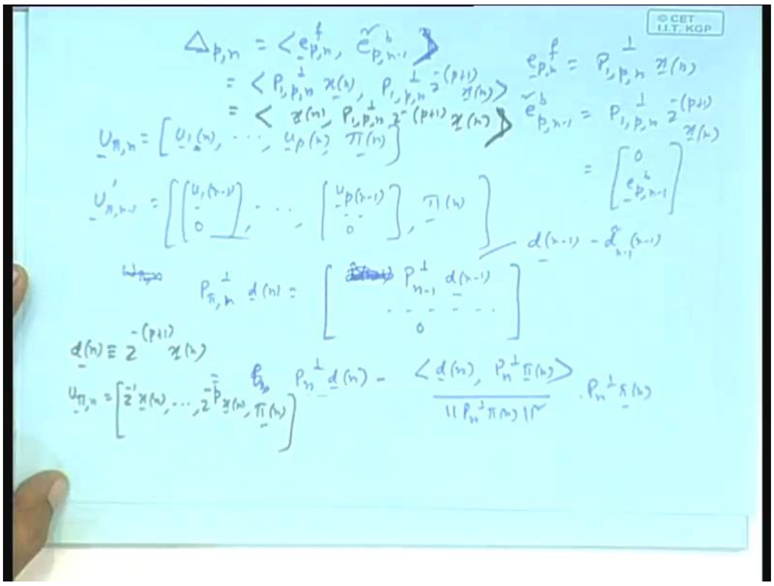
Adaptive Signal Processing
Prof. M. Chakraborty
Department of Electrical and Electronic Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture - 34
RLS Using QR Decomposition

(Refer Slide Time: 01:09)



Our purpose is to update this time up to this quantity. Remember what was this quantity? This was the inner product between this and sorry, this is and you know the definition of this. In this case, forwarded e p n f was p 1 p n perpendicular x n error vector pth order, and e p n minus 1 b tilde. This is actually p 1 p m same operator which is actually 0. Isn't it? This you remember right? Here how the 0 comes because all these follows, the columns of the corresponding matrix and all these follows. They all have 0 at the top first row. So, in the sum of square it up, that 2 does not I want to do anything.

Other rows you add up and I mean you get the optimal filter coefficient for the pth order backward predictions error z minus 1 nth index. So, when you combine the columns, the top follow again becomes this. Other follow becomes the projections and then, take the error and projections errors and all that we have to update this last time. I mean I just quickly go through. Last time we worked out one relation. What is that relation? That is suppose, there are some columns given u 1 n dot dot dot and pi n. This 1 matrix I do not know. I do not remember this 2, but they may get to it. Forget it. There I introduces this matrix u pi n something isn't u pi n, and there was another matrix U prime U prime pi n n minus 1 pi n minus 1 which was this u 1 n minus 1 0 u p n minus 1 0 and pi n, and I say it I prove and I will not re-prove it again that the space spanned by the columns are same. There is column space of the two matrixes are same.

Therefore, if I have any external vector d n, I want to project it orthogonally on the space spanned by this follows. Column space I give a name. I think is w pi n. Isn't it? See if I want to take any vector d n and project it orthogonally, and w pie n, I can take this also the space spanned by this follows and project it there, but there pi n is orthogonally 2. Sorry pi n is here. Pi n is orthogonal to each of this column because pi n has the only 1 1 1 at the end last element and other element is 0, and all these vectors have last element equals to 0. So, inner product is 0.

Therefore, the external vectors d n you project on the space spanned by this 3 follows, compute that and separately project it on pi n, compute that and add that 2. That will be the net projection. That way if you compute, if you take d n and if you linearly combine this u 1 n minus 1 0 up to 1 n minus 1 0, subtract from d n and from sum of squares. Last element here is giving rise to what zeros subtracted from d n. D n minus 0 whole square independent of c 1 c to c p. So, that does not contribute anything to the sum of squares. I mean by other terms.

Other terms in the sum of squares comes from row number 1 to row number n minus 1, and when you minimize that you get those optimal filter coefficients which correspondent to the optimal filter coefficients at n minus 1 nth index. That is, if you have giving data d n minus 1 vectors u 1 n minus 1 up to u 2 u p n minus 1, you are finding out that optimal filter coefficients. That is what we will get here because in the sum of squares here, last element is d n minus all zeros whole square that is d n square independent of the coefficients, right.

So, with those coefficients if you now combine these columns, last element will still remain 0, but then when you will combine them, you will get d cap n minus 1. There is a projection of d n minus 1 on the space spanned by u 1 n minus 1 to up n minus 1. Isn't d cap n minus 1 and when you take d n and project on pi n, you will get a vector with all 0 and last component will be d n? So, you get a vector d cap n minus 1 and then, 1 d n you have to find out the error vectors, projection error vectors subtract it from d n. So, last element cancels out because there is common d n here and d n in and d n also, and the other half at the top half will become what d n minus 1 minus d cap n minus 1 which is projection error. Okay, that is p pi n from 1 angle like this d cap n minus 1. Sorry I am taking the error.

