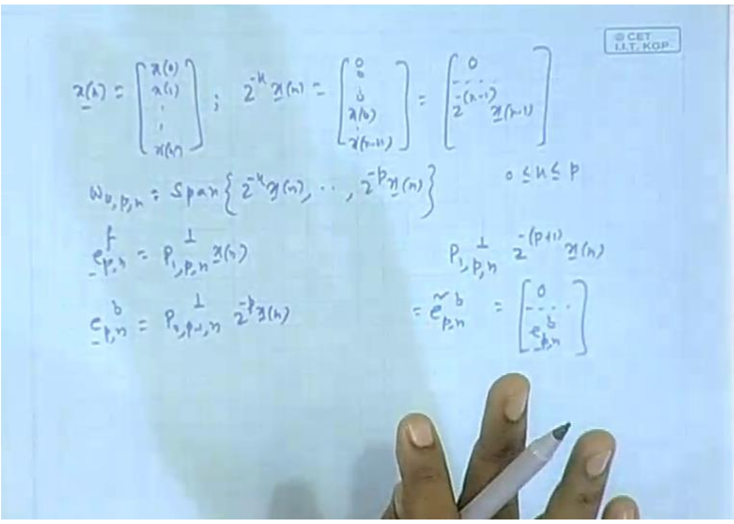
Adaptive Signal Processing
Prof. M. Chakraborty
Department of Electrical and Electronic Communication Engineering
Indian Institute of Technology Kharagpur

Lecture - 32
RLS Lattice Recursions (Continued)

(Refer Slide Time: 01:09)



So, quickly we had those definitions xn I am writing even though this clear cut repetition, but I have to refer to them you know time and again which was these then you have seen one thing the z to the power minus k xn was what 0 0 and then x0 up to xn minus k which we saw yesterday can also be written as 0 z to the power minus k minus 1 x n minus 1, right, this you have seen yesterday, is it not? x n minus 1 vector on top we are keeping as 1 0 and 1 more 0. These are the thing and then we had W kpn was this is repetition, but this my workbook. So, I have to write them.

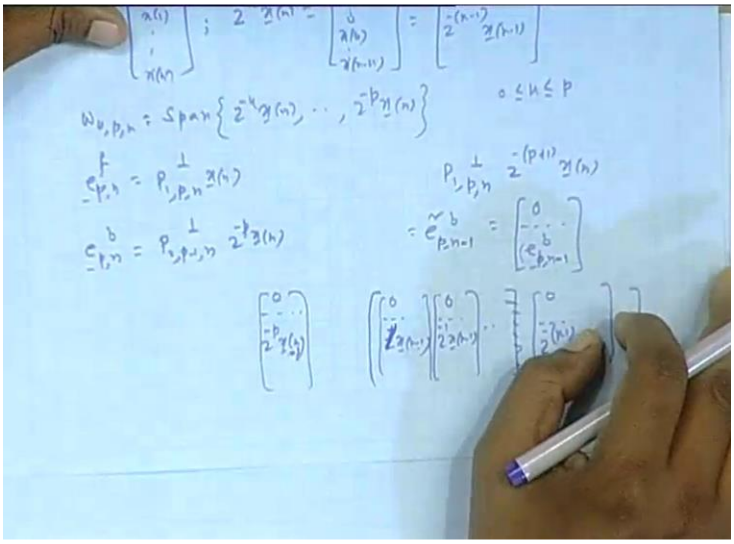
K greater than equal to 0 less than equal to p, these then we defined e p,n to the power f forward prediction error vector that is xn to a projected orthogonally on the space span by z to the power 1 minus 1 xn up to the 2 minus pxn pass p values at each index n those data samples are linearly combined subtracted from the current data sample error. Those errors are squared sum with varying factor lambda minimized. That give raise to those optimum minimize means you get the optimum filter coefficient filter coefficient. Then using these coefficient combine them you get the vector there is projection take the error that is a error that is the meaning of this, so that was 1 Pn.

Similarly, we had this is repetition in this context, yesterday we came across one vector P1 pn perpendicular z to the power minus p plus 1 xn.

This we wrote as e tilde and we said that this should be this. Let us again quickly sum up on what was there instead of projection just cons projection error. Consider the projection, this is just recap, but since this is a crucial thing you know it will be used again in several times today forget about the projection error first the ori projection itself error means from this vector only you have to subtract this projection.

