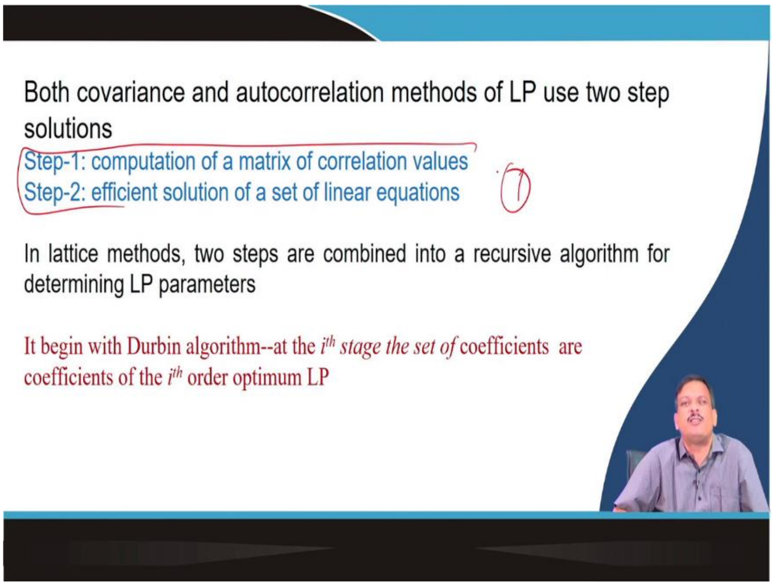
Signal Processing Techniques and Its Applications
Dr. Shyamal Kumar Das Mandal
Advanced Technology Development Centre
Indian Institute of Technology, Kharagpur

Lecture - 48
Lattice Formulations of Linear Prediction

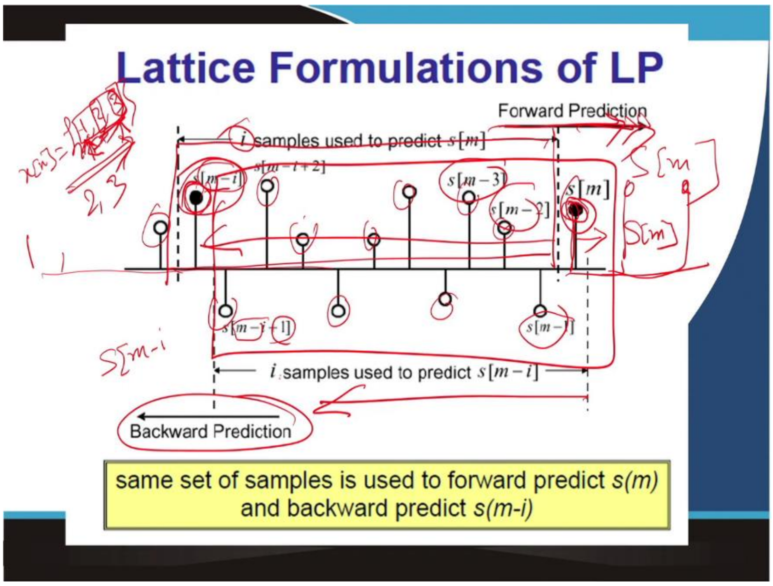
So, now we go for that lattice formulation of linear lead prediction. So, what is there?

(Refer Slide Time: 00:26)



So, as I discussed, the computation of the matrix of correlation value or autocorrelation, whatever the covariance or correlation, does not matter. So, that requires a complex and efficient solution set of linear equations; that means an iterative method is the requirement. So, can I combine these two steps into one step? So, can I combine this step into one step? So, that method is called the lattice formulation method. Let us go for that. What is that?