So, this error will be which error that is? If you take d n, project it on the space spanned by U 1 n minus 1 to e p n minus 1 and take the error that one. So, p n I said p n minus 1 perpendicular d n minus 1, right. This also equal to d n minus 1 minus that d cap n minus 1. Actually there should be subscript. You can indicate that data taken up to n minus 1. I think, but you can drop it also followed by 0, because d n vector last component is scalar d n. This summation projection on this part and projection on this part, that real factor that last component is d of n.

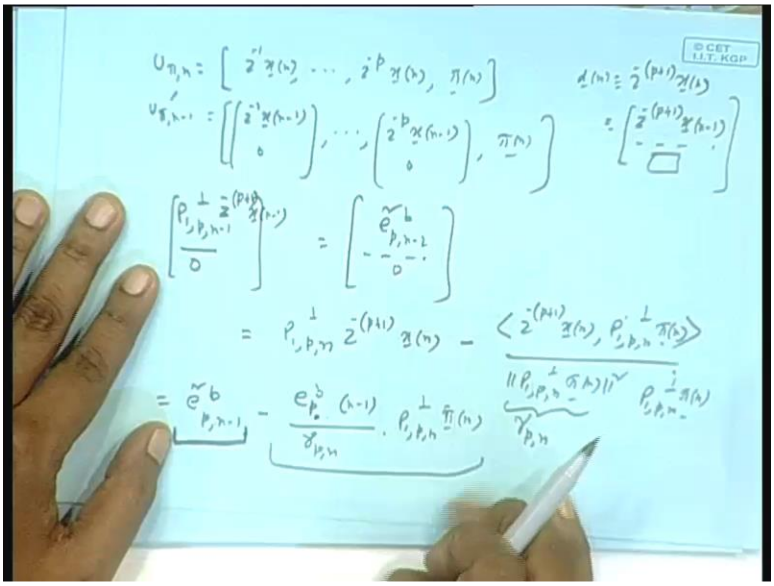
Coming from pi n to subtract 0, this is what you obtain there, but there is same as projecting d n on the space spanned by these follows also because the two spaces are same here. You can orthogonally you can decompose this space as the 1 space by u 1 to up direct sum if you need the projection of pi n on the space in the error space spanned by that error, and d n. I mean project on both add, and you are finding out that projection error on the projection to projection error. So, that would give rise to what? Sorry are you following this? I am taking u 1 to up d n projected on that, and the error there is nothing, but p n perpendicular d n. Projection is p n d n minus because when you compute the error, this will be minus sign will come up minus of what. D n inner product with I use this equation that day p n. I used the equation p n and here, I can show up this p n perpendicular form here on this which will be d cap n.

So, sorry which will be a corresponding error and then, the last component of that will come up. So, this is how the projection error vector at the nth index and the corresponding vectors at n minus is 1 index is related. They are not directly related. I mean this is not that this is equal to this. So simple you have this additional term. This quantity I told it is like an angle parameter. It is up to sin square theta. I explained through a diagram and I will not redo it. It is called angle parameter, fine. These all the external things that are coming up, this I will be putting in this context. This is what I will get. This is general result I will by now using it to update delta p n.

What is delta p n? Delta p n is this. I can also write delta p n as put back the results. You can take away this 1 1 1 operator. The two operators are same. You can take away 1 operator, sorry. So, let us concentrate on this error vector. This error vector is a vector with nth index. I will relate it to corresponding vector at n minus 1 nth index through a formula like this and substitute that here. This 1, this d n is here z to the power minus p plus 1 into x n. That is your d n here in this page I am doing now. D n in this case is z to the power minus 1 p minus plus 1 x n, and u p i n is what I see the projection operator 1 to p that is n nth index.

So, here it was p n perpendicular and here, in this context 1 to p. That means I will construct this. Finally, d n projected on the space spanned by this has to be related to corresponding things at n minus 1 nth index, but as this theory show this extra term comes up and that is why I started with not u n, but u pi n. In these contexts, what is u pi n? You take the columns here. What are the columns? Z inverse x n to z inverse p x n like u 1 into u p n. That is this, but for application of this theory I need the p n also. I need to consider i u pi n not un that is a, of this matrix. So, then using this theory maybe for your sake, I redo this in this context.

(Refer Slide Time: 12:29)



One is u p i n that is as I said z inverse x n p i n, right and another is that is what. By pi n here I called it as pi n minus 1. What is the problem? U 1 n minus 1 if this un I am making it n minus 1 here. Is that fine? So, the same thing you know the space as you know I have done it there in general context. I called it u 1 to u p. I am just giving this specific form that all, but the 2 spaces are same. Space spanned by this either for any external vector. External vector here is d n as I said d n is equivalent to z inverse p plus 1 x n. So, this d n projected on this space is same as projected on this space, and the errors are same. Dn in this case will be equal to the projection errors. You consider the space spanned by this.

Take the corresponding error, you project on this and this, and you take the corresponding errors what is that all you following this. This component if you take this, it is just restating the same thing using this language, these languages of this specific vector, d n last component of this and minus combiner coefficients times zeros will contribute nothing. In the sum of square, only previous terms of these vectors, only previous terms will come up. They will give the same linear combiner coefficients which are the optimal combiner coefficients of n minus nth index. If you use them and recombine them, you will get what? If you take the last component out, remaining part projected on this space spanned by this follows.

