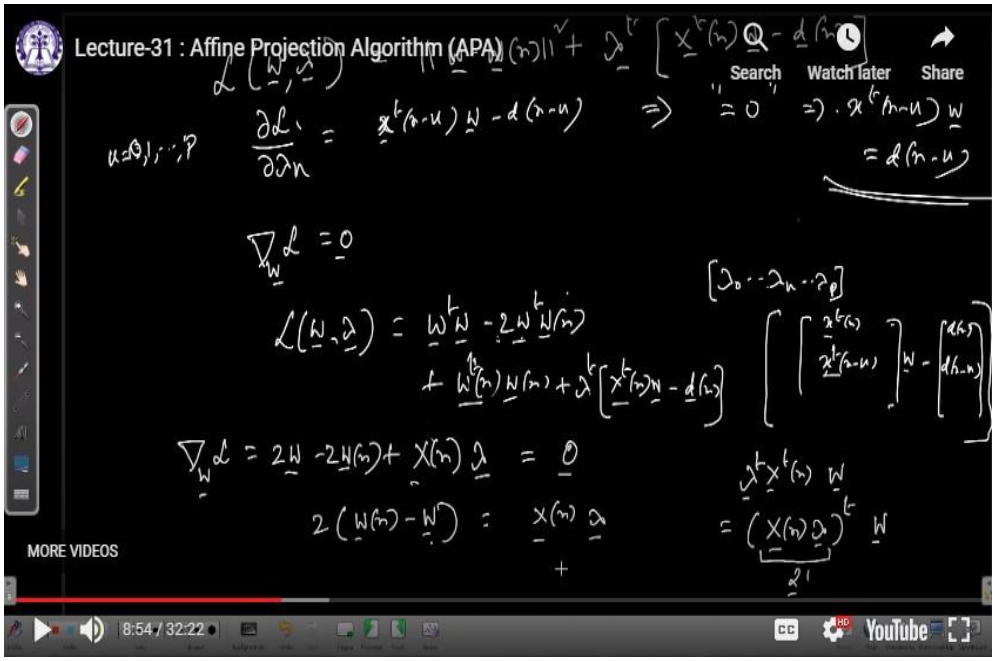
Now, if I multiply pre multiply both sides by x transpose N that is twice 2 remains same. So, 2 is just kept in the front then x transpose N times this gradient vector here also x transpose N xN lambda all right.

$$\nabla_w L = 2\underline{w} - 2\underline{w}(n) + \underline{X}(n)\underline{\lambda} = \underline{0}$$

$$2\left(\underline{w}(n) - \underline{w}\right) = \underline{X}(n)\underline{\lambda}$$

Now, this matrix will if this matrix is invertible x transpose N xN it is a square matrix Hermitian matrix I will talk in detail about this. So, if this is invertible your lambda will be just you know inverse of this will come here twice inverse of this then x transpose N then this all right ok. So, let me go to the next page.

$$2\underline{X}^t(n)\left(\underline{w}(n) - \underline{w}\right) = [\underline{X}^t(n)\underline{X}(n)]\underline{\lambda}$$



Let us study this matrix x transpose N xN what was our xN? It had a vector xN vector then xN minus 1 vector dot dot dot xN minus how many p xN was the filter input data vector at Nth clock. So, it was x of N if size is N cross 1. So, this matrix is N cross p plus 1 ok. If

the columns are linearly independent that is there is no linear relation between them that is you cannot write any column as a linear combination of the others ok. Then we say it is full rank full column rank ok.

In that case x transpose N xN will be invertible how we have coming to that consider this matrix x transpose N xN. So, this is Hermitian. So, obviously, if you call it A A transpose is again if you take the transpose of it you get x transpose N this will get transpose come here and x transpose N will come here upper transpose. So, transpose cancels ok. So, A is A transpose Hermitian not only that it is positive semi definite always why because if you take for any vector c not 0 c not 0 transpose c transpose A c it will be we can see that it will be non negative why because c transpose you replace A by x transpose N xN c xN into c it is a matrix times a column vector you call it d.