That projection will be what this guy will be projected orthogonally on the space span by z to the power of minus 1 xn to z to the power of minus P xn all top of them the 1st entry is always 0 1st entry of this vector also is 0. So, if you linearly combine these and subtract from this z to the power minus p plus 1 xn top entry does not give rise to anything remaining once you are finding out the error each errors is squared up multiplied by weighing factor lambda. Lambda some power of lambda and add it minimized that is same as, as though if you are considering z to the power minus I mean just like this.

(Refer Slide Time: 04:38)



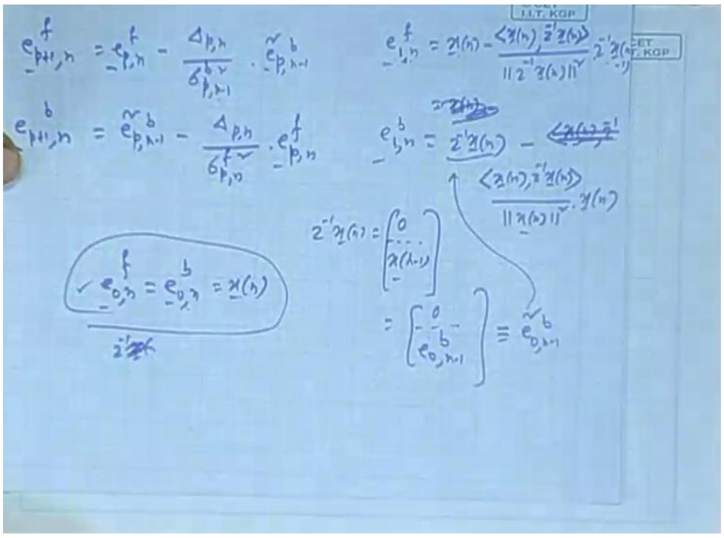
First column was 0 z to the power minus 1 xn minus 1 sorry not z to the minus 1 just xn minus 1 z to the power 0. In fact, z to the power minus 1 xn that is not a 0 and xn minus 1 then next column was 0 z to the power minus 1 xn minus 1 dot dot dot 0 dot dot z to the power minus p minus 1 these are the columns and you are projecting this guy this fellow, this fellow means this is 0 on top into p plus 1 0s, but that you can write as to z to the power p minus xn. You are linearly combining them taking the error forget about the top entry that error vector yes n minus 1. Thank you.

n minus 1 I mean you are linearly combining them take subtract from this part as though the top does not exist because that does not give rise to any contribution in the overall sum of squarer errors . So, you are linearly combining them subtracting from these that error vector you are taking each components squaring up and adding with a varying factor and minimizing that is same as, as though this is projected, this fellow is projected on the space span by this vector this vector up to this vector.

If you not euh see n minus 1 you call it index n prime then xn prime z inverse xn prime dot dot dot z inverse minus I mean z inverse within bracket p minus 1 xn prime z inverse pxn prime. So, this forward prediction error vector, sorry backward prediction error vector at index n prime that is we are going back by n prime by p that is n prime minus p and looking at p future terms. So, that will be backward prediction error vectors pth order at index n prime. So, that will be this entire thing would be what pth order backward prediction error at. So, I was making this mistake I mean instead of this it will be better if I put this that is this is what I did I did not put n minus n, n minus 1, you could have corrected me.

So, this is clear, this is the n minus 1, n prime index backward prediction error pth order backward prediction error at n prime index that vector that is what we had, but all this things and based on this you can do recursions you know.

(Refer Slide Time: 07:15)



e p plus 1,n to the power f was I am not redoing this, this is very simple sigma pn minus 1 b square into ep tilde, b you can take the last component for the time being I am not taking the last component that is very simple and this was I did this also. Isn’t it. Yesterday I did for this also that was minus same delta pn we assume that this is not real thing division by 0 quickly what was the thing when you go for from e pn to the power f to e p plus 1 n to the power f. Basically in that matrix data matrix we are adding another extra column z to the power minus p plus 1 xn.

I am assuming here that that column does not exist in the space spanned by the previous p fellows that is z to the power minus 1 xn to z to the power pxn is not obtain cannot obtained as a linear combination of that, but I told you was also yesterday that is always not always true if you running index current index is much less and order is more I mean this new column could be all 0 column its already existing in that space.

So, that time your projection error vector will be 0 vector and as the result is norm square will be 0. We will have this problem this does not happen in the forward prediction case this is a forward because the columns as such you know x 0 x 1 x 2 dot dot dot and then next 1 is 0 x 0 x 2 0 x 0 x 1 x 2 then 0 0 x 2 x 0; these columns are linearly independent isn’t it if you go for this quantity. what is this quantity this cannot be 0 because past vector xn vector this orthogonally projected on the space span by p pass vectors pa, but xn never lies in the space span by the those passed p vector; obviously, that we have shown and as a result the error vector is non 0 vector the norm square is non 0.

