Real – Time Digital Signal Processing
Prof. Rathna G N
Department of Electrical Engineering
Indian Institute of Science - Bengaluru

Lecture – 18
IIR Filters 4

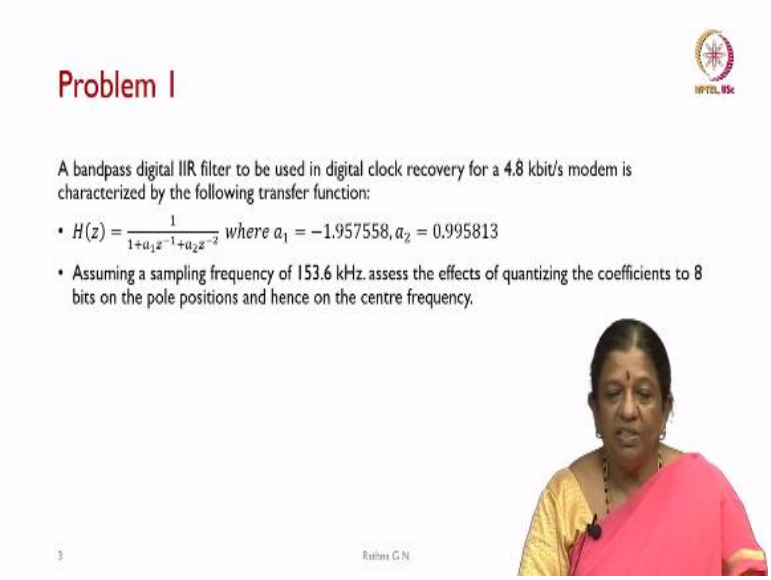
Back to real time digital signal processing course. So, you can see that we are going to discuss the fourth part of IIR filters today. So, why FIR has not taken so much of time why IIR is taking you will be seeing in a while although we have already seen that quantization how it is going to effect. So, in today's class we will work out and then see how the center frequency is going to move from with the coefficient quantization from one value to the other one.

(Refer Slide Time: 00:52)

So, as a recap, so we have been seeing the theory of IIR filters and how to design them and then how the cascade filter section is going to aid us to implement in DSP processor just to comment on it so, we know that cascade section is nothing but it is a multiplication. So, in DSP processor we have seen in the number system that when we do multiplication of 2 numbers, the overflow is not going to happen, only if we have to do the addition we had to take care of overflow and then underflow. So, the parallel section is equivalent to addition.

So, that is the reason why we will not use parallel structure in case of IIR filter design for these hardware units. So, usually we go with the cascade realization, although both cascade and then parallel section have the same effect on the design.

(Refer Slide Time: 02:00)



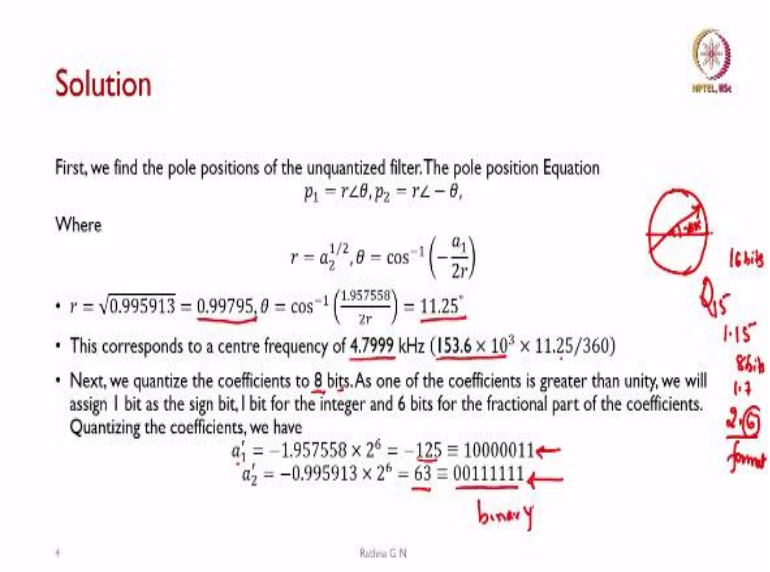
Coming to the thing, so, we will see that, how we will be seeing what happens, what is our transfer function is going to look like? So, we are going to design a bandpass IIR filter to be used in our digital clock recovery for a 4.8 kilobits per second modern what we are using it which is characterized by the following transfer function. So, you have been given your impulse response $H(z) = \frac{1}{1+a_1z^{-1}+a_2z^{-2}}$. So, it should be triggering in your mind that this is just a second order design what we are doing it.