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

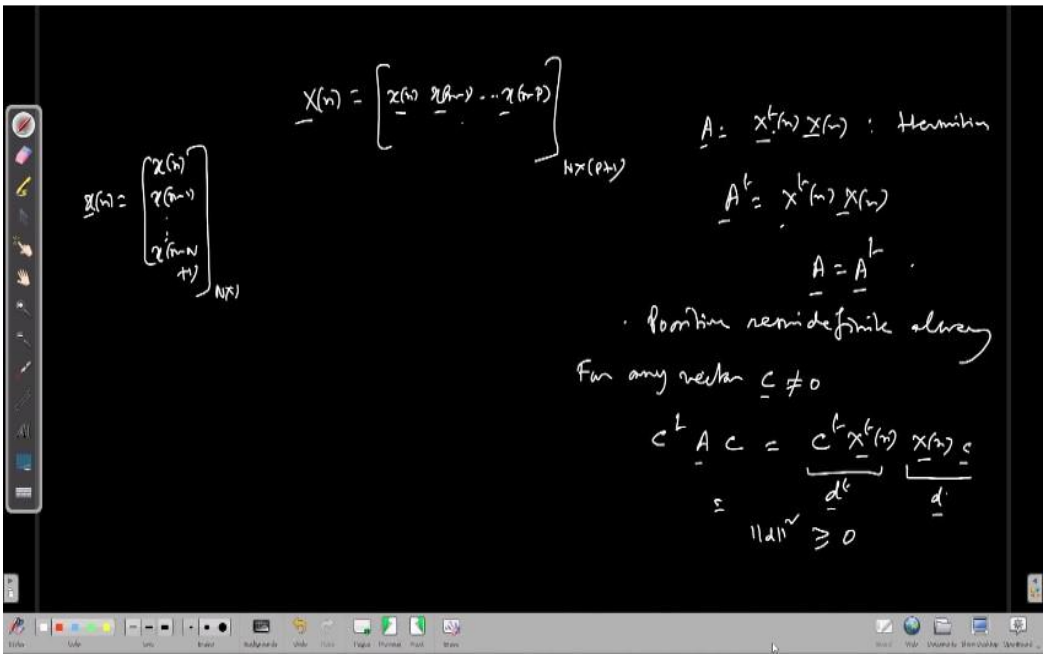
$$\underline{A}^t = \underline{X}^t(n)\underline{X}(n)$$

$$\underline{A} = \underline{A}^t$$

So, d is a column vector and what is this? This is xN c transpose this will be d transpose is it not if you take d as xN matrix into a c and c is a non-zero vector. So, xN into c if you call it d vector if you get transpose of d it will be c transpose x transpose that is what you have. So, d transpose and d transpose d is norm square of d. So, norm square can never be negative. So, it is always non negative ok that is always true.