But this can happen. So, we have to be careful and this for index n which less compared to the order. So, that as you go on order updating and all 0 column emerges then you have to be careful of this comes up you have to do some manipulation the algorithm so that division by 0 does not take place. That is the separate story that I will do and that again because very smart thinking say later, for the time being, we assume that we are running index current index much higher. So, we are not having in this problem also another thing I said that that I am assuming that I have at least 1 column vector that is xn that is as you go back say from recursion p plus 1 to pnp 2 p minus 1 mp minus into p you are starting at this point what is e 0 nf I do not know here e 1 nf I know from that what is e 1 nf. xn projected on space span by z inverse 1 xn the error vector that is this, but if you put 0 here then that is a problem, is it not? xn projected on z inverse 0 xn and then error and all that this is a problem actually.

Remember, when you see epn, we start at what is the problem is we start at what is the when I they say epnf what is the sub space, sub space span by z inverse 1 xn to z inverse p xn if I put a 0 here it is a p, mean that definition is invalid because z inverse 1 xn up to z inverse 0 xn has no meaning is it not? z inverse 1 xn upto z inverse 1 xn, z inverse 2 xn or z inverse 1, z inverse 2, z inverse 3, that is fine, but z inverse 1 xn up to z inverse minus 0 xn, z inverse 0 xn there is no meaning there is a our limiting case is this from this recursion that directly I cannot go for e 0 nf similarly for the backward prediction error. Here also our starting point is this, all right.

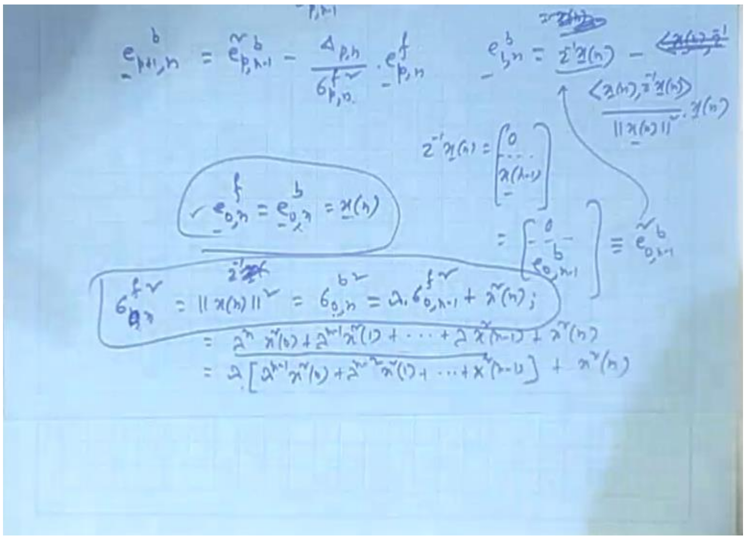
So, giving this you can find out e 2 fn e 2 bn so on and so forth by the recursion, but how to obtain them we will see that by clever identification of some vectors as e 0 nf and e 0 nb we can still maintain the same lattice recursion for the 0th stage also, this is called stage 1 when p equal to 1, when p equal to 0 that is stage 0 that is the 1st stage.

Now, e 1 fn what is e 1 fn is simply what is e 1 fn xn projected on z inverse xn and the error isn't it e 1 xn fn, e 1 fn means what is the soft space spanned by z inverse 1 xn up to z inverse 1 xn that is that fellow only on that I am projecting xn taking the error. So, simply xn minus errors. So, xn minus inner product, xn vector divided by norm square of which you can also write as xn minus z inverse xn is here. I write separately what is z inverse xn z inverse xn you all know is 0 x n minus 1 right and similarly this recursion we write this was the xn minus 1 this recursion we write what is this recursion. I know the genuine definition of epnb means the vector z to the power of minus pxn that is z to the power of minus 1 xn will be projected on the space spanned by z to the power minus 0 xn up to z to the power minus p minus 1 xn.

P is 1 here. So, z to the power minus 1 xn will be projected on the space span by just z to the power minus 0 xn because p minus 1, p is 1, p minus 1 is 0. Is that difficult, space is from z to the power minus 0 to z to the power of p minus 1, p is 1 p is 1 here. So, space is z to the power minus span by only z to the power minus 0 xn that is the xn itself whom I am projecting z to the power minus p, xn p is 1. So, e to the power 1 xn just 1 delete. So that means, the error vector will be minus inner product between these and xn be same as xn and because they are real.

