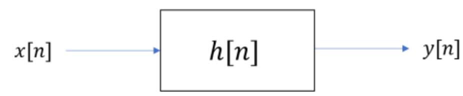
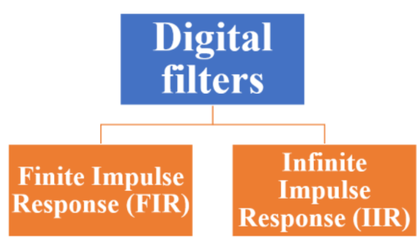
Signal Processing Algorithms & Architecture
Dr. Anirban Dasgupta
Department of Electronics & Electrical Engineering
Indian Institute of Technology Guwahati
Lec 22: Filter Structures - I

Hello everyone, welcome to a fresh lecture on the topic of filter structures 1. I am Dr. Anirban Dasgupta and let us get started. So, what are digital filters? Now, well, you might have come across this kind of system where I say this is defined by some variable $h[n]$, and this is our input $x[n]$, and this $x[n]$, when it passes through the system, gives you an output y[n].



And if I say this system is a linear time-invariant system, or it is sometimes called linear shift-invariant in the case of discrete time systems, then this output y[n] is given by the convolution sum of $h[n]$ and x[n]. and this of h[n] is called the impulse response.

Why? Because if my x[n] equals $\delta(n)$, then y[n] will be h[n]; or in other words, this is the response when the input is an impulse: the output will be an impulse response. Now based on this impulse response, there are two forms of filters. Now, what are filters, by the way? So, filters are anything that alters the nature of the signals in terms of frequencies. Like the frequencies, the content, like the amplitudes of the frequencies in the input signal, will be altered in the output signal.



So, any system can be called a filter which typically alters these. So, it is not about the frequency content, but basically the amplitudes of the frequencies. So, digital filters can be categorized as finite impulse response filters or infinite impulse response filters or in other words, if this is finite, then it is called an FIR filter, and if it is infinite, it is called an IIR filter. So, FIR filters, by the name, suggest that the impulse response is finite; that is, it becomes 0 after a certain number of points. And of course, below 0, I have kept it as 0, assuming that this filter is a causal filter. Causal filter means the input will not depend on future inputs or the output will not depend on future inputs. It will only depend on the present and past values of input. And hence h[0] is 0, H[n] is 0 for negative values of n. So, this is a type of non-recursive filter that has no feedback.

$$h[n] = \{h[0],\ h[1], \dots, h[M]\}$$

$$y[n] = b_0\, x[n] + b_1\, x[n-1] + \cdots + b_M\, x[n-M]$$

$$H(z) = b_0\ + b_1\, z^{-1} + \cdots + b_M\, z^{-M}$$

The output will depend only on the current and past input values and this makes the system inherently stable. Why? Because if you see the Z domain, the Z domain is a polynomial in Z. You can see there is only one pole at Z=0. And that is within the unit circle. So, FIR filters are inherently stable, and this is the generic representation of the FIR filter in the time domain as a difference equation, where in the frequency domain it is a polynomial in z inverse.

IIR filter, on the other hand, the impulse response values can go to infinity; not that they can go, but they definitely go to infinity because if they do not go to infinity, then they will be FIR, and again at the right side, that is positive infinity. But on the negative side, I have restricted them to zeros to make it a causal filter. But if there are non-zero values for n greater than 0, it would not be a causal filter. That means you cannot implement that filter in real time. But of course, you can use a non-causal filter in offline modes.

$$h[n] = \{h[0],\ h[1], \dots\}$$

$$y[n] = a_1 y[n-1] + \cdots + a_N y[n-N] + b_0\, x[n] + b_1\, x[n-1] + \cdots + b_M\, x[n-M]$$