So in this case, you have been given the values of a_1 and then a_2, a_1 is given as -1.957558 and a_2 is given as 0.995813. So, this is your a_1 and a_2 can be designed from using MATLAB or this value has been computed and then you have been given in this equation. So, assuming in this case, because we are using the clock recovery sampling frequency of what we are telling is 153.6 kilohertz to assess the effects of quantizing the coefficients what we are going to do to 8 bits that also you have to keep it in mind.

So, if you want to increase it to 16 bits, you can do it and then verify what will be the center frequency which is going to remain that is what one of the assignment what I will be putting it for

you. So, then we had to say that how the pole positions is going to affect our this thing center frequency. So, we will do that.

(Refer Slide Time: 03:56)



So, as we know that we have the equation how to calculate our p_1 and then p_2 values. So, p_1 is my this thing pole position of the first one, which is given r at an angle of theta and then p_2 is given r at an angle of minus theta because we are designing the complex conjugate poles in this case. So, we had the equation that $r = a_2^{1/2}$ and then $\theta = \cos^{-1}\left(-\frac{a_1}{2r}\right)$. So in this case, how do we compute r?

So, we have been given the value of a_2 as you can see, a_2 is given as 0.995813 which we are going to take it as square root of it, because that will give me a_2 value. So, we are going to get 0.99795 what I will be getting it here so that is r squared is a_2, so r will be $\sqrt{a_2}$ what you are going to have, so, $\theta = \cos^{-1}\left(-\frac{a_1}{2r}\right)$. So, we know that a 1 is negative. So, that is the reason why you will be seeing positive here this is $\left(\frac{1.957558}{2r}\right)$.

$2r$ is going to be whatever what you have computed here, so we will be putting into that, which gives me in terms of degrees as 11.25°. So, this corresponds to your center frequency of 4.799 kilohertz. So, what do we mean by that, so, we know that this is my unit circle. So, this is my centre. So, we are calculating r here. So, this is r it is an angle of what we call it as 11.25° you

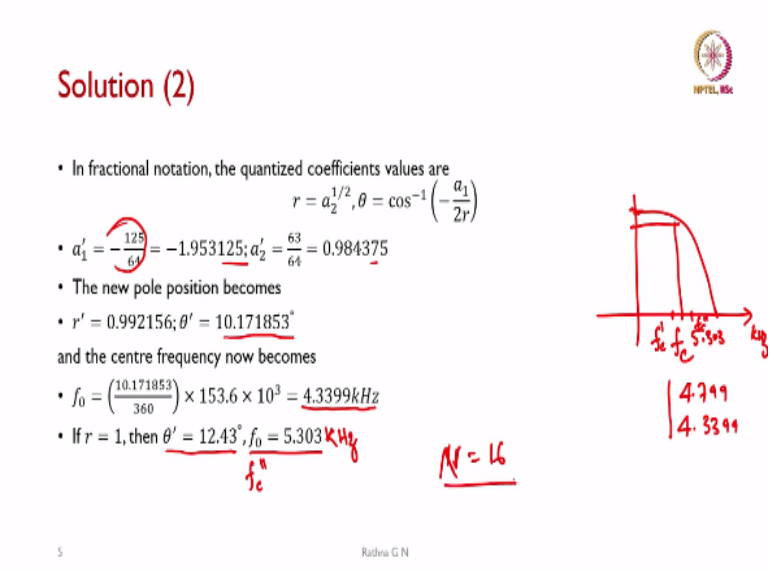
have the pole, this is 11.25°. So, when I calculate center frequency that what we are going to do is this is my sampling frequency what it has been given.