Okay, now you see if I make the substitution if I say that e 0 nf equal to if I make this assumption then you see things will be correct if I put this e 0 is equal to xn suppose I say this; that means, e 1 fn and what is the z inverse xn this z inverse xn is nothing but 0 dot dot dot xn minus 1 which is also this conforms to the form. If I assume this you see if I define this what happens to this lattice recursions xn is immediately e 0 nf, p plus 1 p that is coming again delta n delta pn was what? what was delta pn inner product between these 2 that is coming you see z inverse 1 xn means 0 xn minus 1 and since xn is e 0 nb is e 0 n minus 1 b and by our notation this is this and this is this inner product this coming of to be of this form between these 2 between e 0 nf and this fellow e 0 tilde, e 0 n minus 1 tilde b right this fellow this goes here. Same here also that e 0 tilde you see this is coming up ep tilde. So, e 0 tilde is coming up here same here also epnf xn is e 0 nf and norm square getting plus 1 xn norm square is simply norm square of this recursive norm square of this part this 0 means nothing. So, this is nothing but norm square at index n minus 1 or backward prediction vector at index n minus 1 fine and xn square means xn is this. So, norm xn square means norm square of these coming to this form. So, at a 0th stage this is what you start with.

(Refer Slide Time: 17:54)



Then as you can see here, I already have repeated sigma p say nf square this is norm xn square very simple which is also sorry sigma 0 which is also by the way, but non xn square means what? What are the elements x 0 x 1 upto xn we are squaring up the terms adding, but anyway the weighting factor what weighting factor last 1 multiplied by 1 last, but 1 multiplied by lambda this way. Now suppose from this part from this part I take lambda common, I take lambda common from this part lambda times these previously it is the norm square of the vector xn minus 1 so that means, how this can be updated directly in time this is equal to the same norm square index n minus 1 multiplied by lambda, lambda weight brings down the contribution of previous norm lambda times we call a for b does not matter plus new guy this is the relation.

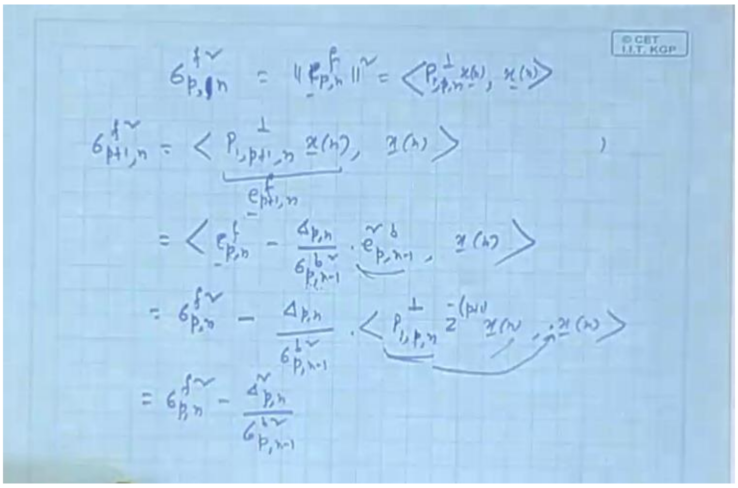
So, at 0th stage norm square directly time up vector each index n they recomputed recursively from that previous index value and using the current data previous value that is the value at the previous index current data using the they are computed is per time update at 0th stage they cannot you cannot get them back from any previous stage because that is the starting stage. So, there you are obtaining them in time are previous index value and current data putting to in this formula you are getting the new value.

But such time updating cannot be done for this general sigma pn square sigma pnf square and sigma pnb square for any stage p that will be order update that you know we have tried time updating equations do not walk out 1 cannot get order updating order updating means once I know them then keeping the index n constant at this index only from this I will find out sigma 1 nf square sigma 1 nb square and sigma 2 nf square, sigma 2 nb square. So on and so forth at the same index. So, once find once I do time updating at 0th stage get the corresponding values for index n then at this index n I go ahead by the order updated formula for various stages I get the respective values mind you unless I have them I cannot have this multiplied in the lattice this is the purely adaptive thing at the every index the multiplied value changes. So, for pth stage I first have should obtain this, obtain this, obtain this, obtain this and then only I can proceed Sigma pnf square and this fellow can be easily order updated you know using the we need for stochastic lattice we will just do that.

Problem will come with delta pn this guy cannot be order updated. So, what they do, they you can then you can derive a time update formula of this guy at each index n for each order this can be time updated. So, for all for every stage I will obtain this from its value of the power from its past values.