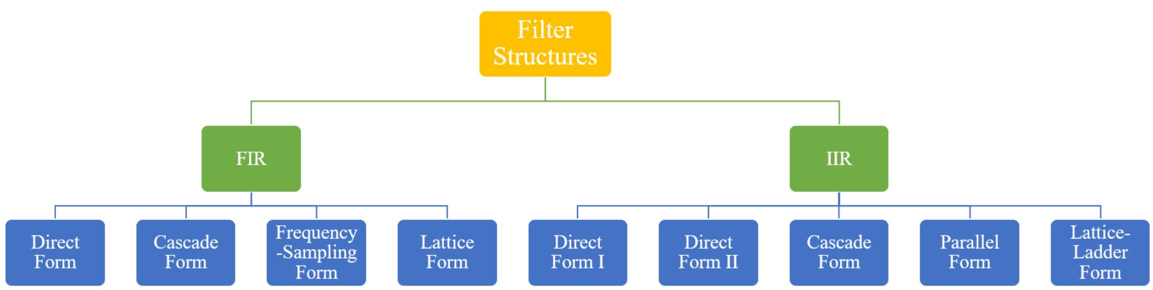
$$H(z) = \frac{b_0\ + b_1\, z^{-1} + \cdots + b_M z^{-M}}{a_1\, z^{-1} + \cdots + a_N z^{-N}}$$

So, the output in IIR filters will depend on past and present values of input, but also on the past outputs. And it will use feedback. So, because of feedback, it may not always be stable; inherently, it may not be stable. That is, there is a numerator and denominator term for this, and for any value for which this may become 0, those values are typically called poles; for such values, this H(Z) will go to infinity, and hence, the stability for the poles is a factor, like whether it is within the unit circle or not, that will typically define the stability of such filters. So inherently you cannot claim that IIR filters are stable filters.

And in the time domain difference equation, you can see that apart from this part, which is the FIR part, there is a recursive part. And that is the beauty that, of course, just with h[n] you cannot represent the IIR filter because there are infinite terms; you do not know how many terms there are. But you are basically representing them in a numerator-denominator form, or in other words, you know the expansion of binary negative expansion. So, if I say this part, this is a numerator and denominator. This is equivalent to the numerator, and if I say this is the denominator inverse, this power of minus 1, this is this polynomial in Z, so this will have infinitely many terms, the expansion of this negative index, and that will make this thing infinite, and that is the reason this h[n] is an infinite-valued impulse response.

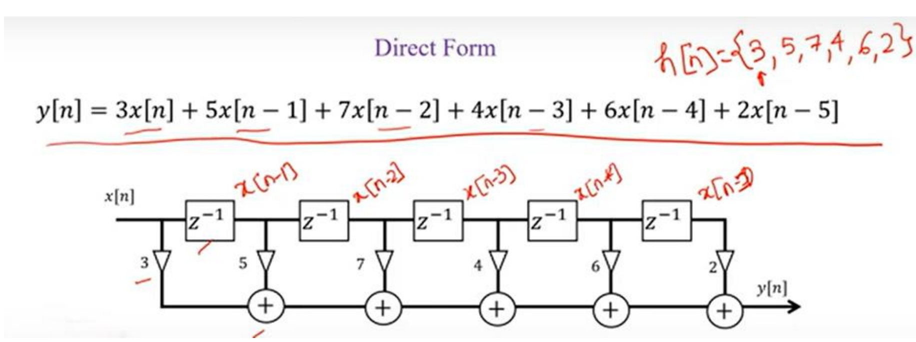
$$H(S) = \frac{N(S)}{D(S)} = H(Z) \cdot H(Z)^{-1}$$

So, there are several filter structures. So, for FIR and IIR, we will restrict ourselves to the structures of FIR filters, not the design. And typically, there are four important structures.



First is a direct form of structure. The second is a cascade form. The third is the frequency-sampling form. And the fourth is the lattice form. In IIR, we have direct form 1, direct form 2, cascade form, parallel form, and lattice-ladder form. We also have transformed forms which I have not mentioned here. But let us focus on the FIR part.

So, what is the direct form? Now, instead of writing the generic form, let us take an example to understand this clearly.