And I know its degrees, $11.25/360$ which is going to give me the center frequency. So, for this, it is at 4.799 kilohertz is the center frequency, it is in the original state what we will call it. Now what you have been given in the problem is we have to quantize the coefficients to 8 bits that is what our constraint as one of the coefficients is greater than unity. So, you will be seeing that $a_1 = -1.957558$. So, we need at least 1 bit to represent my coefficient in the integer format.

So, what will be the representation here? So, in the normal case, we say that it is Q 15 format is 1.15 format for 16 bits number here, you have been given 8 bits, basically. So normally, if we allocate all the 7 bits, 1 is the sign bit and rest of the 7 bits, then I will be talking about 1.7 format, but since my integer value is greater than 1, so I need 1 bit for my integer. So, what happens to this, we will be representing it as 2.6 format, this is the format what I need to represent this value. Then what happens I have 6 bits for my fractional representation, which I had to convert it. So, my coefficient from a 1 it is going to be a 1 dash what I will be putting it -1.957558×2^6, which is equivalent to, I will be rounding of or truncating one of the thing what you can do it so which comes out as -125. So, if you are present in binary, this is the value of what you will be representing it. So, coming to a 2 dash so which is given as -0.995913×2^6 which comes out as 63, so it may come as 63.5 or 63.6 exact value, you can check the thing.

So, why we are representing this also in 63 I will hold on for a while we will come to that this is your binary representation basically.

(Refer Slide Time: 08:51)



So, now we will go back and then recalculate what are the values we have got it. So, my maximum value what is it 125 for a_1 what I have represented in with 8 bits, then we said that it is $2^6$is nothing but 64. So, I can re-divide the value and calculate what I am going to get it? So, it becomes 1.953125 so, what was the original this thing 957558. So, you will be seeing that there is a quantization which has already happened.

Even a_2 you can see that we are doing $63/64$ which comes down to 0.984375. So, what was the original one was 0.995913. So, now with this modified, calculate your center frequency, apply the same equations as the previous one we call that as r' and θ' what we will be calculating. So, you are seeing that from 11.54° it has come down to 10.17. If you want to round it off to 2 digits it is going to be that degree.

So, now calculate is your f_0 that is the center frequency, this is what you will be getting it $\left(\frac{10.171853}{360}\right) \times 153.6 \times 10^3$ because it is in kilohertz. So, you can see that this is going to be 4.3399 kilohertz. So, what was our original thing, it was 4.799 kilohertz, so, you will be seeing that your center frequency has moved, what we call it as center frequency in this case is where I am dropping down this is we usually call it as my cutoff frequency f_c.

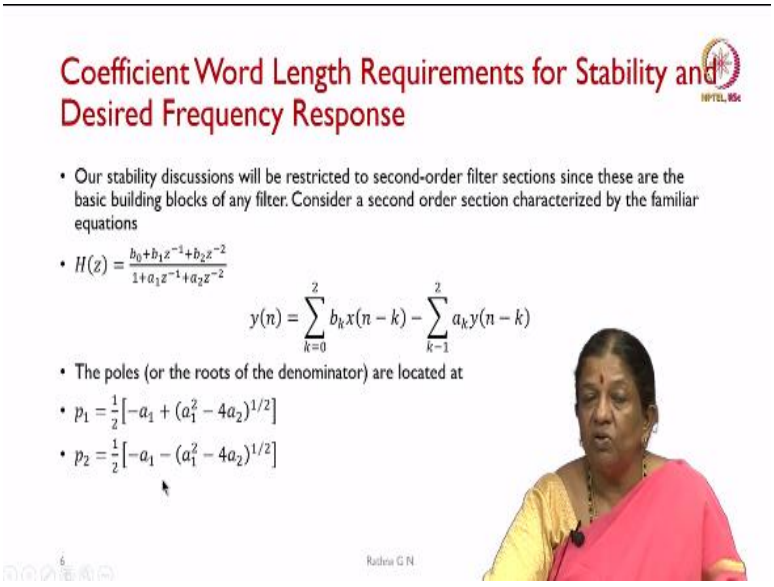
So, you will be seeing that your thing from 4.799 it has got move to 4.3399 so, you will be seeing that your thing has moved to left. So, you are allowing more whatever we call it as in the stopband