I will not go on I will not work like this for 1 particular stage I will do time updating and then go on updating it in order that is from delta 0 n to delta 1 n to delta 2 n to delta 3 n not that because that is not do all that is not that is does not work out is not is not possible to get this kind of relations. So, there are for every stage of lattice as each index n we have to compute this directly to use the recursive, time recursive formula that is from its past value for a particular p from its past value that is delta pn minus 1 will give raise to delta pn and this have to do separately for all the stages no order updating this things for norm square time updating only at the 0th stage in this manner and then you go on order updating for the same index n from sigma 0 nf square to sigma 1 nf square to sigma 2 nf square, so on and so forth. See, you first obtain the time order update relation for this norm square and then I will tackle this. This tackling this will be beautifully linear and not easy but I will get exact time of that relation.

(Refer Slide Time: 23:25)



So, we have got this quantity sigma pf, sorry sigma pnf square just for your sake I am writing down this is your epnf norm square; that means, that means norm inner product with itself at epnf means what p 1 pn this your epnf comma itself, but then I can drop this projection operator from here by those tricks we have the same projection operator here column projection, but I can drop that operator there you remember na this simply xn.

So, that means, if I have to find this guy now sigma p plus 1 nf square which is this. This is basically we have already find out, found out ep plus 1 for your benefit I am writing in this expanded form, but this is nothing but this and you are already know its order update relation. We know this lattice recursion what is this guy you already know you can substitute here minus this have you looked this I am substituting here.

Actually this is absolutely identical to order updating of those I mean prediction error variances we need in stochastic lattice procedure is absolutely same, and then xn with this xn with this what is epfn. epfn we know this quantity p 1 pn perpendicular xn that with Xn means I can repeat this operator on xn anyway, it will it will becomes epfn with itself this is what we did that time also minus these 2 quantity.

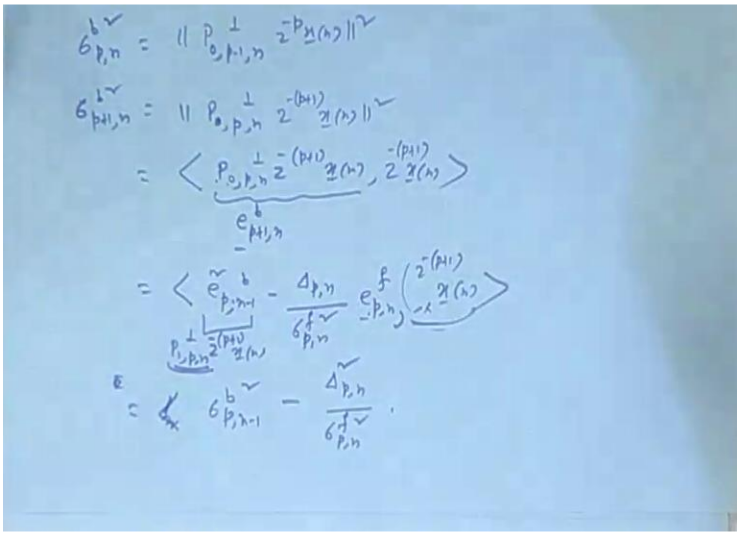
So, delta pn by this remain outside this with this what is this guy I give you definition this guy is nothing but p 1 isn't it what is this guy I in the very beginning today I told you I told you I used it this is nothing but this case 1 p comma n, 1 comma p comma n that that is space we are projecting this fellow that is why you add a 0 on top and all that and this e tilde fellow came up you remember na we did in the very beginning today.

This would itself is this operator you can again pull on this that will give rise to what epfn, epnf xn and this p 1 pn perpendicular that will give rise to epnf and epnf and this fellow which is this fellow their inner product is same old delta pn here there is no difference we did similar absolutely similar thing that time also in stochastic lattice are you following what I am saying this operator I will just apply on this and then I will get epnf here epnf and this total quantity is epn minus 1 tilde b they are inner product both are real. So, it does not matter who is 1st who is 2nd that inner product is delta pn.

Earlier in the stochastic lattice this will become delta p star because, I was dealing with complex value with in general and it will become star, because it has got reversed also it become mod delta square and also those there, but we had this nothing like that because all are real , so delta pn square.

Once again I am assuming there is no division by 0, division by 0 comes from the same thing na from the same lattice recursion. So, this is an order recursion suppose if find the pn stage you already know delta pn important thing is you must know this suppose you know this and you know this to norm square for pth order, pth stage you can find out the same thing for p plus 1 nth stage for the forward prediction in a same manner you can get for the backward prediction I will do in a minute this quantity.