This is an FIR filter which is represented by this difference equation. Now, how do I know this is an FIR filter? The simple trick is that there is a present value of input and there are past values of input, but there is no value of output or previous value of output. So, this makes it an FIR filter, or you can say I can easily see that the impulse response h[n] is nothing but 3, and this is my 0 index; then I have 5, then I have 7, I have 4, I have 6, and I have 2.

$$h[n] = \{3,5,7,4,6,2\}$$

So clearly, if I see, with each delay I can represent them as x[n-1], x[n-2], x[n-3] , x[n-4], x[n-5].

And see now, this is three times, so these are multiplier blocks, this is a delay block, and this is an adder. So, these are the three components by which we are going to make this filter structure. So, if you see, what is being added here? 3 times x[n], 3 times x[n], and plus 5 times x[n-1]. Now, to this sum, I will add x[n-2] multiplied by 7. So, to this, here I will add 7 times x[n-2].

And to this whole expression, I am adding 4 times x[n-3] and to this whole thing, I am multiplying 6 times x[n-4] and then to this whole thing, I am adding 2 times x[n-5] , right? So, this is being added here. So, this is my y[n] as

$$y[n] = 3x[n] + 5x[n-1] + 7x[n-2] + 4x[n-3] + 6x[n-4] + 2x[n-5]$$

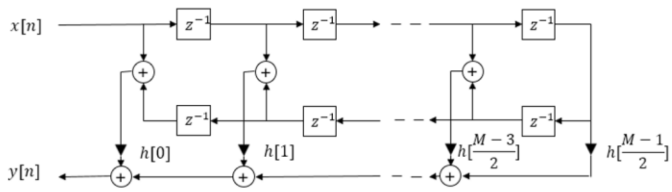
So, you can see this is definitely the structure that is representing this difference equation.

Now here, if it is having the length m, I would not say order m, but I would rather like to say that if the filter length is m, the order is M-1 because the order is usually counted by the number of delays. So, you will need to require memory; memory basically means delays or the past inputs, which are M-1. Then M-1 adders are required, and M multipliers are required. So here is what the value of M is? So here are 6 values. So M is here 6. So hence I need 6 multipliers, 5 adders (6 -1), and 5 delay elements (1, 2, 3, 4, and 5).

Now there is another form of FIR filters that is called linear phase filters.

$$h[n] = h[M-1-n],\ n = 0,1,\ \dots,\ M-1$$

So, what is the likeliness property of linear phase filters? Now, of course, the linear phase filters, by definition, mean that the phase will vary linearly with frequency. But from the structural point of view, or the frequency response, or I would say the impulse response

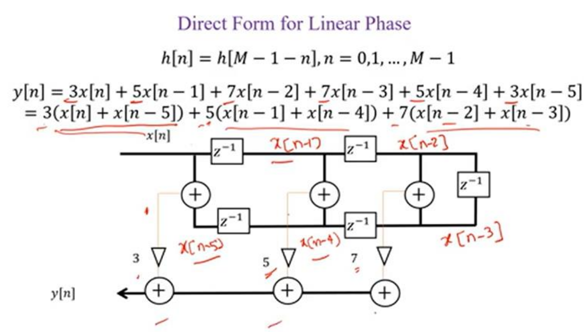


coefficient point of view, this is the relation. The relation h(n) is equal to

$$h[M-1-n],\ n = 0,1,\ \dots,\ M-1$$

or in other words, the coefficients are symmetric. So, what is the advantage here? So, what is the coefficient doing? It is basically used as a multiplier, and hence I can use or club 2 units, that is, $x[0]$ or x[n], and $x[M-n-1]$ with a multiplier $h[0]$ because both have the same coefficient. Similarly, these two x[n-1] and x[n-2-n] will have the same coefficient $h[1]$, and so on and in this manner, I can reduce the number of multipliers by half. If it sounds confusing, let us take an example.

$$\begin{aligned} y[n] &= 3x[n] + 5x[n-1] + 7x[n-2] + 7x[n-3] + 5x[n-4] + 3x[n-5] \\ &= 3(x[n] + x[n-5]) + 5(x[n-1] + x[n-4]) + 7(x[n-2] + x[n-3]) \end{aligned}$$