region, so, that if there are going to be some aliasing, so, it may creep into the thing. So, this is the effect of moving quantization basically. So, now, I said I can take $r = 1$ what happens in that case, so, we recalculate this. So, we have calculated and only we are giving the final thing, so, you will be seeing that what happens to your theta dash.

It becomes 12.43° and $f_0 = 5.303$ kilohertz actually in this case. So, what happens this is f_c this is I call it as f_c' and this I can call it as f_c''. So, this will be coming to 5.303 kilohertz here if it is represented in kilohertz. So, my f c double dash, so, you will be seeing that more frequency will be coming into your input and then you will have a problem here you are going to reduce the thing if there are any frequency component present in this thing is going to be cut off.

Whereas, here more frequency has come in so, you will be having the aliasing effect. So, that is the reason why we choose $r = 63$ in this case, so, which is almost nearer to our 4.799 compared to going beyond the frequency. So, this shows that how your number of bits is going to effect. So, now you can calculate $n = 16$. So, what is the frequency how much difference you can get it you can calculate and then give the result.

(Refer Slide Time: 13:15)

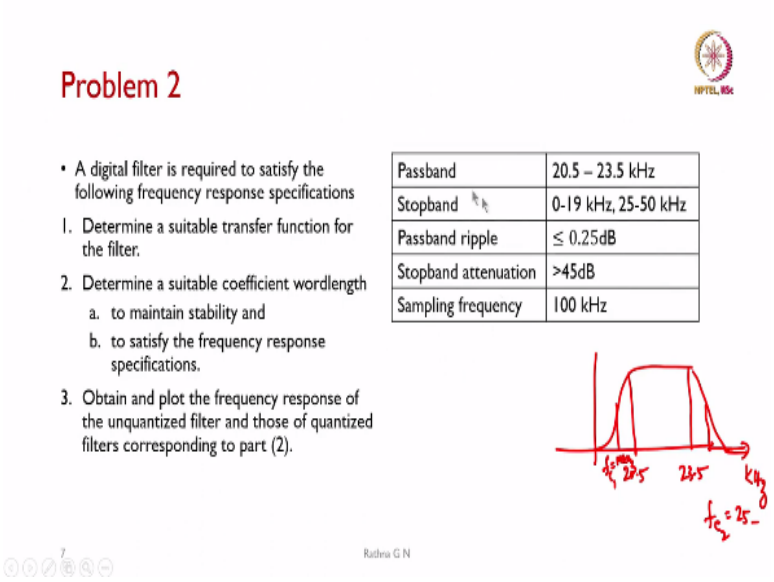


Continuing with the thing, how the next one is what is the word length requirement for stability and desired frequency response what we have to say. So, our stability discussions will be restricted to second order filter sections, because each individually if they are stable then we say the complete

IIR filter is stable, since these are the basic building blocks of any filter and consider our second order section characterized by the familiar equation what you are seeing it here.

That is a $H(z) = \frac{b_0+b_1z^{-1}+b_2z^{-2}}{1+a_1z^{-1}+a_2z^{-2}}$ so, a_1 and a_2 our pole position. So, in the equation what you will be getting is $y(n) = \sum_{k=0}^{2} b_k x(n-k)$ this is our 0 representation minus our pole position this is the feed forward section and this is the feed backward section. So, we know that poles are the roots of the denominator are located at what we call it $p_1 = \frac{1}{2}\left[-a_1 + (a_1^2 - 4a_2)^{1/2}\right]$ and $p_2 = \frac{1}{2}\left[-a_1 - (a_1^2 - 4a_2)^{1/2}\right]$. So, of p_1, what we will be representing it.