(Refer Slide Time: 28:54)



Suppose, I just start of n minus 1.Let us start with pn only, if I know this I can use it at n min n plus 1th index I can store it, because I need it. I need the value at n minus 1th index here is it not? So, instead of what are updating this quantity where index is n minus 1 let me try with n first, that means, this value will be stored in memory for usage in n plus 1th index. Let us start with that first, so this quantity is what norm square of and in case I land in to difficulty then I will switch over to minus n minus 1, let me try this first.

This is a quantity xn with itself, but here I drop this operator x. I can do that this quantity with itself and then here I drop this operator and this quantity is what by our definition backward prediction error backward prediction error and you starting at 0th up to pth, z minus 0 xn to z minus pxn on that space. The space span by them you are projecting this guy taking the errors this directly with this comma xn and I already have obtain the backward pred this formula for na this recursion. So, I will just substitute that there.

Absolutely, similar procedure you know this is very routine and between these 2 what are this quantity you remember this was p 1, pn I gave the definition of this tilde today again for I mean as repetition in the morning you have to start at 1 go upto p all have 0 on top this quantity, between this quantity and xn I repeat the operator this bring the operator on xn immediately I get back just a minute oh here I get just a minute, just a minute ya here, here I will have say sorry.

Thank you very much I am doing fast.

So, here thank you.

So, this operator I bring back on this thank you very much, yeah. So, moment I bring back this operator on this I get the same e e the tilde term here also this is this thing this operator I apply on this fellow same e tilde pn minus 1 b would come there and inner product itself. So, norm square and that norm square is same as minus this remains as it is into between these 2 what is epnf? epnf is p 1 pn perpendicular xn that I will apply on this. So, again the tilde will come up and inner product between epnf and that this tilde term which is again delta pn. So, you will get delta p square, not understanding this is very simple pretty simple. So, this is a order updating I remind you I am updating the guy at index n, but in the lattice recursion I need the same fellow for index not index n, but n minus 1.

So, that means, whatever I compute for index n that needs to be held in memory for use at n plus 1th index similarly what was computed at n minus 1th index that was held in memory, now we used in this recursion index n. So, it needs a lot of memory.

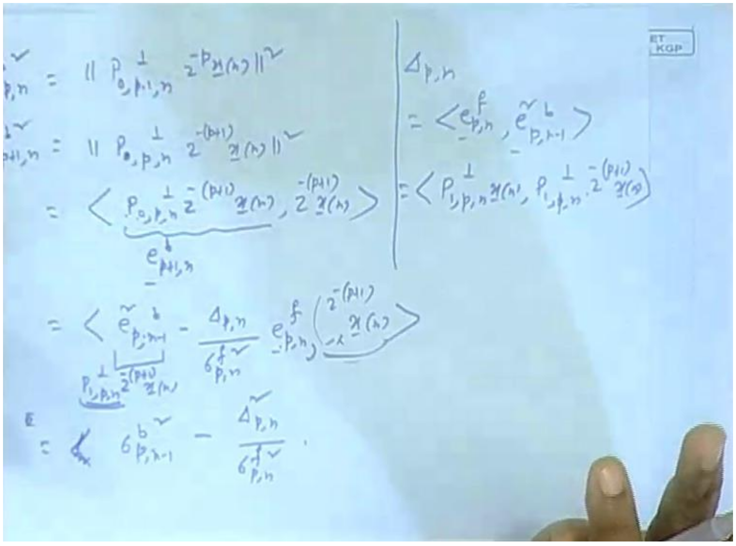
For holding this order updated value for only the this fellow sigma p b square whatever you obtain from current index this is to be stored because lattice recursion needs them for the past index not current index here its fine, but for not for this norm square see you need to store.

So, order updating is fine past means also when you switch on the algorithm we have to give initial condition for these fellows at n equal to 0 starting. So, at n equal to minus 1 what value it had you have to give initial conditions and there I will inject something carefully. So, that never division, division by 0 occurs anywhere that separate story, but you understand this fully I feel that since it depends on past value whenever I switch on the algorithm I have to give a initial value for this.

Similarly, for the 0th stage wherever there is time recursion I have to give a initial value for the 0th stage there is a time recursion to be that is current norm square was using previous norm square multiplied by lambda plus square of the current term you remember this thing na. So, here again for this also I have to give initial conditions, because there is a time recursion any time recursion means, but am obtaining current index value from previous index value I have to give initial condition. Is it not?