(Refer Slide Time: 01:07)



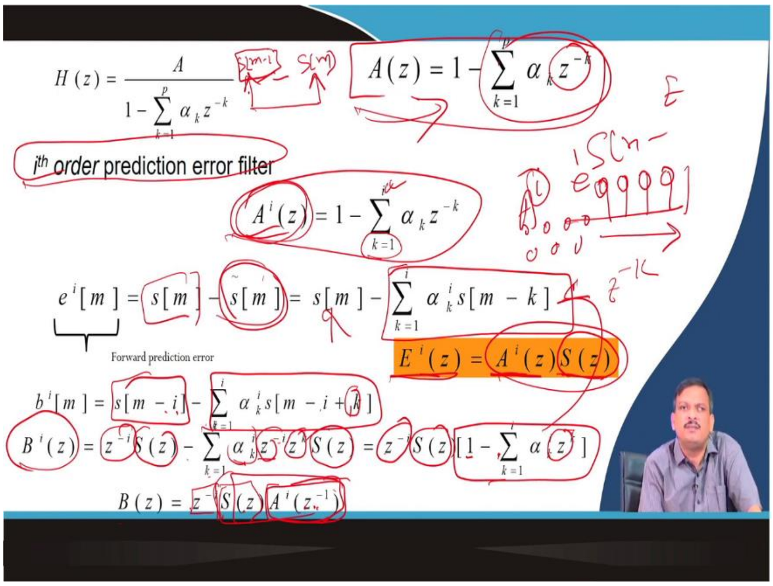
Let us say. So, when I say the prediction, let us say I have a sample. So, let us say I can say this is a signal which is the many samples are there, many samples are there. So, this is my most number of samples. So, if I say I want to, I will use the sample to predict s[m] sample. So, I want to predict this sample from the previous i^{th} sample. What is the meaning? Let us give an example; suppose I have a signal x[n] 1 2 3. Let us say I want to predict 3 based on our previous 2 samples.

So, here also I am predicting s[m] based on the previous i^{th} sample. So, what is the I s[m-1] s[m-2] s[m-3]? Ultimately, s[m-i] was the previous i^{th} sample. So, when I predict that way, it is called forward prediction. So I can, I can say that I am predicting 3 from the previous 2 samples. Also, I can say I can predict one from the past, I can say that from this 2 and 3.

So, prediction can go this way, once it goes from 1 2 3, that is called forward prediction. This way, I can predict the next sample. So, this is a forward prediction. I can predict the backwards. also, I can predict 1 from 2 and 3 because x[n] is known. So, once the x[n] is known, I can predict both ways. So, when I predict this way, suppose I want to predict s[m] I minus 1 sample from the previous, or I can say this side with the sample. Understand or not?

So, I am predicting this sample from this previous sample, or I can predict this sample from this previous sample. So, when I predict s[m] from a previous i^{th} sample, that is forward prediction; when I predict this side, this is called backward prediction. So, prediction can be forward, and prediction can be backward. I can predict 3 from 1 and 2, I can predict 1 from 2 and 3. When I predict 3 from 1 and 2, it is a forward prediction; when I predict 1 from 2 and 3, it is a backward prediction.

(Refer Slide Time: 04:16)



So, there is a forward prediction and a backward prediction. Ok, do you understand that part? Now I come to that: what is the equation of the forward prediction, and what is the equation for backward prediction? So, what is my prediction filter? This one. So, if I say i^{th} order prediction error filter. So, this is nothing but a prediction error filter s n minus the previous sample. This is the previous sample sum. So, let us say i^{th} order prediction error filter.

So, Ai(z) is equal to 1 minus k, equal to 1. So, I can say i^{th} iterative. So, this I represent the iteration, and this I represent the order. As I said, suppose there is a sample. So, the prediction error is maximum when I am predicting this sample from the previous 14th 0 sample. So, in the first iteration, I am predicting this sample: the second iteration, this one; the third iteration, this one; and the fourth iteration, this one.

So, once I go to the iteration, that error will be minimized. Ok or not? So, let us say i^{th} iterative iteration process is this one. So, what are the prediction errors? Ei(z) is equal to s[m] signal minus predicted signal. So, s[m] is my predicted sample. So, in the z transform, I can say Ei(z) is nothing but an Ai(z) multiplied by S(z), and S(z) is the input signal. Now, what is backward prediction? I am predicting m. So, what am I predicting forward? In forward prediction, I am predicting s[m]; in backward prediction, I am predicting s of m minus I, if the i^{th} order prediction is used.

