Mathematical Methods and Techniques in Signal Processing - I

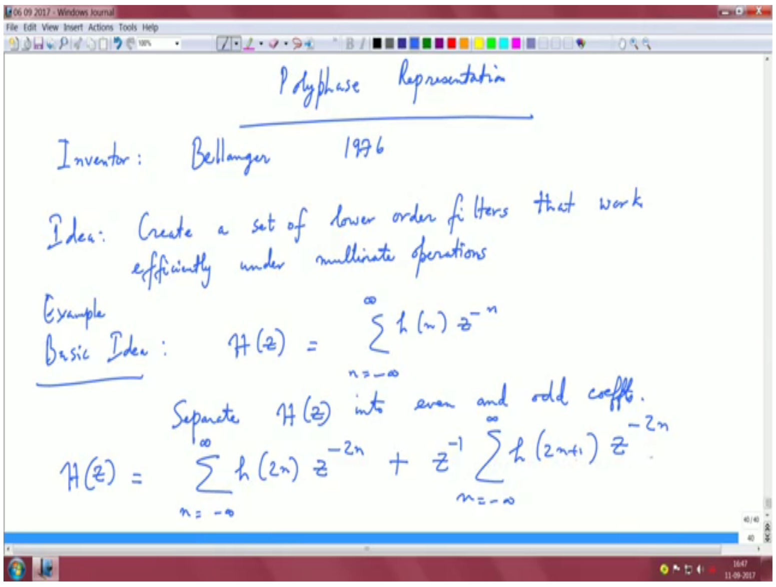
Prof. Shayan Srinivasa Garani
Department of Electronic Systems Engineering
Indian Institute of Science, Bangalore

Lecture – 40
Polyphase representation

So, we have seen through the last few lectures, that we have these down sampling and up sampling operations followed by filters right I mean there are there are equivalence etcetera. So, in the down sampling process we are reducing the sampling rate, in the up sampling process we are inserting zeros right. So, we may up sample and then we may pass them through filters, towards the purposes of interpolation

One of the questions that you may ask is, am I not wasting my computations because I have a lot of these zeros that I am inserting because of the up sampling process. And when I am filtering and basically filtering these zeros right my computation is basically wasted by filtering these zeros, can I do something efficient towards this purpose? And this problem was an issue way back in the bell labs, when people were investigating into multi rate filters essentially towards adjustable multi-level filters or in possible applications of TDM FDM conversions etcetera, we had to deal with these adjustable sampling operations followed by filters and you have these situations where you have these huge filters and you want to basically filter zeros through such filters, it is a computational burden has huge latencies right you suffer on the throughput it infuses latencies now it is basically not efficient.

(Refer Slide Time: 02:14)



Polyphase Representation

Inventor: Bellanger 1976

Idea: Create a set of lower order filters that work efficiently under multirate operations

Example
Basic Idea: $H(z) = \sum_{n=-\infty}^{\infty} h(n) z^{-n}$

Separate $H(z)$ into even and odd coeffs.

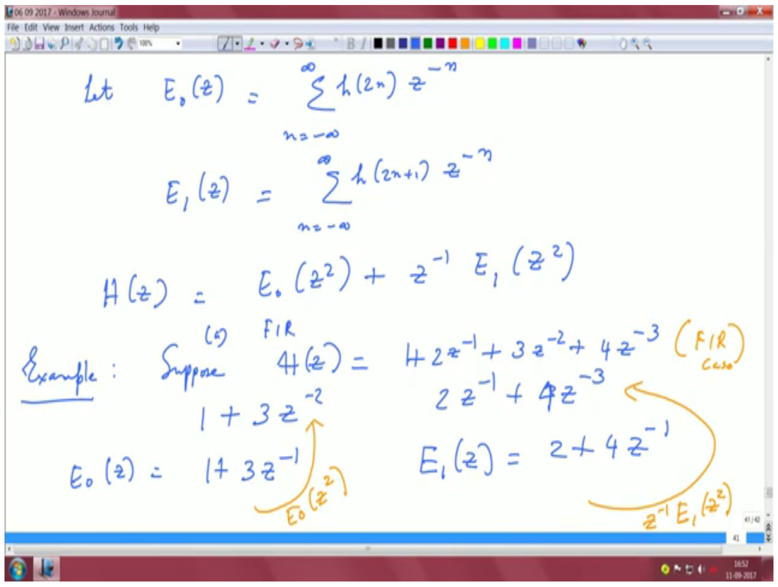
$$H(z) = \sum_{n=-\infty}^{\infty} h(2n) z^{-2n} + z^{-1} \sum_{n=-\infty}^{\infty} h(2n+1) z^{-2n}$$

So, fortunately a researcher thought about these problems, and that was maurice Bellanger and his team in 1976. So, the idea in polyphase representation is create a set of lower order filters, that work efficiently under multi-rate operations and this idea also leads us towards simplified theoretical results and has a lot of very interesting consequences for designing efficient architectures for multi rate systems.