What is last component? If you take out what is it? That last guy is not any to me. Last guy is x n minus p minus 1. You agreed with this form. Now, that z inverse p plus x n you have. So, many zeros on top and then, x 0 x 1 and dot dot dot if you take the last guy out, then the nth index previous is 1 n minus nth index. If you stop here, we are getting as do the x n minus 1vector is delayed by p plus 1 times that component and the last component. Are you following this minus linear combiner coefficients? I mean this into one coefficient, this into one coefficient. When you add last component minus zeros and square up, they contribute nothing. Previous part minus c 1 times this c p times this that error vectors norm square is minimized. So, you get those coefficients which result when you take this part and projected orthogonally on the space spanned by these fellows.

Now, by those coefficients, you combine them bottom. The last row will contribute 0 and the upper part will contribute what. The projection of this fellow on the space spanned by this fellow z inverse x n minus to z inverse p x n minus 1, and this guy if you take the thing, only last component will come. You take the error and last component will disappear. So, this projection in 1 1 context, the projection is giving rise to what. Last component went out by subtraction p 1 p n minus 1 which is tilde. When it was n, I substituted n minus 1. Isn’t it? Because actually this is the thing and it was the 0, and added up to inverse. This is the backward projection error corresponding previous index.

Together I gave it a name this whenever time I have to write a name vector like this 0, and to avoid that I will give it a name. This time the projections is not nth index, but n minus 1 nth index, vector not x n z inverse p plus 1 x n, but z inverse p plus 1 x n minus

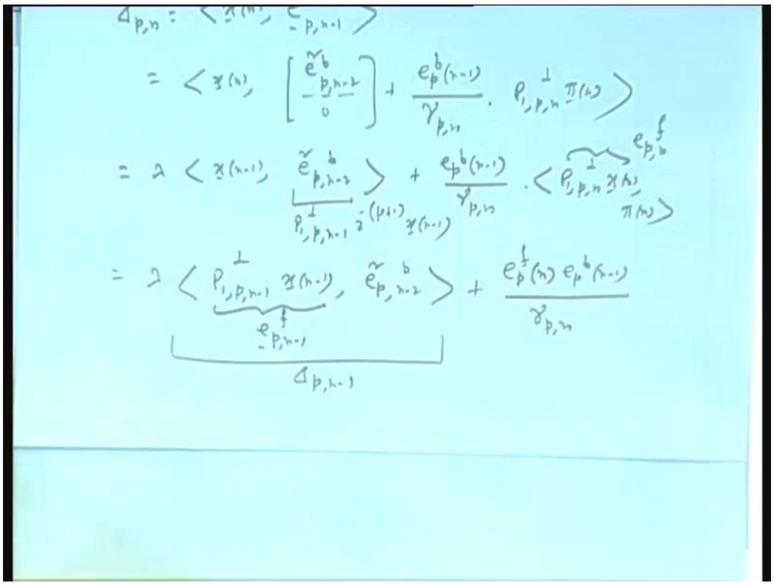
1 and all that. On other hand, if you project that guy on the space spanned by this, you consider this. This is related to simple consider this 1 1 part is spanned by this, and another part is you know what is called projected pi n on the space spanned by these fellows. Take the error and take the span of that on that you projected d n. So, this one component and then, take the error because I am taking the error here. So, one component will be what? Are you getting me? This fellow you are projecting on this.

So, this is I am not taking the error and there is a minus sign because you take the error instead of plus. You must plus the projections and projections error, this will become minus. Minus of what? This guy projected on what first you projected pi n on this, and take the error this on that component. P 1 to p n this is very simple now. Pi n projected on that error. So, this is that error thing. This quantity I will call this angle parameter, this I will call gamma p n positive quantity x square here. What I will do is very simple. You see I will take this operator, put it on this p i and it will become pi n. I will not solve this term. It will become pi n. This operator will sit on this.

What will that give rise to? E tilde b p n minus 1, Isn’t it that into pi n e tilde bp n minus 1 means this into pi n means last guy of this fellow this vector. This is your e tilde by definition minus last component of this. I should write the subscript, n minus 1, but as I told you since always the two subscripts will be same, we drop when we consider when we constitute the scalar error component. Always remember ideally I should put a subscript here with n minus 1 showing that data up to index n minus 1 is considered in finding out this projection error, but just for saving of space, nothing else I am dropping that index because I know that the two time indices are the same. We can always drag this here.

If you want some clarification just for saving space and no other reason, I am dropping it here, but this means if you take the data up to n minus th index, and then carry out p th order backward prediction error, the latest backward prediction error component is this. This times gamma p n into this guy which means this fellow is what LHS plus this. This much LHS plus this much in delta pn, I have to carry out the inner product between the two and finally, this. There is finally between x n and this guy actually between x n. Either you write in this projection form or this form. You have to do the inner x n and this guy I substitute as LHS plus this much.