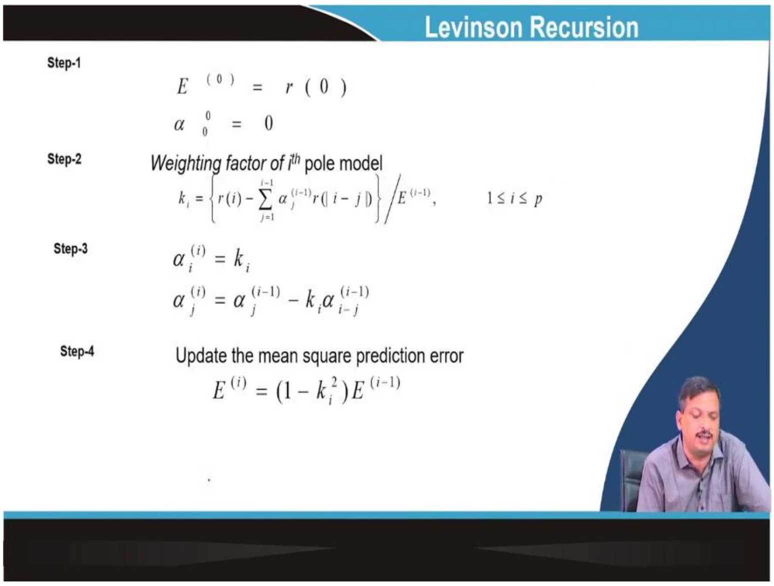
So, I am predicting this sample forward prediction, I am predicting this sample. So, this is minus the estimated sample. So, backward prediction similarly this is my signal minus estimated signal s[m] I minus so that one plus 1 plus 2 plus 3. So, that is why plus k if you see. This is s[m-i]. This is s[m-i+1]. Suppose I represent by s of. So, suppose this is the third, so I am predicting 1. So, this is nothing but a first sample. So, I can say the 2 minus. So, this one is the next part of this.

So, 1 will be added to this. So, that is why this is added k is added. Now, if I take the z transform, so, this will be B(z) this will be z^{-1} multiplied by S(z) m minus i minus k equal to 1 to k equal to 1 to i α k i z i, so z minus i z plus k into S(z). Now, if you look at this, k is equal to 1 to i. So, I can say that z^{-i} is independent of this sum. So, z^{-i} is taken out, and S(z) is taken out. So, this is 1 minus this one.

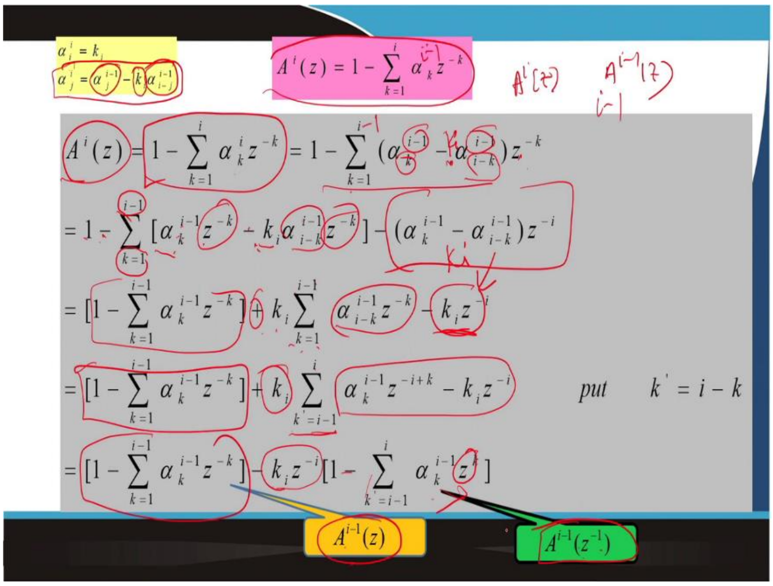
Now, if you see this one and this one are almost the same; the only difference is that here z is z^{-k}, here it is z^{k}. So, I can say that instead of A(z), this can be $A(z)^{-1}$. So, I can say B(z) is nothing but $A(z)^{-i}$ multiplied by S(z) into $A(z)^{-1}$. So, E(z) is equal to Ai(z) S(z) . So, it is Ai(z) z^{-1} S(z).

So, this portion, this portion, except this error filter, z^{1}, this z^{-1} and multiplied by z^{-i}.