So, let us start with a basic idea. So, it is this is the original idea is to create a set of lower order filters, that work efficiently under multi rate operations. So, let us start with a basic idea I mean an example I would say. Now we have H of Z is basically the Z transform of h of n. So, if h of n are my filter coefficients right in the transform domain, I can write it in this form this is no big deal.

But what is interesting is, we will follow the divide and conquer approach that is often seen in computer science right. We separate H of Z into even and odd coefficients. So, we do this then H of Z can be written as follows we have a sum of n equals minus infinity to plus infinity h of 2 n Z power minus 2 n. So, at all even points I am I am getting this I am grouping all these terms of this filter and then I group all the odd; odd parts in this form, I just wrote it as Z power minus 2, n here this is basically h of n Z power minus 2 n plus h of 2 n plus 1, Z power minus 2 n plus 1 that the additional factors Z power minus 1 I just pulled it outside here ok.

(Refer Slide Time: 05:46)



Let $E_0(z) = \sum_{n=-\infty}^{\infty} h(2n) z^{-n}$

$E_1(z) = \sum_{n=-\infty}^{\infty} h(2n+1) z^{-n}$

$H(z) = E_0(z^2) + z^{-1} E_1(z^2)$

Example : Suppose $\overset{(5)}{}$ $\overset{\text{FIR}}{H(z)} = 1+2z^{-1}+3z^{-2}+4z^{-3}$ (FIR Case)

$1+3z^{-2}$ $\quad$ $2z^{-1}+4z^{-3}$

$E_0(z) = 1+3z^{-1}$ $\quad$ $E_1(z) = 2+4z^{-1}$

$E_0(z^2)$ $\quad$ $z^{-1}E_1(z^2)$

So, now let us see how this is important. So, if we let E naught of Z to be h of 2 n Z power minus n and even of Z to be h of 2 n plus 1, Z power minus 1 if I let these 2 filters like this then my H of Z can be written as E naught of Z square plus Z power minus 1 E 1 of Z square right. It is pretty straightforward you do not touch this h of 2 n or h of 2 n plus 1 instead Z is replaced by Z square.

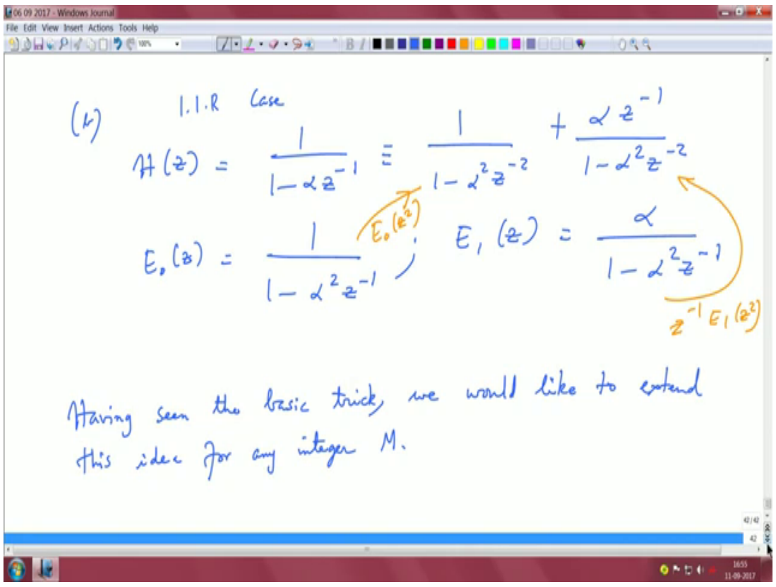
So, now what is happening here, we have decomposed the original filter into 2 sub filters that are potentially that could be potentially of lower order? So, we will see this with an example to make it more clear say suppose H of Z is 1 plus 2 Z power minus 1 plus 3, Z power minus 2 plus 4, Z power minus 3. You first group the terms, this is one part the other part is 2 Z power minus 1 plus 4, Z power minus 3 right and the E naught of Z is basically 1 plus 3 Z power minus 1 and my E 1 of Z pull the Z power minus 1 outside is 2 plus 4 Z power minus 1. E 1 of Z square will be 2 plus 4 Z power minus 2 and Z power minus 1 times even of Z square will give you this and then E naught of Z square will give you this right just to give you from here I can go to in order Z square will give me this get this I have Z power minus 1 times E 1 of Z square.