(Refer Slide Time: 14:51)



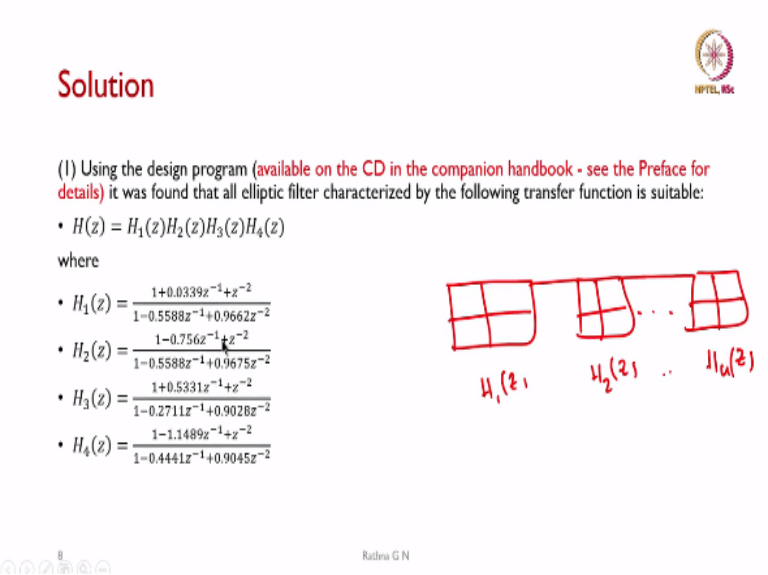
So, what happens in this case, a digital will take up an example, a digital filter required to that is satisfy the following frequency response specifications determine a suitable transfer function for the filter and then determine a suitable coefficient word length to maintain stability and satisfy the frequency response specification. So, you will be taking obtain and plot the frequency response of the unquantized filter and those of quantized filters corresponding to this part basically how you will be number of bits what you will be considering.

So, what is the specification what we have? Passband region is given by 20.5 to 23.5 kilohertz and then stopband is given. So, you will be seeing this is a bandpass filter basically, so, how it is represented, this is the way I represent it. And then this is given as 20.5 to 23.5 is my passband

region, this is in kilohertz I will put it. So, then what are the stopband so, we will call it as this as $f_{c1} = 19$ kilohertz sorry, it is becoming little small and then here $f_{c2} = 25$ to….

Because it is sampled at 100 kilohertz as you can see, till $\pi/2$ it is going to be your stopband region in this case 25 to 50 kilohertz and then the ripple what you want is less than or equal to 0.25 dB and then stopband attenuation what I want is greater than 45 dB. So, as you know that it is better to design this filter using MATLAB and then get the values of your coefficients basically.

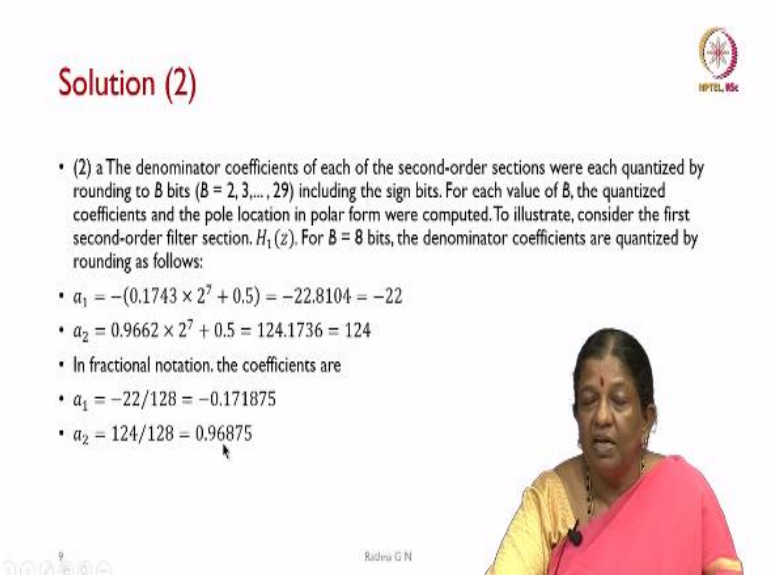
(Refer Slide Time: 16:57)