(Refer Slide Time: 08:54)



(Refer Slide Time: 08:56)



So, now, if I say that I want to write down this one, how can I write down Ai(z) in terms of the previous error? So, what Levinson said is that recursion is possible. What recursion is possible? I can say that Ai(z) can be predicted from A i minus 1 z. So, I want to write down that part. How can I include recursion Levinson recursion so that I can calculate Ai(z) from the previous value? So, I know Ai(z) is this one. Now I know α k I; this is the Levinson recursion equation. A j i is equal to the previous iteration result minus k i with the Parcor coefficient and α previous iteration.

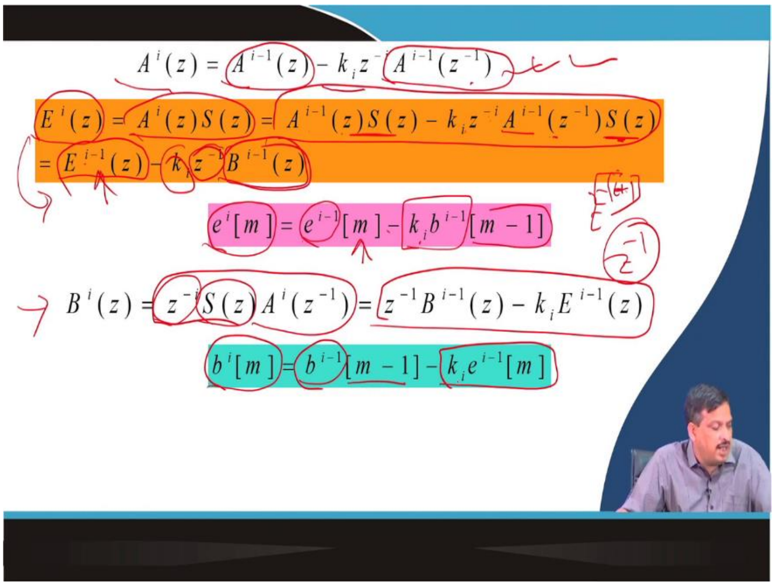
So, I just put this one instead of j; I said k and I minus 1 would be there. So, I can say α k I with order; that means ith iteration can be written as I minus 1 iteration k instead of j I am k k writing k. Now, if you see this is k equal to 1 to i. Let us say I said this summation will be i minus 1. So, k equal to one to i minus 1 α k i minus 1 z^{-k} minus k i minus.

So, this is nothing but an α that will be k I. So, k i into α i minus k z^{-k} multiply and what I have written one term I have not written. So, this is my last term. So, this is nothing but a k i. So, in the second stage, I writing this one as a k i, and this one if you see minus. So, I can say 1 minus this one and minus plus k I into k equal to 1 to I minus 1 this one minus k I into zi z^{-i}.

Now, I can say this is one term, and if I say k i, it is because summation is not independent of i. So, k i is this side k equal to i minus 1 to i. I can put this one, okay? So, I put k dash equal to i minus k i minus k is equal to k dash ok. So, when k is equal to 1, it is nothing but I minus 1. When k is equal to I minus 1, it is nothing but i. So I can write down this one. Now I can see this one is nothing but an A i minus 1 z.

Because Ai(z) is equal to this one, A i minus 1 z is nothing but a α k i minus 1. So, A i minus z, and if I say k i and z^{-1} is outside, then it becomes 1 minus this one. If you see this, it is z^k. So, if I make z minus k. So, it is nothing but an A i minus 1 z^{-1}.

(Refer Slide Time: 12:34)