(Refer Slide Time: 23:01)



So, delta p n this make the substitution LHS. For your sake I am writing in the vector form this 0 and plus this. Now, consider these two last components of x n and multiplied by 0 will yield nothing. In other terms last, but one term will be having multiplication factor lambda. Previously lambda square and then, previously to that lambda cube like that. Last 1 0 into x n back is free of lambda, but that is 0 before that lambda before that lambda square before that lambda cube. So, you can take lambda common. If you do that lambda times, it turn on to x n minus 1 vector into this because this is a scalar into x n times. Then, if you permit me at this stage only, I can pull this operator, put it on x n. Otherwise, x n times this and in the next step, I take this operator apply it on x n.

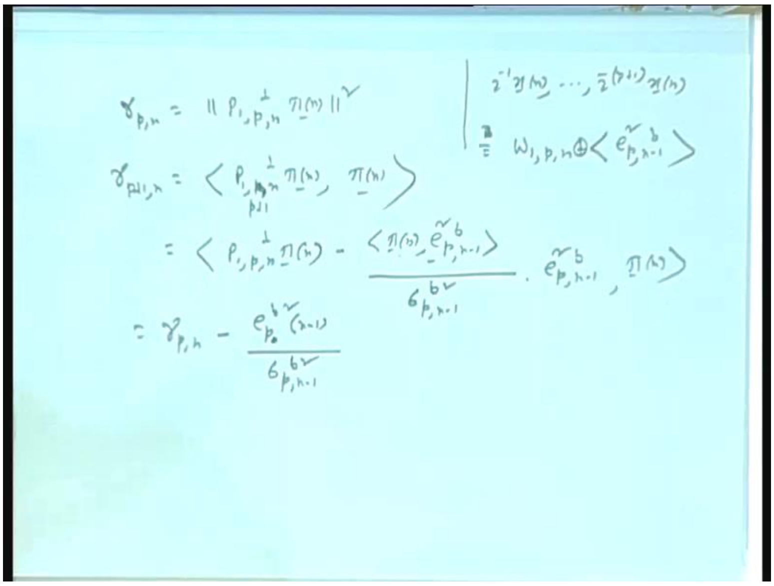
This guy is e p n f last component of that easily here. This quantity is what? Take the operator; apply on this and what will you get? No if you do one step at a time, if you take this operator, apply on this vector, this forward prediction error vector n minus 1 is the current index, so that current index vector current data vector projected on 1 to p. So, with reference to this current index delay from index 1 up to delay index p corresponding data vector minus space, on that I project this vector, take the error. This much only will be taken away. This much is applied on this and here, you see it is just very nice. The result e p f last component of this vector I can put a subscript n, but again for the same reason I will drop that subscript, but this means actually there is a subscript, n. That is indicating the data up to nth index has been taken on that error. Using that error vector has been computed last component of that, but again the same index prevails here. So, I just dropped for saving of space and nothing else.

E p f n here e p b, please try to understand the difference between the two. Here n and here n minus 1 meets here. The vector from which it is derived here that they are belonging to different you know. I mean they have defined in nature in the sense though even then they are forward and backward, they are otherwise different, but here it means data up to index n was taken based on that forward prediction error or you perform calculated last component of that, and here data up to n minus 1th index was taken based on that backward prediction error pth order was computed last component of that.

Just looking at these you know that aspect does not come as how much data was taken in the least square estimation or projection computation. This should always be there in your mind gamma pi n will equal to 0. P i n does not lie in the sub-space spanned by those x n up to z inverse p x n minus 1. This quantity is e p n minus 1 f, right and this inner product is delta p n minus 1. So, this quantity is delta p n minus 1. These are time update relation. For pth stage, you already know these two terms, they are coming from previous stage of the lattice multiply the 2. Suppose this is known. So, compute that, add to that, add that to this previous value. Find delta p n using this delta p n and these two errors, find out e p plus 1 f n e p plus 1 b n. Those two multipliers in the two arms, they are having the values delta p n. They are taking the value on numerator delta p n.

So, first you have to compute this from pth stage error. These are extra term, update term pass value, and get this using that e p f n and e p b n minus 1. Get the corresponding figures 40 plus 1 th index. So, let us try to write down the algorithm and see one more here. Gamma p n is this has come up because we are doing exact least squares minimization you know. No e operator and no approximation nothing. So, that is why these extra terms are coming. So, we have to take care of gamma p n. Gamma p n here fortunately can be order updated. Delta could not be and that is why you had to do all these kind you know I mean complicated analysis and all, and this more computation intensity unit division multiplication yesterday, and this extra terms coming and all that. Order recursion does not require such you know extra operations and extra variables and all that. Fortunately gamma p n can order update it. Let us see what is gamma p n? You can also do.