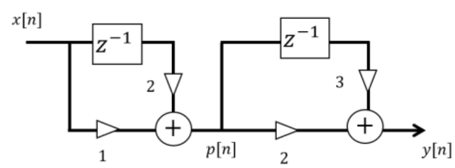
So, this is happening. So, for M even, you have your number of multipliers as $\frac{M}{2} - 1$, and if M is odd, we have $\frac{M+1}{2}$. So, let us take an example. So first, is this a linear phase filter? So, let us see. So yes, the first and the last have the same coefficient, the second and second last have the same coefficient, and the third and third last have the same coefficients. So, I can combine them as 3 into this, plus 5 into this, plus 7 into this, and this is what I have done here.

So, this is x[n], this becomes x[n-1], this becomes x[n-2], and this is x[n-3], and then another delay, so x[n-4], and then this is x[n-5]. So, you see that clearly this is x[n]; this section is $x[n] + x[n-5]$, which is this portion, then multiplied by 3. What is this portion? I have intentionally made a different color for this branch so that it does not look like it has touched this wire, the black wire, and it is basically summing the two things. So here, x[n-1] and x[n-4], that is, these two parts are being summed, and this is

multiplied by 5. And similarly, here (n-2) and (n-3) are summed, and then this is multiplied by 7.
And finally, I am going to add all three. Now, this is a 2-input adder. So, I cannot use a single adder. So, I have to use three adders here, and this is the output that I get.

Another form is the cascade form.

$$H(z) = 2 + 5z^{-1} + 6z^{-2} = (1 + 2z^{-1})(2 + 3z^{-1}) = H_1(Z) \cdot H_2(Z)$$



Now, cascade forms are typically used if your H(Z) can be factorized like this. Now, what is the advantage? You are getting smaller structures that can be represented, perhaps using a single delay or two delays, compared to a bigger structure. Now this is just for illustration, but if you have a very long polynomial in Z inverse, it is easier to do cascades and design smaller filters like this and if there is a problem with a filter, you can easily debug it by checking each component separately. So, you can easily see that this can be factorized into this.
And now, basically, you can say this factor is $H_1(Z)$ and that factor is $H_2(Z)$. And we know that if two filters are multiplied, they are in the Z domain, which means they are cascaded. And you see this is the direct form for $H_1(Z)$, and this is the direct form for $H_2(Z)$. The next structure is the frequency sampling form,

$$\omega_k = \frac{2\pi k}{M}$$

$$H(\omega) = \sum_{n=0}^{M-1} h[n]e^{-j\omega n}$$

$$H[k] = \sum_{n=0}^{M-1} h[n]e^{-j\frac{2\pi kn}{M}}$$

and this is basically developed to design the filter. Now, what is the filter design problem? The filter design problem is given the specifications such as this is my cutoff frequency and this is my passband ripple, stopband ripple; what should be the best values of h[n]? And in this case, you can see that if I am given my frequency response, which is $H(\omega)$, I can say that I can sample it at M number of points, and this is nothing but converting our DTFT to DFT.

But here, if you see, this is the desired response; say I am designing a low-pass filter. So, say this is my actual response of the filter, say for example, so what will I do? Say that this is π, from 0 to π, you can take from 0 to $-\pi$ to π. The main concept is that I will sample this into M points, and this is what I basically get. This is the DFT, M point DFT of my impulse response n[n]. So, how do I get h(n)? I get my h[n] by using the IDFT.

$$h[n] = \frac{1}{M}\sum_{k=0}^{M-1} H[k]e^{j\frac{2\pi kn}{M}}$$

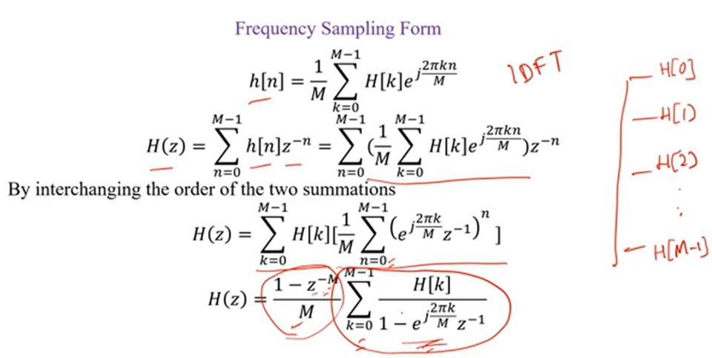
This is the IDFT. Right? And this is basically the design problem that is getting my h[n] from my $H(\omega)$. But for the structure, what I need is H(Z), and H(Z) is defined as