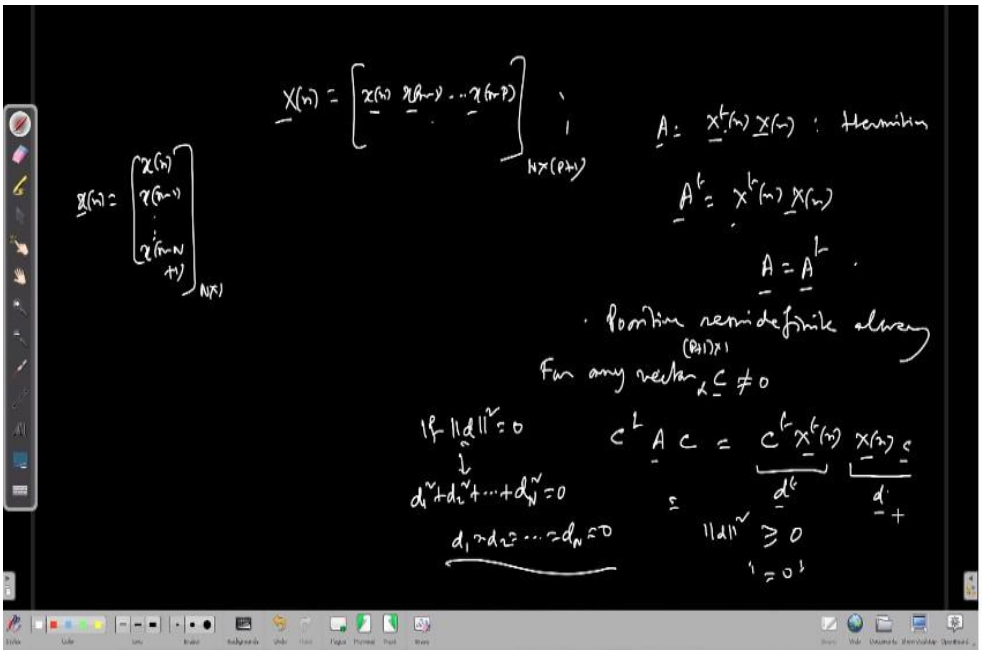
$$\underline{C} \neq 0$$

$$\|d\|^2 \geq 0$$



But if it is not only greater than equal to 0 if it is strictly equal to 0 strictly greater than 0 then it is positive definite then we have seen in the very beginning of the lecture positive definite matrices have positive eigenvalues and they are invariable. Now when this equal to 0 occurs equal to 0 it will be 0 if now what is this norm of d square? Norm of d square is d1 square d2 square d what? xN into c ok. So, it will be N cross 1 vector xN is N cross p plus 1 you are multiplying by c ok c is p plus 1 cross N for any vector let me write. So, p plus 1 entries in a vector c you are multiplying by xN there are p plus 1 columns. So, resulting vector will be length N vector right that is why.

So, d is length N vector norm square of d is this if this is equal to 0 since every term is positive or 0 no negative no term is negative every term is either positive or 0 positive or 0 and they are adding there is no subtraction. So, they can only add they can only contribute positively. So, the whole thing has to be 0 every term has to be 0 d1 square has to be 0 d2 square it cannot happen that this is also positive this was a positive, but there is a minus sign here. So, they cancel that is not there it is all positive signs additions only. So, if the sum total is 0 and every term is non negative there is either 0 or greater than 0 the only possibility is everybody 0 right only then it is.


Hermitian
Positive semidefinite always
If
So, only if d vector is 0 vector only if d vector is a 0 vector then it will be equal to 0 it will be therefore, but d equal to 0 what does it mean d equal to 0 means xN into c0 is c is 0 this c c0 c1 let me write and xN what does this matrix times column vector what does it mean it is basically linearly combining the columns by this coefficients this actually c0 times xN vector c1 times xN minus 1 vector like that. So, c0 times xN vector c1 times dot dot dot cP times plus guy ok that is all 0 this means there is a linear relation among the columns suppose c0 is non 0 ok. So, I keep this term c0 xN on the left hand side take the others on the right hand side c0 non 0. So, divide both side by c0 that is possible because c0 is non 0 I am assuming. So, I will have xN right hand side will be minus c1 by c0 xN minus 1 minus c2 by c0 xN minus 2 dot dot dot minus cP by c0 xN minus P.

So, xN will be a linear combination of xN minus 1 to xN minus P. That means there is there is some linear relation amongst the columns we assume that no such linear relation exists we assume no such because it is all coming from random data there is no model inside. So, that the columns are always governed by an equation of that kind ok because the data is just randomly coming alright. I am giving xN data random purely random I am not giving data where there is a hidden linear dependence relation like that amongst the

data samples amongst the data samples. So, that the columns they become linearly dependent ok.