(Refer Slide Time: 29:56)

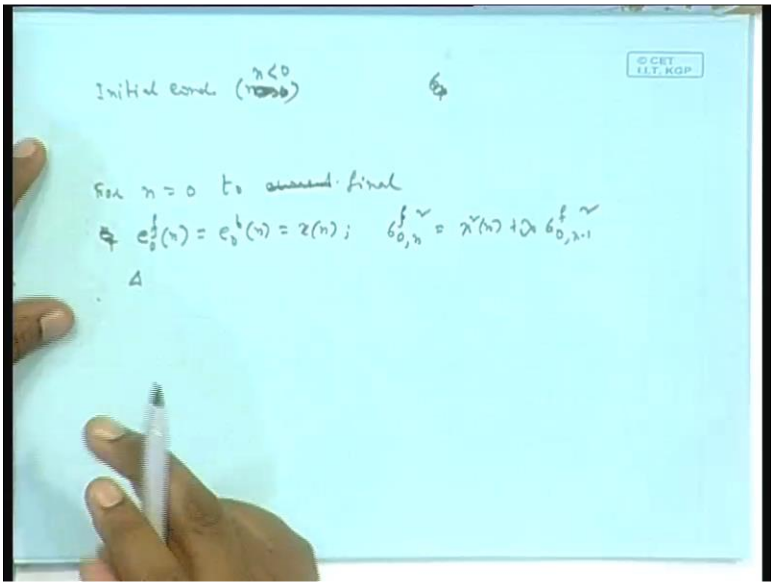


Gamma p n is what? Norm square of if I am correct. What is gamma p plus 1 n? That is norm square means inner product with itself just instead of p, I have to put p plus 1, but I can take away operator from one of that. Sorry, this is p plus 1 and here I take out the operator. This is always simple. What is the sub space here? Space spanned by z inverse 1 x n to z inverse p plus 1 x n. Z inverse 1 x n once again it decomposes space spanned by that is equivalent to w 1 p n. I think by now you are familiar with these notations and all that. Otherwise, what happens is come this guy, take its projection on the space spanned by this fellow and takes the error, but the main definition is this. So, simply to compute this error, pi n on this pi on this and the error. Pi n on this and error will give rise to minus pi on this. Pi n is not projected on this part, on this part. This part is here that corresponding error and pi n on this part will remain.

Here are norm squared. Norm square of this is same as sigma p. You have seen this. The two guys are defined for only this fellow has 1 0 on top up to p b n minus 1. That does not change the norm square. So, this times that fellow pi n on this. So, inner products by norm square into this fellow and comma his pi n. This projection is this much. So, comma pi n and you can easily see when I take this pi n and this. I will repeat this operator and need it on pi n again. It will be the norm square, but at pth order that will give rise to gamma p n. Obviously, order updating you know, it is not difficult. After a while it becomes kind of a routine. You can just clear out.

Domain of mathematics after while you catch hold to the domain, and you can just clear once think of operator is nothing. Initially it might look bit tricky, but kind of becomes very routine afterwards, and here this is scalar p i n times. This pi n inner product to this means last component and pi n inner product to this means again last component. Last two last components are same. So, the square of that and last component is e p b n minus 1 square. I can put again. I am repeating the steps, so that I can put n minus 1 here, this subscript, but I am dropping in square.

(Refer Slide Time: 34: 23)

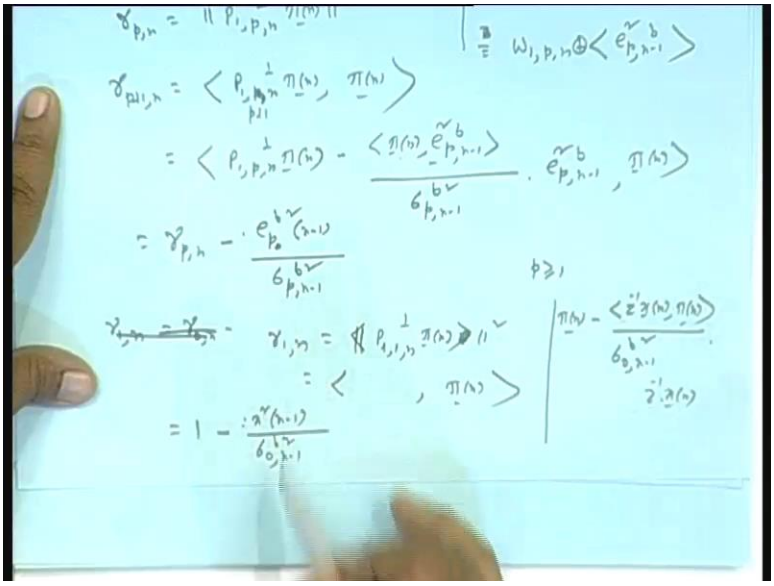


So, let us try to write the algorithm. Algorithm is not so easy. Initial condition that is n equal to 0 and then, for n equal to we start at n equal to 0, right x 0. So, this should be n less than 0. Isn’t it? Up to n less than 0 I think we will see because x of 0 is a valid data is not part of initial condition x of 0. So, for n equal to 0 to any index, current index anything you know or you choose a very large number. Of course, you will do the estimation are final time update or time recursion and at each time index, there is ordered recursion for the 0th stage complete.

The thing for 0th stage, first you write something for 0th stage and then, for p is equal to 1 2, something this way. So, at 0th stage you please correct me if I make mistake. You see it is purely extempore for me. Not even a speed of second I thought over this of coming to class. I am working out here. This is a workbook. Please go with me and correct wherever there is mistake. This is the thing. That is one thing, and this is the norm square sigma 0 f. What is that notation? It is sigma p sigma p n. No here this is the thing sigma p n f square. No that is equal to current data. Either you can write in terms of e 0 or directly x square n plus lambda times hold once. So, understand if you mean the index starts at n equal to 0, n equal to minus 1 is coming that is those quantities I will pull here, I will write here.