$$H(z) = \sum_{n=0}^{M-1}(\frac{1}{M}\sum_{k=0}^{M-1} H[k]e^{j\frac{2\pi kn}{M}})z^{-n}$$

which is above. Now, what we will do is have two summations; we will interchange the order, and if we interchange the order, what we get is basically this multiplied by this. So, now if I see this is a GP series, this part is a GP series, and if I try to solve it, I get this 1 by m comes here, and this will be,

$$H(z) = \frac{1-z^{-M}}{M}\sum_{k=0}^{M-1}\frac{H[k]}{1-e^{j\frac{2\pi k}{M}}z^{-1}}$$

This is what I got, and one summation I have solved using this, and this remains as a summation. Now, what is the use of this? The use of this is that, so what is it? This is an M-order filter with multiple poles at M. So, this is basically that you can easily draw this direct form structure, and this is cascaded with a parallel bank of M filters, where if you are directly given the responses in the form H[0], H[1], H[2], up to H[M-1], you can directly use this structure. So, this will be a parallel filter bank, and in this parallel filter bank, basically you are using this as a delay; like this is the basic form, and then this is cascaded with this filter that is 1 minus z to the power of minus m by m. So, if you know the desired frequency response, or in other words, if you know the values of h[n], even if you know the h[n], you can easily find h of k, and then you can directly use this to find the efficient structure.



And here, if you see, these are basically responsible based on the desired frequency response. Now, another form is the lattice form. So, what is a lattice form? What is a lattice? So, lattice is used in solid-state physics, which is basically the building block of a bigger structure. It is like the tile says you have to do the floor tiling. So, you have a unit tile that is called a lattice, and using that unit tile, you make up the whole floor.

So, similarly, we will make a small structure called the lattice, and that will be replicated by placing or cascading them in series. Now, to understand the concept, say we have a sequence of FIR filters, and this is the form H(Z),

$$H(z) = A_m(z),\ m = 0,1, \dots M - 1$$

$$A_m(z) = 1 + \sum_{k=1}^{m} \alpha_m[k] z^{-k} \text{ and } A_0(z) = 1$$

which is some polynomial $A_m(z)$ where small m will vary from 0 to M- 1. And what is M-1? It is a polynomial in z inverse that is 1 plus the coefficients $\alpha_m[k]z^{-k}$. For simplicity, $A_0[z]$ is 1. So, the unit sample response of the mth filter, H_m(0), is defined as 1, and other values like $H_m(k)$ are equal to $\alpha_m(k)$.

So, for mathematical convenience, we have $\alpha_m[0]$ to be 1. So, what do I get now? Now, basically, we have y(n) = x(n), and this coefficient is considered to be 1 for simplicity.

$$y[n] = x[n] + \sum_{k=1}^{m} \alpha_m[k] x[n-k]$$

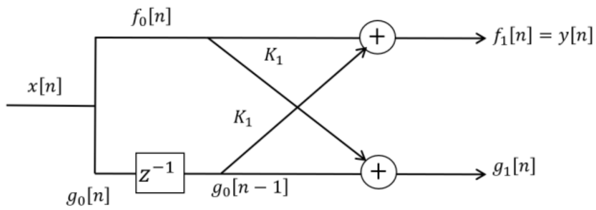
I would say that h(0) in this case is designed in such a manner that its value should be 1. So, the remaining terms can be written in this form; let us take a very simple case study.