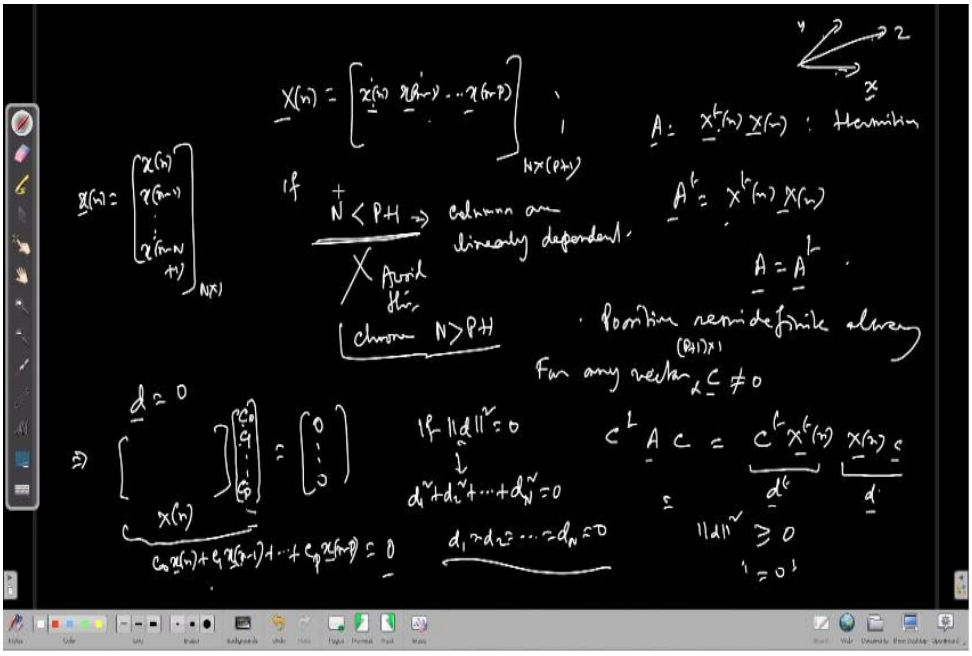
If that be then obviously d cannot be 0 because the moment d is 0 it implies this which means the columns are linearly dependent that is there is a linear relation between them. That is I mean that something we can always say it would not occur because I am giving xN data randomly there is no linear relation hidden linear relation amongst the data they are all coming independently you know because random purely randomly calculating. So, you can never have these columns generated by the data as linear dependence, but this may not happen if this number of row if number of row N becomes less than P plus 1 ok. Then columns are linearly dependent ok.

I will tell you why ok. That means there will be some linear relation between the columns and there you can by appropriately choosing c0 c1 to cp you can find ok a linear combination which is d that is xN times c d equal to 0 even if c is non-zero ok because of this linear relation that will always happen why? Because now every column vector is length capital N ok. So, every column vector lies in a space of dimension capital N. So, I can have maximum capital N number of axis which are called basis, basis vectors ok in an N dimensional space. So, you have more columns than the number of axis number of basis like suppose this is one axis this is one axis these two are fine equality x axis you call it y axis, but if you have another fellow here z in the same plane x y z ok that is dimension was 2 I had 2 vectors spanning this plane there if I pack one more fellow z then this set is linearly dependent because z I can always write as sometimes N times x something times y by the parallelogram law of vector addition. Similarly, every xN is length capital N.

So, they are belonging to a space of capital N cross 1 vectors that has dimension capital N. So, it can have maximum capital N number of axis that is basis any extra vector will be writable as a linear combination of this basis that is of this axis capital N number of axis. So, there will be a linear dependence relation in that case. Therefore, we should always avoid we should always avoid avoid avoid this that is choose ok. Then we can always argue

that the data which from these columns you know is such data is generated randomly from outside and there is no linear relation then you know governing the data.

So, that there will be some linear relation between these columns and all that that is a reasonable assumption, but if N is less than p plus 1 you cannot make that assumption because then every column vector is an N dimensional space, but p total column vector is p plus 1 which is more than N. So, we need capital N number of axis that is basis vectors are taken out ok. There are still extra vectors left they can be written as a linear combination of those x basis vectors those x in vectors ok. So, this is important. So, now this is the equation this is lambda for a minute this is lambda.