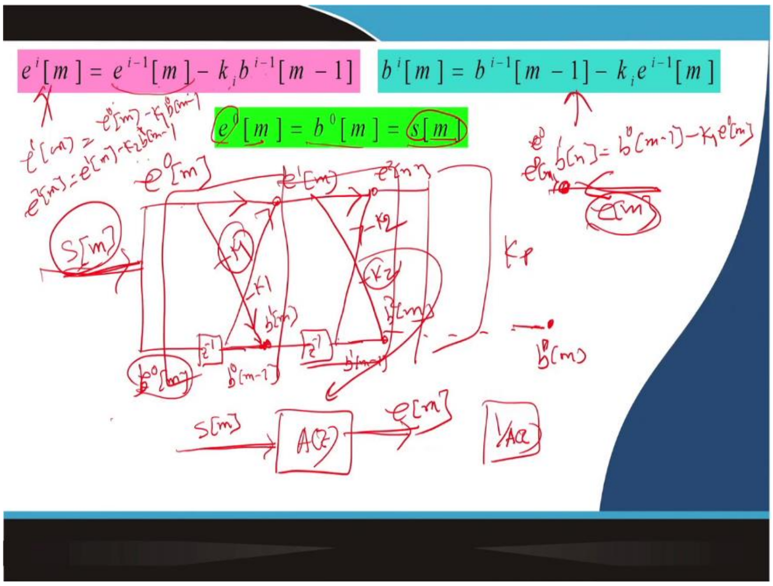


So, I can say that Ai(z) is nothing but a A i minus 1 z minus k i into z^{-i} A i minus 1 z^{-1}. Now, if this is my error prediction, error filter, the prediction error is nothing but multiplied by the signal I put this S(z). So, if you see A i minus 1 z S(z), I can say it is nothing but an E I minus 1 z and A i minus 1 z to the power z minus 1 S(z), which is nothing but a B(z), B(z) is this one only in this form.

So, I can say I write down the B z, but one thing, this is z i minus 1. So, I required z^{-1} because I required z i minus 1 previous one. So, 1 minus z to the minus 1 will be written. So, it can be i minus 1 plus 1, or I can say if i is common j i minus 1. So, that is why 1 z^{-1} will remain, and k I will remain, and this B I will be minus z. Now, from this equation, I can say that Ei(z) is in the time domain.

So, this is in z domain i minus 1 E i minus 1 m minus k i into B i minus 1 m minus 1. Understand or not? Similarly, if I put Bi(z), so z^{-1} S(z) A(z) to the power this one. Again, I can say that this is nothing but this one. So, which is nothing, but a b i m is equal to b i minus 1 m minus 1 k i into e i minus 1. So, now say that this is my equation forward prediction error. This is my backward prediction error.

(Refer Slide Time: 14:52)



Now what? So, I can say that ith iteration is nothing but I minus 1 iteration, so if I say 0th iteration. So, I equal to 0 e0[m] is nothing but a s[m]. There is no iteration. So, the error is nothing but a signal itself; the signal itself is the error. So, it is nothing but a s[m]; similarly, b0[m] is nothing but s[m]. So, I can say that when I do the signal flow diagram, I can say let us know if this is my speech signal s or a signal s[m].

So, I can say this is nothing but an e0[m], this is nothing but a sorry, this is nothing but a b0[m], this is nothing but an e0[m]. So, this is the 0th-order prediction and 0th iteration. So, this is nothing but a signal itself. Then what is e one m? So, I know this equation e one m is nothing, but an e0[m] will be there minus k. So, i equal to 1 minus k1 equal to b 0. So, e0[m-k] 1 into; k1 into b0[m-1].

So, this is b0[m]. So, how do I synthesise b0[m-1]? I put $A(z)^{-1}$ delay. So, this is nothing but a b0[m-1]. So, this will be added up with this one with a minus k1 coefficient. What is k? k is nothing but a partial reflection coefficient or Parcor, a partial correlation coefficient. If it is on the go for tube model, it is called a partial reflection coefficient.

So, it acts as a partial reflection coefficient. Parcor partial correlation coefficient is. Now, what is b1[m]? b1[m] is equal to b0[m-1] minus k1 e0[m]. So, I can know that b 0. So, this is my b1[m]. So, b1[m] will be this one will be multiplied minus k1 and added up with this one, I get b1[m]. Now, when I say e 2 m is nothing but a e 1 m minus k2 into b 2 b1[m] minus 1. So, b1[m], I know.