We have m equal to 2. So, this can be represented as,

$$y[n] = x[n] + \alpha_2[1]x[n-1] + \alpha_2[2]x[n-2]$$

Now, how do we proceed? So, this is what we got on the previous slide. So, this is what I call our lattice. And this structure is that I am splitting the input x[n] into two signals.

First is called f_0 and the second is g_0. So g_0 I will delay by 1, and then each of these I will cross to get two values, which are kind of recurrence functions. So, I will call this f_1, and this is g1, and this will again repeat in the second stage. So, from this diagram, what I find is $f_1(n)$ is the sum of $f_0(n)$ and k_1 times my delayed version of $g_0(n)$, k_1 times g_0(n minus 1). What is f_0? f_0 is x[n], and what is g_0? g_0 is k_1 x[n- 1]. And similarly, how do I get g_1? g_1 is k_1 times f_0 + g_0 [n-1], and this f_0 is basically our x[n], while g_0 [n-1] is x[n-1].

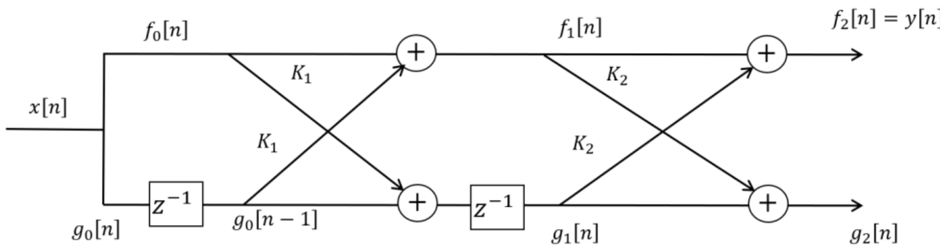


$$f_1[n] = f_0[n] + K_1 g_0[n-1] = x[n] + K_1 x[n-1]$$

$$g_1[n] = K_1 f_0[n] + g_0[n-1] = K_1 x[n] + x[n-1]$$

So, this is clear from this diagram, okay. But this is just a single stage because we have only one delay. We need a second delay. So, how do we do that? We added a similar lattice. See, this is very similar in structure.

And these k_s are often called reflection coefficients. So, in the second stage, we will have k_2. So, K_2 is crossed again. And how do we get this f_2[n] and g_2[n]?



$$f_2[n] = f_1[n] + K_2 g_1[n-1] = x[n] + K_1 x[n-1] + K_2(K_1 x[n-1] + x[n-2])$$
$$g_2[n] = K_2 f_1[n] + g_1[n-1] = K_2(x[n] + K_1 x[n-1]) + K_1 x[n-1] + x[n-2]$$

So f_2[n] is $f_1[n] + K_2 g_1[n-1]$.

$$\alpha_2[1] = K_1(1 + K_2) \text{ and } \alpha_2[2] = K_2$$

And what is $\alpha_2[2]$ to a coefficient of x[n-2]? This is our K_2. So, therefore, $\alpha_2[1] = K_1(1 + K_2)$. So let us take an example; say,

$$y[n] = x[n] + 15x[n-1] + 4x[n-2]$$

if this is our equation in the time domain or difference equation, we know that K_2 is $\alpha_2[2]$ and K_2 is 4. This is K_2; sorry, this is not K_2, but this is $\alpha_2[2]$. But so k_2 will be

$\alpha_2[2]$, which is basically 4, and the second equation says that $K_1(1 + K_2)$. is our $\alpha_2[1]$, and we know what $\alpha_2[1]$ is, which is this; this is $\alpha_2[1]$and this is $\alpha_2[2]$.

$$K_2 = \alpha_2[2] = 4$$
$$K_1(1 + K_2) = \alpha_2[1] = 15$$
$$\Rightarrow K_1(1 + 4) = 15$$
$$\Rightarrow K_1 = 3$$

So, this is what we are getting, which is 15. Now plugging in for k_2, which we got as 4, k1 plus 4 is 15. Hence, k_1 equals 5, so 15 divided by 5 is 3. So, we will just draw the structure, and we know what the value of k_1 is and what the value of k_2 is.

And this is our lattice structure. So, thank you so much. Have a nice day.