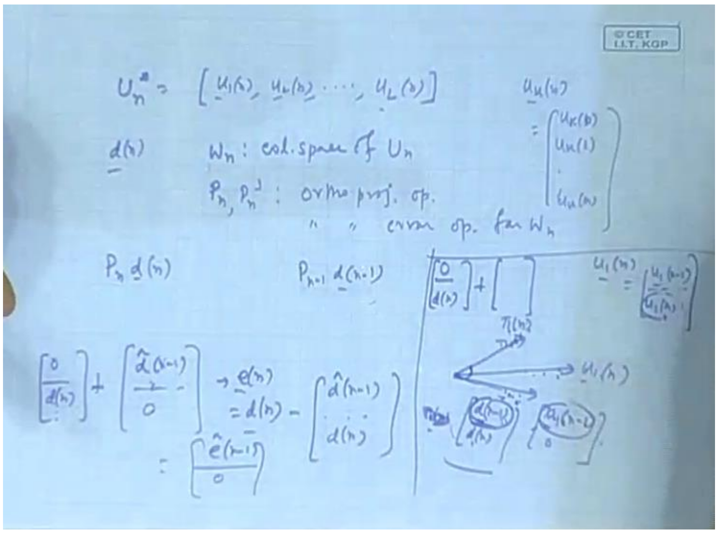
So, here I need for this whether it is a for b does not matter and for every stage I need only for the backward for norm square of the backward prediction vector, now comes this delta fellow that is the most difficult case delta you know is a inner product with in those 2 quantity epnf and ep tilde e tilde pn minus 1 b those 2 fellows that is.

(Refer Slide Time: 36:53)



Separately our biggest story is with this guy in terms of projection this is p 1 pn perpendicular Xn here also space is same most importantly space is same, but vectors are different this guy and here people are tried many of space order recursion is not possible here we could not find out, but they can find out a time recursion there is delta pn to obtain from delta pn minus 1 and some expression and all. Not so easily, but at least time recursion delta pn to be obtain for each p delta p to be obtain from delta pn minus 1 separate time recursion, because other way order recursion are not possible to be found out we have to calculate out the time recursion. If I do that let us do because that will not be possible entirely today let us do something.

(Refer Slide Time: 38:05)



I will be considering a general, now suppose I have got a set of vectors u 1 n u 2 n dot dot dot uLn, L number of vectors and each of the vector starts at what n equal to 0 n u 1 0 u 1 one u 1 2 up to u 1 n that is u 1 n vector I am not writing it out I hope you understand this you already know in the form pattern this formula for xn vector or if you want for your sake let me write this is what data at 0th index data at 1th index dot dot dot data at nth the index k could be 1 to L right.

You form a matrix call Un and suppose there is an external vectors dn also let wn be the column space of Un that is simple that, is space span by this columns instead of writing span of this vector this is a column space this span of with this corresponding pn, pn perpendicular orthogonal projection operator and ortho pro error operator for wn fine.

Pn dn what is this projection on the space span by this guy the column space by this matrix and pn minus 1 dn minus 1 how are these 2 related can you find relations between them it is a projections that is a thing it is not easy it does not come up, so easily you understand that suppose I started counting time n equal to 0 down 1 n equal to 1 and I stopped at n, n equal to n minus 1. I got data matrix that matrix column space I am projecting dn minus 1 vector on that projections. Now, you introduce the data current data U 1 n, U 2 n, U L n for each of the columns.

So, data matrixes gets 1 extra row new column vector just span a space on that I project dn orthogonal I get pndn, obviously, the entire vector is new, because I got a new sets of linear combined coefficients how about the related that is not easy it is not easy.

You remember I define the pining vector phi n 0 0 0 only last guy is 1. Consider a figure that suppose instead of so many columns and I just have only 1 column u 1 only u 1 fellow um.

Suppose, I have a vector u 1 n minus 1 and a 0 like this same. So, 1 0 I put and the vector pi n, pi n will be what 0 0 0 here and 1 here you understand those 2 are orthogonally to each other pi and these 2. So, that means, pi n could be like this 90 degree is a pi n and consider the vector u 1 n what is u 1 n u 1 n is this guy plus the latest component and u 1 n isn't this fellow plus the latest component that will be somewhere here u 1 n. In fact, I should have made it 90 degree, I do not know let take it as 90 degree.

You can see 1 thing u 1 n is simply a linear combination of these 2 what is u 1 n this part is common last guy is u 1 n scalar and here last guy is 1 see if I take this and multiply this by this last scalar fellow add I will get this with last scalar fellow if I take this multiply pi n by that this is a bottom most entry a pi n will be u 1 n all other entries on pi n will be 0s and here I am living it as last entry is 0s and other entries are this and add that 2 and you get this is it not?

That is a space spanned by pi n and this fellow that is included that is sub space of that is sub space of space span by pi n and this fellow.This is a linear combination of these 2. So, the space span by pi n this that is a sub space of that is contain in the space span by pi n and this.