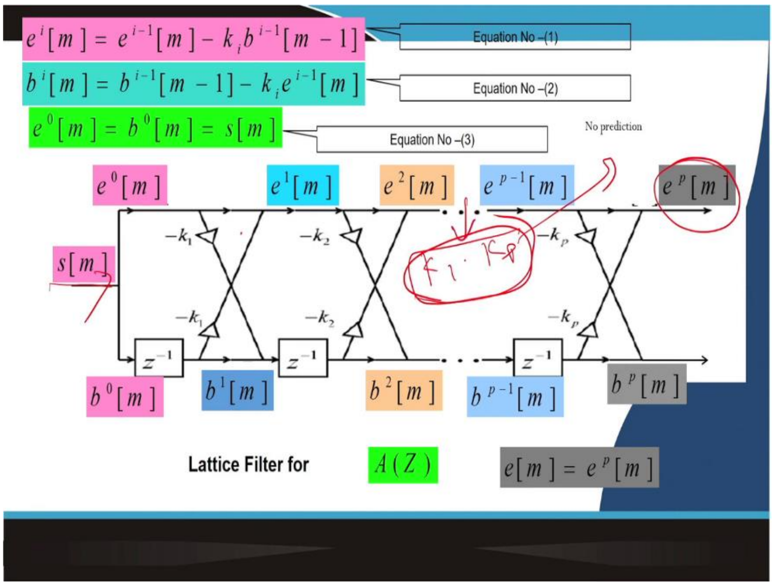
So, that will be delayed by 1 sample, b1[m] minus 1 I get. And this is my e 2 m, e 2 m is this signal multiplied by this signal with a minus k1 k2 and b2[m] is nothing but a this will be a multiplied by minus k2. So, that way, I can say this is nothing but a, let us say, pth order. So, ep[m] I will get, and I will get bp[m]. So, on this side, I will get bp[m], and on this side, I will get ep[m]. So, what is bp[m] and ep[m]? bp[m] is nothing but a final error.

So, the final error is nothing but a bp[m] or ap[m]. Whatever I can say, that is nothing but a final error. So I can get the final error. So, what I said? I have a filter. If I apply s[m], and this is what I implemented here, I will get e m. So, what do I have to know? I have to know k1 k2 kp. So, let us say all k values. Once I know all k values, I can implement that signal flow diagram on a computer, and if I apply s[m], I can calculate e m.

So that is called lattice implementation because if you see one block. This block, there is another block, there will be another block. So, each block is simple, only k1. So, I can say the one for loop only if you say the for loop is repeated for k1 k2 k 3 k 4 k p. So, the p stage will be there for pth order prediction. So, I can implement A(z) immediately if I know k1 k2 k p. So when I say synthesis, let us say I want to implement a synthesizer; that means which is nothing but a one by A z.

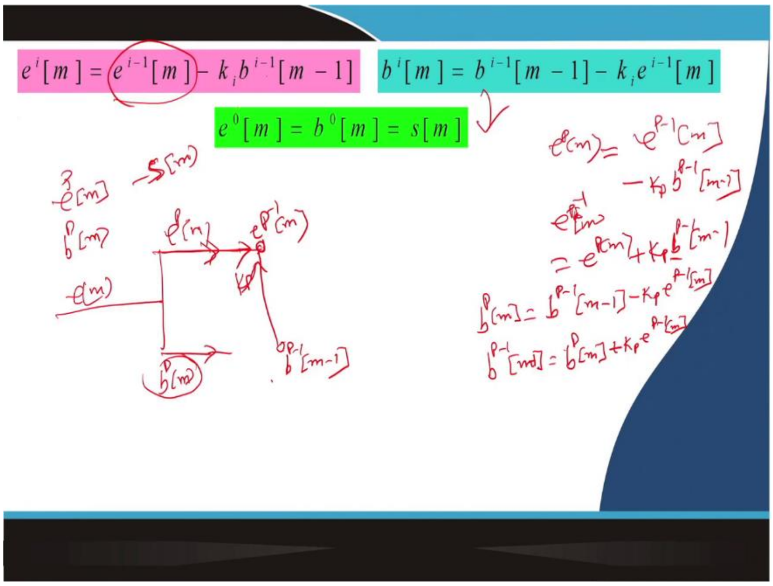
So, here I am, applying for s[m] and getting m. Now, if I see the signal flow is reversed. So, I apply e m. I should get back my s[m], but the signal flow will be in the reverse direction. So, how do we do that how do I do the reverse direction signal flow? This is the diagram that I did.