So, now, I am assuming x transpose x N is indeed positive definite and therefore, invertible. So, both left hand side and right-hand side I multiply by the inverse of this ok. So, I get lambda vector 2 x transpose N x transpose x inverse x transpose N wN minus w ok. So, this is lambda and the constraint that we had x transpose N w was equal to dN if I take you know all these partial derivatives and put in a stack we get we have discussed this at length earlier that this constraint will be satisfied it will be x transpose N w equal to dN.

So, I have got another thing that is x transpose N into w equal to dN all right x transpose N w is dN.

Then come to this equation 2 wN minus w is x N lambda, lambda I will put here 2 wN minus w is x N lambda ok.

$$2\left(\underline{w}(n) - \underline{w}\right) = \underline{X}(n)\underline{\lambda}$$

So, lambda you put here. So, these two comes x N this x N x transpose N x N inverse then this x transpose N wN minus w all right.

$$= 2\underline{X}\,(n)\left(\underline{X}^t(n)\underline{X}(n)\right)^{-1}\underline{X}^t(n)(\underline{w}(n) - \underline{w})$$

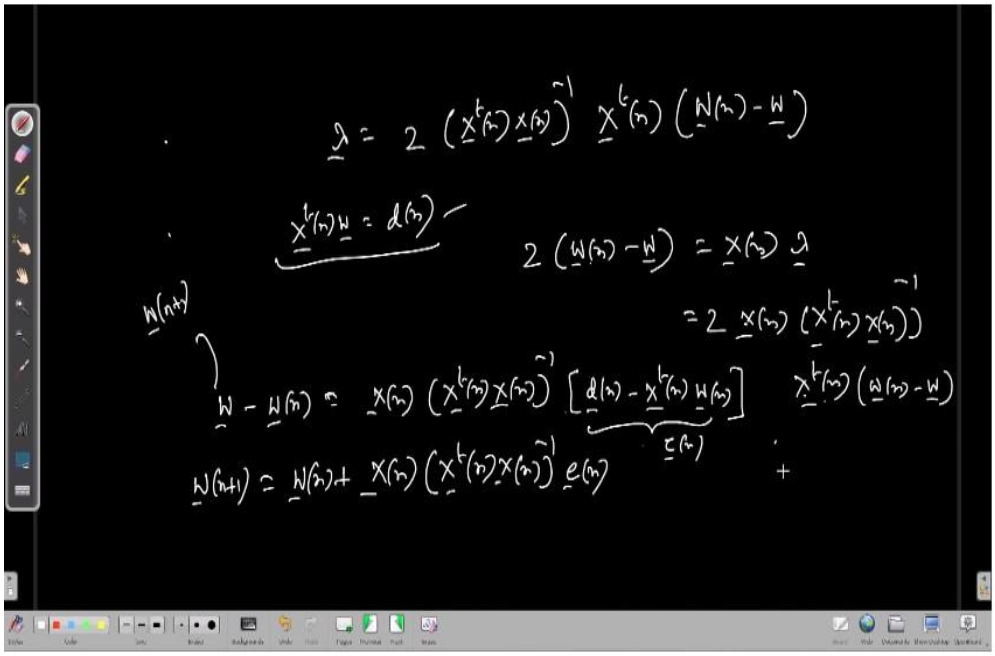
So, from here we will find out we have to I mean apply this constraint in this I see the 2 cancels 2 and 2 cancels right.

So, x transpose N w equal dN. So, x transpose N w will be dN. So, what if you have that you will have 2 2 is already cancelled, I take a minus negative of both side 2 and 2 cancel then take negative of both sides. So, w minus wN remember this w will be by wN plus 1. So, w minus wN is minus and minus sign I will observe here. So, do not need to write minus separately ok 2 2 cancels.

So, we have got x N now x transpose N wN minus will come on that and x transpose N w is this dN. So, it will be x transpose N w because I have taken minus of both sides. So, that will be positive this is negative x transpose N w is dN. So, this is my dN vector minus x transpose N wN all right. So, if then w will be what there is w there is that that I will equate as wN plus 1.