Now, the aim at least the goal is if I not use lower order filters somehow, instead of this higher order filter and some of use lower order filters in parallel to do certain operations and maybe I am cutting down my time, in my filtering process right. If I have to filter take a signal and filter it through this filter H of Z, I have a latency because I have to go through the delay of this filter. So, I incur latency, but instead of that I cut down this into 2 sub filters of lower orders and maybe let them work in parallel and somehow combine the outputs from each of the filters in parallel, you know can I improve my computational efficiency.

So, we now have an idea here right like even in cs 4 sometimes, it may be in computer science, you may see the scenarios where you might divide and conquer right you have a computation problem, you will just separate them into even parts and odd parts or some 2 parts depending upon what your constraints are and then you do the you carry the computation separately on each of the parts right.

So, the same idea is sort of extended in some sense to the filtering operations and we will see how this polyphase, this is basically 2 phase representation. So, we looked into the case of FIR, we can think about the case this is FIR case finite impulse response we can think about the scenario for the IIR case as well.

(Refer Slide Time: 11:00)



(b) I.I.R Case

$$H(z) = \frac{1}{1-\alpha z^{-1}} \equiv \underbrace{\frac{1}{1-\alpha^2 z^{-2}}}_{E_0(z^2)} + \underbrace{\frac{\alpha z^{-1}}{1-\alpha^2 z^{-2}}}_{z^{-1}E_1(z^2)}$$

$$E_0(z) = \frac{1}{1-\alpha^2 z^{-1}}; \quad E_1(z) = \frac{\alpha}{1-\alpha^2 z^{-1}}$$

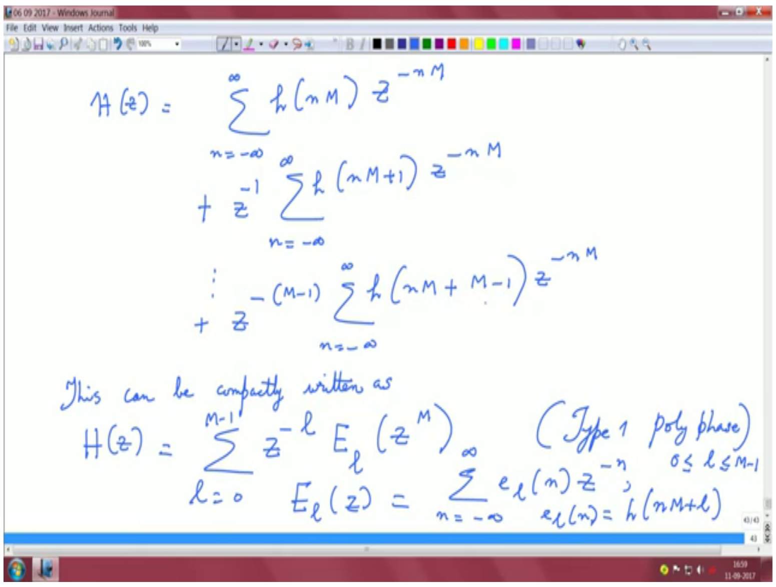
Having seen the basic trick, we would like to extend this idea for any integer M.

So, this is a case a and this case b which is the IIR case. So, suppose I have H of Z to be the simple first order IIR, low pass filter, I can write this in this form; this is saying that this is a plus b divided by a square minus b square nothing more right its very simple to see this structure.

Now, my E naught of Z is 1 upon 1 minus alpha square is Z power minus 1 and E 1 of Z is basically alpha upon 1 minus alpha square Z power minus 1 right. These are 2 sub filters in the same form basically if I wanted to go back from here I could get to this step with E naught Z square, and from this I could go to this step we have Z power minus 1 E 1 Z square right. We look at both the cases for the FIR as well as for the IRR case.

Now, we would like to extend this idea, we would like to extend this idea for any integer any integer M. So, let us see the generalization.

(Refer Slide Time: 14:02)



$$H(z) = \sum_{n=-\infty}^{\infty} h(nM)\, z^{-nM} + z^{-1}\sum_{n=-\infty}^{\infty} h(nM+1)\, z^{-nM} \;\vdots\; + z^{-(M-1)}\sum_{n=-\infty}^{\infty} h(nM+M-1)\, z^{-nM}$$

This can be compactly written as

$$H(z) = \sum_{l=0}^{M-1} z^{-l} E_l(z^M) \qquad (\text{Type 1 Poly phase}) \quad 0 \le l \le M-1$$