(Refer Slide Time: 20:44)



So, I want to draw the reverse direction signal flow. Let us say I just deleted this one, and again, I want to draw it. So, what I said is if I apply s[m] input, I get e m.

(Refer Slide Time: 21:09)



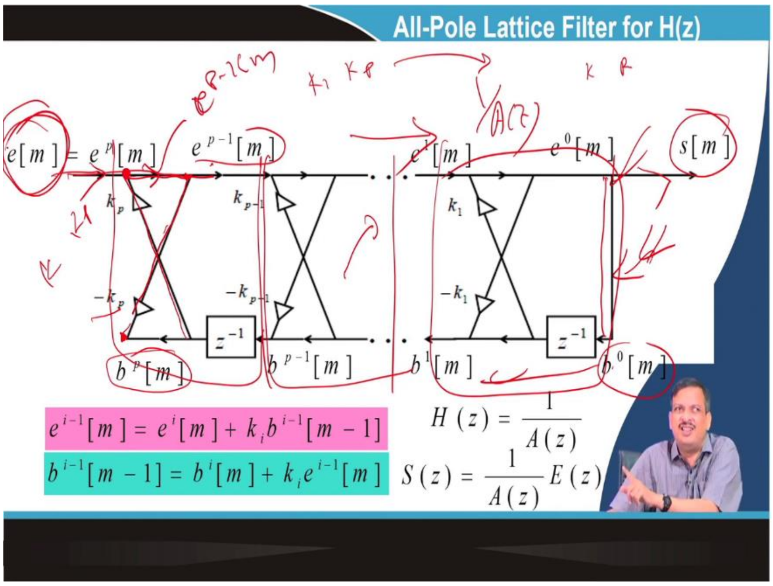
Now, my target is to know e m, ep[m], and bp[m]. I have to get back the signal s[m]. So; that means, I know ep[m], I know bp[m]. So, all are ep[m] is equal to bp[m] at the final stage. So, let us say I know e m. So, which is nothing but an ep[m], here is nothing but a bp[m]. Now, what do I want to know? So, what is known? I know ep[m]. What do I want to calculate? ep minus 1 m. So, I can say that ep[m] is equal to ep minus 1 m minus k p into bp minus 1 m minus 1.

Now, what do I want? I want ep[m]. So, I said the ep[m] ep minus 1 m is nothing but an ep[m] plus k p into bp minus 1 m minus 1. So, let us say this one is my point minus 1 m at this point. So, how do I know? ep minus 1 is nothing but the ep[m]. So, this signal directly goes here, and I required k p multiplied by bp minus1. So, let us say this one is b, this one is bp minus1 m minus 1, let us say this one.

So, this one will be multiplied by plus k p, and I will get up minus 1 m. So, how do I get bp minus1 m? So, from this equation, I know bp[m] is equal to bp minus 1 m m minus 1 minus k p into ep minus 1 into m. Now, I say bp[m] p minus 1 m is equal to m minus 1 is equal to bp[m] plus k p into ep minus 1 m.

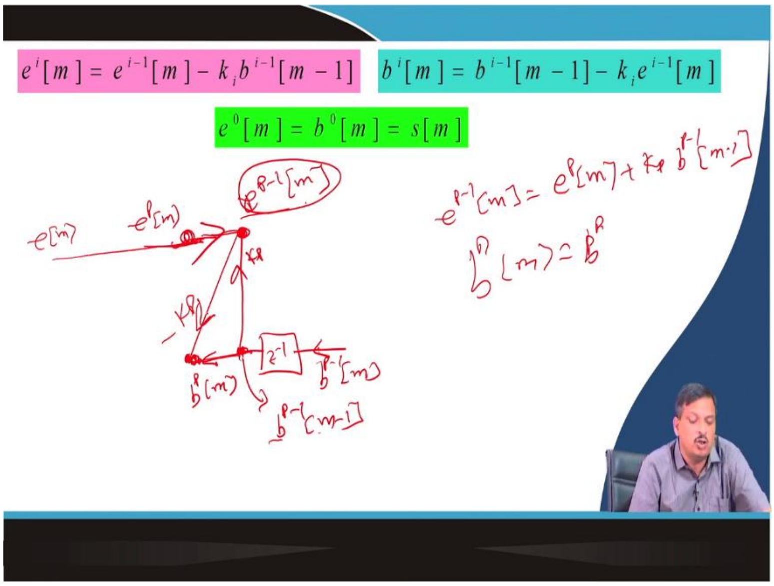
So, if I know ep minus 1 m, if I know b p. So, this one bp[m], I know bp[m]. So, if I want to know if the BP is minus 1, I have to delay the signal. So, what is required? So, what can I say? I am okay. Let us say the bp[m] I want to generate.