Otherwise, recursion cannot go on. You are getting me? This is recursion that n equal to 0. X square 0 is coming, but in this recursion see lambda into first. So, immediately become minus 1. So, I have to provide data for minus 1. That will be this initial condition thing.

(Refer Slide Time: 38:09)



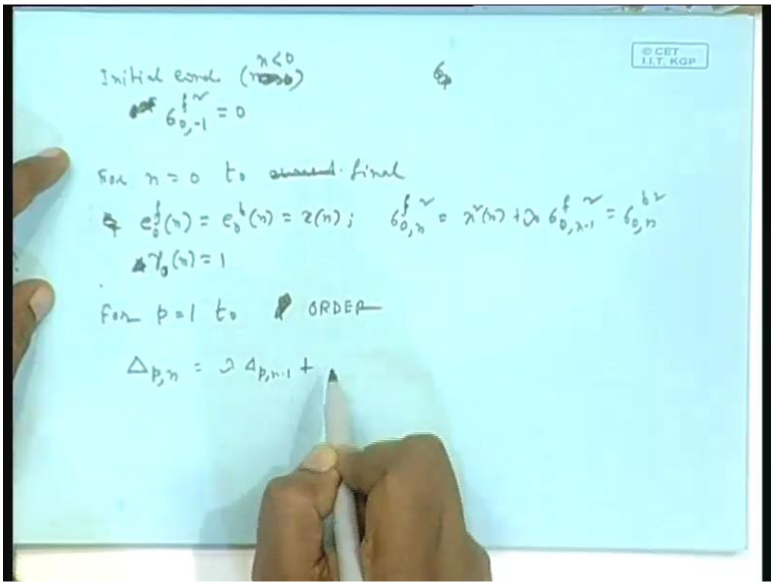
Gamma, I forgot to mention this order recursion. I will come to the algorithm if not today, next class. Again this is an order recursion you know. So, when does it start at? What is gamma 0 n? Suppose I do this algorithm, this is valid for p greater than or equal to 1. Suppose p equal to 0 in case we will do separately. P equal to 0 means like if I want to find out gamma 1 n, can I write it as some kind of gamma g 2 n minus. I mean this way, this kind of form. See this algorithm gamma 1 n gives rise to gamma 2 n, gamma 3 n, gamma 4 n and all that.

P equal to 1 if p equal to 0, there is a problem because p 1 we have already starting at 1 and going up to 0. You know there is no difficulty the way I did that first stage of the lattice, separate the same and previous we will apply here to be correct. So, that is I am saying this recursion I did for p greater than or equal to 1. So, from 1 to 2, 2 to 3, 3 to 4 like that, but p equal to 0, in case we will work out separately. Suppose I have to find out the recursion for gamma 1 n. What is gamma 1 n? By this formula p 1, 1, n. Sorry norm square p 1, 1 means z inverse x n only z inverse x n. On that pi n is to be projected taking the error. So, this is you write inner product of this will take out the operator from the last guy. Same way as here and same way as here, and p 1 1 n p 1, 1, n perpendicular pi n, just pi n p 1, 1, n perpendicular pi n. You just do directly.

What is the meaning of this pi n projected on z inverse x n? I have taken the error that is error. So, pi n minus pi n minus z inverse x n pi n, thereby norm square of z inverse x n which is sigma 0 into z inverse x n. What you substitute here is pi n with pi n inner product is 1 pi n with pi n is 1. So, 1 minus it is very much like this, and again z inverse x n z inverse x n with pi n. So, this is the scalar same thing. This component this much was scalar, so only between the two.

Similarly, this much is a scalar only between these and pi n. So, that is the repetition of this inner product. These and the same inner product square of that. What is that last component of x n minus I mean x n minus 1? Last component that is x n minus 1 square of that is these vector last component again same thing, these vector multiplied by pi n means last component of these same thing squared up. So, x square divide by sigma 0. Now, compare the two formulae, general formula and the special case. This quantity and this quantity are same. You agree this quantity and this quantity are the same. They both are matching.

(Refer Slide Time: 42:26)