Alternatively, I said this guy again is a linear combination of these 2 how to get these guy from these 2 take last fellow reverse the sign minus multiply phi n by this minus 1 here add that 2. So, last guy entry canceled and we get 0 fellow; that means, space spanned by this and this pi n and this is content in the space spanned by pi n and this because this is linear combination of these 2.

That means space span by pi n and U 1 n is same as the space span by pi n and this, but pi n this are orthogonal see from the same space I want to project somebody instead of giving the space as though it is span by this 2 fellow I can give it to span by these 2 fellows which are orthogonal.

So, that projection will be what summation of projections on this axis and projections on this axis you see only 1 thing I have already got the data vector of the past my purpose is to relate past projections in the current projections I already got 1 data vector already I have taking the case of only 1 column vector basically, I have already brought in pa column vector which is the past column vector past data vector.

But one thing is sure that any external vector projected on space span by pi n, and this p 1 n same as the projection space projection on the space span by pi n and this because that spaces are same. When you take the external vector and projected on the space span by pi on and this it will submission of projections of space here and here.

Now, what is the projection of any fellow on pi n only the last component, last component isn't other value multiply by 0 last components into 1 and multiply by 1 no lambda. So, last component.

And similarly that external vector dn projected on this will be what dn and this inner product, inner product between dn and this last entry you forget isn't other entries will be multiplied dn will be what a vector like this dn minus 1 vector and dn. So, dn and 0 will cancel in a inner product I am able to take this guy and this guy lambda will come up if we take the inner product last 2 entries multiplied with 0 fine, but last, but 1 entry there lambda will come up take lambda common lambda lambda square up to power n, but lambda common see if you take 1 lambda common if it is set to the inner product between this guy and this guy isn't.

So, that means, what is the overall projection, projection on this I will be doing it elaborately, but I am just telling this is just through this figure I am giving it what is the projections of dn vector on this space; one is the projection on this pi n fellow which is 0 0 vector and only a scalar dn, and what is the projection on this that is projections on this dn minus 1 that being inner product of this whatever doing what is the projections of orthogonal projections of dn vector on this dn inner product with this s divided by norm square of this into this isn't that inner product last guy goes remaining inner product is lambda times the inner product between dn minus 1 vector and dn minus 1 vector divided by norm square of this fellow norm square of this fellow will be what lambda times is norm square into this fellow alright lambda will cancel there.

So, that means as though. So, the projections last component is 0 we are projecting this what is the inner product divided by norm square into this guy. So, last component is 0 this guy multiplied by constant is it not? Let me do it here for your sake 0 dn and you have to project this on this you have to project this on this what I am saying is this that projection will consist of what this vector first this vector multiplied by constant what is that constant inner product between these 2 divided by norm square of that lambda will canceled you have seen inner product between this 2 means what lambda times inner product between this and this.

Norm square means norm square of this ratio. So, is in it what you get when you are doing the same projections at n minus 1th index as got dn minus 1 vector was projected on u 1 u 1 n minus 1 vector isn't um.

So, you get that term that is dn minus 1 vector inner product u, u 1 u 1 n minus divide by norm square of the u 1 n 1 minus into u 1 n minus 1. So, basically this is estimate at index n minus 1, estimate means dn minus 1 it projected orthogonal this vector that is u 1 n minus 1 what you get is this and then 0.

You understand this tomorrow I will next time I repeat this again have you understood this part.

Inner product between these and I am projecting dn and on the this axis. So, I have to take this orthogonal I mean this vector as it is this 0 is this 0 as it is and this will multiplied inner product by his 2 divided by norm square lambda will cancel inner product last guy will disappears since lambda canceled this is simply the inner product by these 2 guys divided by norm square of this guy.

That times this u 1 n minus here there is nothing but projection of dn minus 1 vector on u 1 n minus 1 vector it is an estimate based on data up to n minus 1 along this axis divide 0 this is the overall projection.

So, what is the over projections error the submission has to be subtracted from dn vector an immediately this guy will be disappear isn't and dn minus 1 is what is the error, error will be it you called error will be dn minus this summation. Summation is what last guy goes and 1 dn is what na dn 1 minus vector and then dn, dn cancels dn 1 minus vector minus is estimate that is the error, that is error which error dn minus the dn minus 1 minus on the projections on this axis that error not on the overall projection.

Okay, this relation will be using in more generalized from next class you will see how it comes there this I just is said diagram I will again start from here just come prepared with this little bit it is not easy, but very interesting.

Thank you very much.