(Refer Slide Time: 24:24)



Let us say I want to generate bp[m] instead of bp[m], and I am generating bp[m], okay?

(Refer Slide Time: 24:36)



So, what will I do? I will draw it fresh; what I know I know let us say I know e p e m is known ep[m] is known ep[m] is known. And this one is, let us say, bp[m]. So, if this one is delayed by z^{-1}, then this one is bp[m]. This one is nothing but a. So, this is delayed by z^{-1}. So, this one is nothing but a one-sample delay, ok? So, this is bp minus 1 m delayed by one sample. So, at this point, I get bp minus1 m. This point lets us know this is bp[m].

So, let us say this is bp minus 1 m. So, I know ep[m]. Let us say this one is my ep minus 1 m. So, ep minus 1 m is nothing, but this sample multiplied by this will be multiplied by k p. Understand or not? So, this point is my ep[m], and this point is my ep minus 1 m. So, this point is bp minus 1 m minus 1 multiplied by. So, I know that ep minus 1 m is equal to ep[m] plus k p into bp minus 1 m minus 1.

So, I know this is bp minus 1 m minus 1. This will be multiplied by k p, and I get up [m]. Now, what is bp[m]? It is nothing but a bp minus 1. So, this is bp minus1, and this will go here minus. So, this is nothing but an ep minus 1. So, I can say this will go here: multiply minus k p. This is nothing but a bp[m]. Understand? So, the ultimate signal flow diagram will be what? This is ep[m]. So, this is my plus k p.

So, I get ep minus 1 m. After this point, it is ep minus 1 m. So, from that, the only thing is the signal diagram. So, from here, if I take up minus 1, I am taking multiplied minus k p. I get bp[m] in here. Understand? So, if I apply e m here, I get output s[m]. Because I know e 0 and b 0 is the same.

So, s[m], so this is backward, and this is forward signal flow. So, why am I implementing it? I am implementing one by A(z). first, I am implementing A z; now, I am implementing one by a z. Understand? Again, it is also a lattice, 1 lattice. Understand?

So, now, if I apply the error signal if I know k1 k2 k p when I say the voice is transmitted. So, I analyzed, and I said k1 to k p. I extracted and transmitted in the receiver side from a k1 to k p. I can generate this filter, and I can pass the e m. What is e m? e m is nothing but a glottal excitation in the case of voice.

So, that is nothing but an impulse. So, if I pass an impulse, I can get the speech signal back again. Is it clear? So, that is called lattice formulation. Now, how do I get the k1 value? Can I calculate the k1 value using auto correlation? Yes, you can do that. But again, I have to do autocorrelation, and also I have to implement the filter.

So, on the synthesizer side, I know that I have to implement the filter only with these methods, but on the analyzer side, this is how I use it for the analyzer. So, how can I directly calculate the k value from bp[m] and ap[m] and how can I directly implement it? So, how can I directly calculate this k1 to k p-value? One solution is that if I know the k1 and k p values, I can synthesize the signal s[m] from the error signal. In ideal cases, the error signal is 0, and the error is minimized. Understand? So, I can synthesize the signal from the error signal. So, I have to know k1 k2 k 3 k p.

Now, when I am analyzing this diagram, I am inputting the s[m] signal I can calculate the error signal if I know k1 k p. So, how do I extract the value of k1 and k p? I can extract those values using auto correlation ok no problem, I can extract those things. But can I directly compute that k1 and k p value? So, in the next class, I will describe how I directly calculate the k1 and k p values, and I give an example. Let us give a signal, and I can give you an example of how it can be implemented in computer programming,. You should know, ok.

Thank you.