So, update equation will be this I am taking to the right-hand side plus xN this vector dN is a vector x transpose N wN that is also a vector this vector let me give a name en. So, this is my en vector ok. Now what is en? Now let us see en means dN minus this right. So, dN

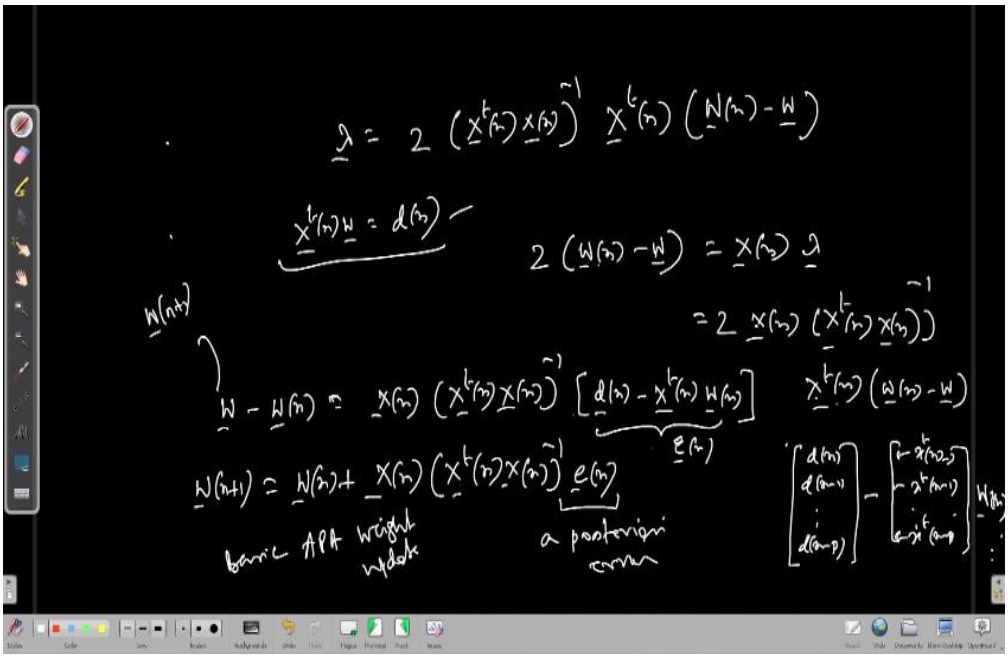
was dN dN minus 1 dot dot dot dot dN minus p and minus x transpose N was this was x vector then x N minus 1 vector all these were done earlier x transpose N minus p vector into w this is my en.



What does it mean that w instead of w now it is wN. So, wN is the filter coefficient vector at the Nth clock right. Suppose I hold it at Nth clock filter output will be x transpose N wN I subtract dN that from dN I get the output error. But now suppose I use the same filter coefficient, but take a look back I say that suppose I go back use these coefficients only, but use it in the previous clock N minus 1 that time my input data vector was xN minus 1. So, then filter output with these coefficients will be x transpose N minus 1 wN. So, that if I subtract from dN minus 1 that will be the next component of error.

You understand that filter coefficient is not changing at Nth clock whatever filter coefficient I have suppose I am freezing and I am taking back I am looking back that suppose I now use these coefficients only not wN minus 1, but this wN, but still at the previous clock ok at the previous clock that time my filter input vector was xN minus 1. So, filter output would be x transpose N minus 1 and this filter wN and that I subtract from dN minus 1. So, I get the corresponding error and same next is x transpose N minus 2. So,

if I use the same filter coefficients, but at go back and you know to N minus 2th clock then my filter input is xN minus 2 vector x transpose N minus 2 wN ok that is my filter output that I can subtract from dN minus 2 ok and this. So, this that is why it is called a posteriori error that is I have wN and now I am going back to previous clocks using that filter coefficient recalculating the output error ok that better vector.



So, this is the APA basic APA weight update formula. Some modifications will be done on this as before. So, that I will do in the next class. Thank you very much.