$$E_l(z) = \sum_{n=-\infty}^{\infty} e_l(n) z^{-n}; \quad e_l(n) = h(nM+l)$$

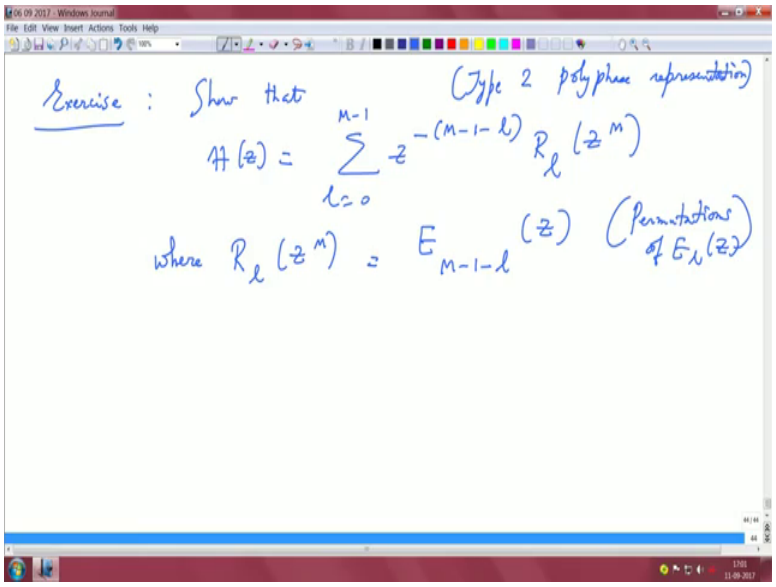
Now, I can write H of Z as summation n equals minus infinity to plus infinity, h of n times M Z power minus n times M. So, basically I look at modulo M components right I just basically have a delay followed by by this dot dot dot plus with M minus 1 delays, we have right. It is just grouping into even and odd for the case of split of 2 if you can split by 3 we take modulo 3. All those with modulo 0 modulo 1 modulo 2 and we grab those filter coefficients and then we regroup the filter as follows right and this can be compactly written H of Z is sum l equals 0 to M minus 1 some delay Z power minus l times, E suffix l Z power M and this is basically type 1 polyphase and the term polyphase indicates multiple phases if it is just a 2 phase system it is even odd, 3 phase 0 1 2, for phase 0 1 2 3 so on right M phase 0 1 2 so on till M minus 1.

So, this whole this is function we this filter H of Z can be written in this form very compactly basically we are folding there delay also and then this is a unit delay followed by some filter so on with an M minus 1 delay followed with this filter. So, the Z power minus M of a basic filter which is E l of Z m is where we can write where E l of Z we will link it with the be the impulse response of the filter, which is.

So, E l of Z is given by this quantity, E l of n Z power minus n and E l of n for the lth phase is basically h of n M plus l, where l is between 0 and n minus 1 this is pretty straightforward el we start with this E l, which is our it is nm plus l equals 0 we have this l equals 1 we have this l equals M minus 1 we have this right and then this is our el of n our E l of Z is basically Z transform of E l of n, and everything just falls in a straightforward way.

So, now this gives us an idea that given any arbitrary filter, we can do an n M polyphase representation of the filter; that means, we take the filter we decompose it into sub filters of lower order which will be connected through a delay line right the original filter is connected to all to all the sab filters we have a delay line.

(Refer Slide Time: 18:58)



Exercise : Show that (Type 2 polyphase representation)

$$H(z) = \sum_{l=0}^{M-1} z^{-(M-1-l)} R_l(z^M)$$

where $R_l(z^M) = E_{M-1-l}(z)$ (Permutations of $E_l(z)$)

Now, one of the exercises or homework exercises is show that there is an alternative representation to this filter, which is termed as type 2 polyphase given by this representation, where you look at the permutations of the filters E l of Z and then you connect them to the original filter and this is this representation is basically called type 2 polyphase type 2 polyphase representation and a remark that we need both of these 2 representations, when we discuss how we want to bring in efficient architecture sometimes you want type 1 representation sometimes you may want type 2 representation to simplify your architecture.

So, therefore, you need to know both of these representations and this is nothing really fantastic here, but it is just basically if you if you look into the permutations of vl of Z you can simplify the transfer function right. Just follow this definition and convince yourself that H of Z can be written in this form an alternative way the proof is pretty straightforward its very simple exercise

So, I think with the above tools in mind I mean we have a very powerful tool now which is polyphase representation. So, the concept is take a long filter that you have right and then chunk it into sub filters, and each of these sub filters are connected via delay line and some operation to get back to your original filter we saw that right H of Z is basically e naught of Z square plus Z power minus 1 E 1 of Z square.

Similarly, for you know modulo 3 split, we can have a Z cube term there right e naught of Z cube E 1 of Z cube E 2 of Z cube and combining elements would be these delay elements right. So, we can think about an architecture which is very similar to that and these architectures are very efficient we will realize, when we have to do multi rate operations whether we have to filter first or filter later on, it will cut down the latencies because we are dealing with lower order filters provided we are operating in parallel we can cut down the latencies and second we will see how we can do efficient computation by not filtering these zeroes.

Because if you take these zeros and we filter through an interpolator I mean it takes quite a lot of time right and we can save some of those cycles by doing filtering operations efficiently right and that will be part of the next module.