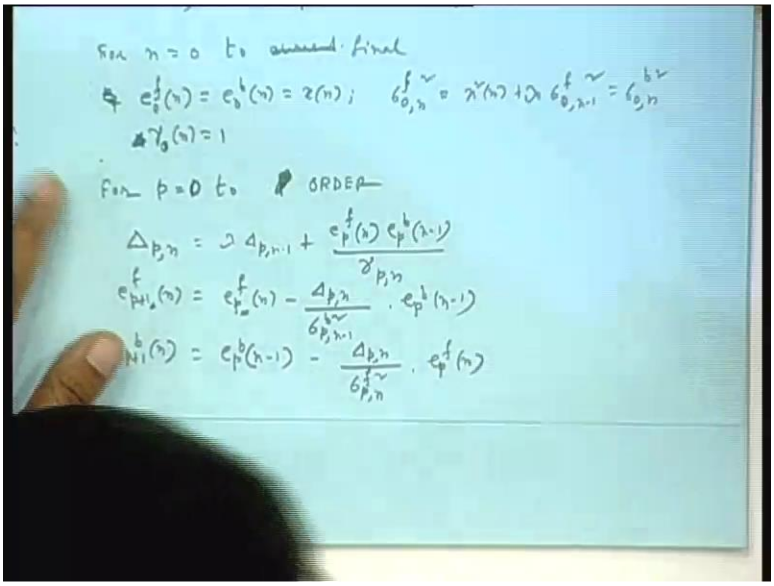
So, gamma 0 n, I find out gamma 0 n to be like this one. So, gamma 0 is 1. So, here in that algorithm is gamma 0 n is always 1 at any index n independent of n. Then, by the way as I told you here it takes sigma 0 f n minus 1 square, but there is no data before n equal to 0. So, those norms are 0. So, pi only for 0th order, it is true for all orders. This sigma, ask me sigma, but I am talking about index minus 1. Though to be frank in the lattice algorithm, what comes forward prediction error? This sigma f I do not need this at any stage other than 0 because always I will come across sigma 0 f n square only in the first stage while updating these guys comes, but only order update the norm square sigma 0 f n whole square n does not become n minus 1. Problem comes with the backward prediction. They are all as I told you one layer of memory is required to store the previous norm square of e p b n minus 1 vector. That memory is required.

That is why for all orders remember that is for all orders we need that. You need that here also for all orders. These are very crucial quantity mind you. So, these amounts to these say that you need to store. Isn’t it? You need to store because nth index you need the data use. I mean obtained at n minus 1 th index, but there is some backward prediction error p th order at any general p. So, all for that p, this needs to be provided. So, therefore, in that initial condition I should have data for that backward prediction error, and norm squared for all order at index n equal to minus 1 because for doing computation for n equal to 0, also I need that data, right, but for forward prediction error, I need only for the 0 th order. In all other computations from sigma p f n square, you get sigma p plus 1 f n square.

N does not become n plus 1 and all that in all those relations n minus 1 thing come only in context with backward prediction error. If you take all the relations we derived, you will see there what problem comes, and what problem we discussed once and moment has come to tackle that problem, the problem of linear dependence. So, here sigma 0, no problem with these sigma 0 minus 1 f square equal to 0. I will come to this row fill up more things as and when as I find, I look, I need I will come back to that.

So, it is not that this is done. You can hear I am not sure, but now it is for I mean in the general index n for any index n 0 th stage, lattice 0 th stage e 0 f n e 0 b n. I know what corresponding norm square is. These also equal to sigma 0 n b square. These are verified and gamma 0 n equal to 1. Now, for p equal to 1 to some name you give because finally is the time index order. For each index, I am doing order update. So, capital P is a projection operator. You give a name order equal to 20 equal to 30 after which what all you want to work up. At its stage, you first need delta using delta update. So, first we know the formula delta p n. That means, it is equal to lambda delta was there. Isn't it? Delta p n minus 1 plus, this formula that we derived, where is that page gone?

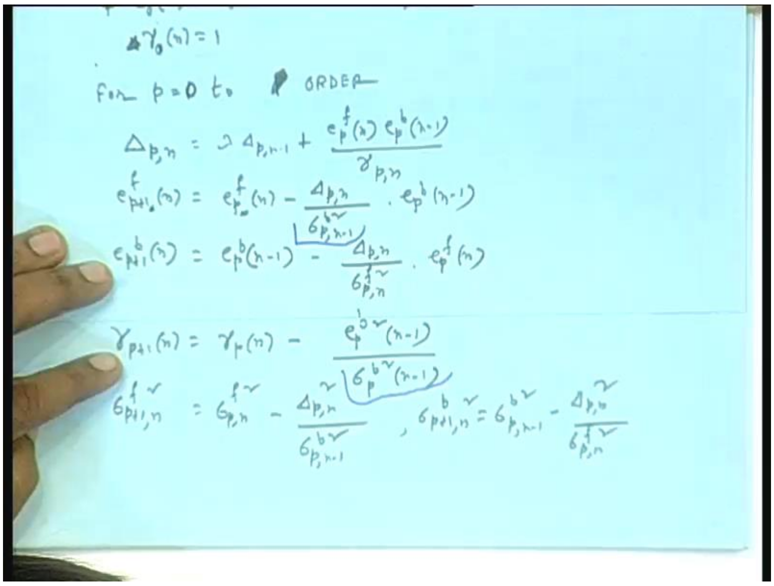
(Refer Slide Time: 48:16)



I know, but where is that page? This is that thing lambda into delta p n then. So, p equal to 1 equal to p equal to 0 because when we are in the 0 th stage of the lattice and then, first stage, second stage delta 0 n delta 0 n minus 1 and then, e 0 f n using the given e 0 f n e 0 b n minus 1 e 0 b n minus 1, I have to give here remember. So, that means, e 0 b why only 0? This will come at any p. So, e p n minus 1, it should be written for all p from 1 to order this equal to 0. When you write the program, but I am not writing it when I put e p means for all p from 1 to order. So, e p b n minus 1, that is minus 1 0. You understand how I am proceeding. The way I write algorithm, immediately I will find wherever I have to provide data for n minus 1th index. I am going back there and writing the data. Where I have done? I have not mugged up the algorithm you know. So, I have to reconstruct by this back and forth way of doing.

So, at 0 th stage say I found out the multiplier value from its first value. So, I have to store these also. This also minus 1 will be 0 for all orders I got. You had that done. I need to give first value that is for all order, not just 0 th stage, not for p equal to 0 for any p. So, you know that I know these two values. So, I get this fine using this minus by sigma p. This was n, right n or n minus 1. This was n minus 1. For p equal to 0 I already know the norm square. They already given here delta have been obtained here. First delta 0 n obtained here. These prediction errors are given to me either directly or through their first values. So, I know these two. I found out these, but I need to now update the norm square and I need to update gamma update formula needs this. Nowhere is gamma update formula, reverse side gamma update formula. Gamma can be updated here.

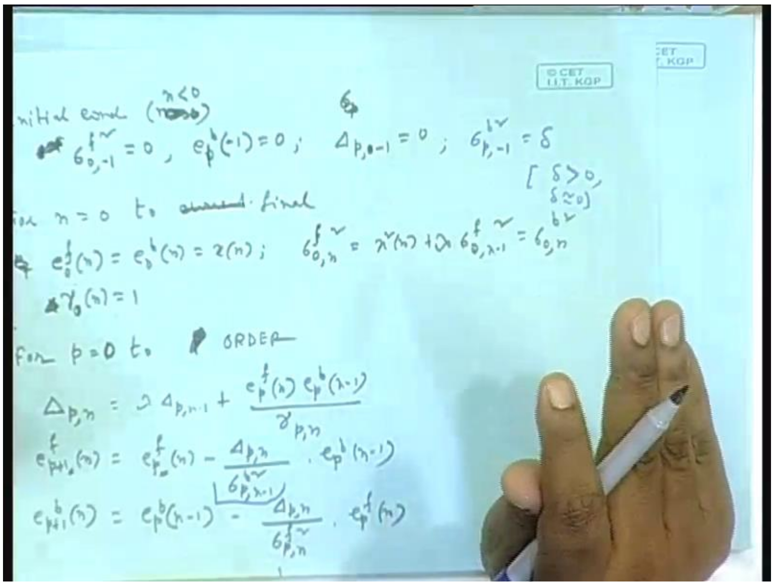
(Refer Slide Time: 52:21)



Sigma values have to be updated. This we worked out. So, it does not cause any problem here. This is a simplest of all relation. Whatever I need the formula, they are already obtained. Deltas are obtained, all these lower order norm square variances are obtained, and all these will go around. Problem is here this guy. This guy is coming here anywhere else here. So, I have to give values for this for n minus 1 nth index. What is this guy? This guy is the norm square of the backward prediction error vector. What is the backward prediction error vector? It is z inverse p x n projected on the space spanned by x n up to z inverse p minus 1 x n.

So, if that z inverse p x n is lying in the space spanned by x into z inverse p x z inverse p minus 1 x n, then that projection error will be 0. At that there will be division by 0. You remember I did it elaborately that time and when does it happen, number of row less than number of column. If that all 0 column or more than 1, all 0 columns appear in the very initial stages to take care of that. So, till that our index n increases beyond that number of row columns, number of columns order plus 1 whatever this problem will continue. To avoid that what we do? We verify. I will come back to these issues. There is no time that instead of giving this error as 0 sigma p minus 1 b square.

(Refer Slide Time: 55:01)



Instead of giving it 0, we give it as very small quantity delta, delta greater than 0, but approximately 0, very close point 0 0 0 1 or something like that. Effect of that you try to investigate in next class I will take up. Next class what I will go for; you have to do justice to the title given. Now, there is another category of ionized lattice filter called q r. Q r means q is a unitary matrix and r is upper triangular matrix. Given a data matrix say in that as I consider matrix with columns x n z inverse 2 x n dot dot dot z inverse p x n in a column matrix like that at any index, you have got a big rectangular matrix.

That matrix can be written as product of 1 matrix q which is a rectangular matrix, but columns are usually orthogonal or orthonormal. So, it is a unitary matrix times upper triangular matrix, and as time index changes, elements of q and r can be updated by simple formula. There are various ways of doing it. One is by orthogation. We see that it results in change that. Another is by rotation. Have you come across this term rotation? We will see it. It is a beautiful thing. You know rotation, it is very picturesque. Actually the way rotation tapes takes place is beautiful. So, by rotation we will consider this one that given a data matrix, we will basically get its equivalent upper triangular form and then, getting those linear combiner coefficient will be very simple.

It is all being an upper triangular matrix, and it will basically be done through a systolic array. Those who know I mean some people have done my course earlier, so they know, but those who do not know, it is an array, pipelined array of processors arranged in a very regular manner, and overall computation is very fast. It is really pipelined. So, we will see that from the next class. This just I will take up in the next class. About this delta issue, what I will do? I will take out this paper today, so that I want to rewrite the entire algorithm. I will bring it back. This delta you also think over. You see this delta will prevent the division by 0. Delta will always make sure that even if there is all 0 columns, its norm square of the corresponding error vector is not 0 by at least delta. It is delta. This we verified.

Thank you very much.