So, when you do that, this is from the book I have taken the example. So, you can refer to the book and then see CD of the book is going to give you this example, this is basically from Ifeachor what I have taken the thing, digital signal processing book. So, in this case what happens, it generates as in the lab we will be seeing it that how many sections it is going to create. So, here there are 4 second order sections which has been created and then using their zeros and then poles, you will be representing your impulse response like this H_1, H_2, H_3 and H_4.

So, you will be seeing that this is in a cascade form, basically, there will be 4 sections what I am going to have, this is my $H_1(z)$, $H_2(z)$ and then this is $H_4(z)$. So, these are the values what you will get it. So, you have to take each second order section, and then see whether there is going to be over flow or underflow or what is the number of bits what you needed, so that you are going to have the stable filter.

(Refer Slide Time: 18:27)

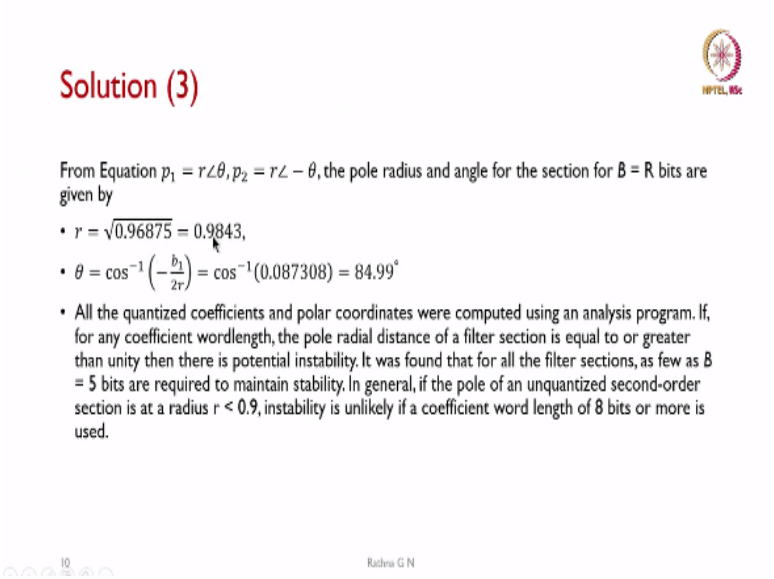


As you can see in this case, so you will be varying your number of bits, that is 2 to 2, 3 like that you can keep on changing and then you can go up to 29 bits. So that is what the theory gives basically. So, we will work out for $H_1(z)$. So, what is the number of B bits we will assume, because even in the previous example, we had taken the 8 bits, we will see whether it is going to give us a stable filter. So, you will be seeing that a_1 will be because all the coefficients are less than 1.

Now what I can represent this as a 1.7 format, that is why you will be seeing multiplication by 2^7 is happening and then we are doing the rounding that is the reason why we are adding 0.5 to that. So, the value is going to be -22.8104. After that we will be truncating the value. So, which is going to give us -22 same things will calculate a_2. So, it will be giving us 124.1736 so, you we will be truncating here, it is going to give us 124.

So, the fractional notation the coefficients are nothing but $-22/128$. So, this is in decimal value. So, when I represent it in fraction, so it will be giving out as -0.171875. So, you have seen I think something triggering in your mind. So, the coefficient quantization has already happened with 8 bits, 0.174 to 0.171 what it has got reduced in this case. So, a_2 will be becoming 0.96875 so, original was 0.9662. So, you have seen that it has got because we have taken around off it has got a little bit increased compared to the original one.

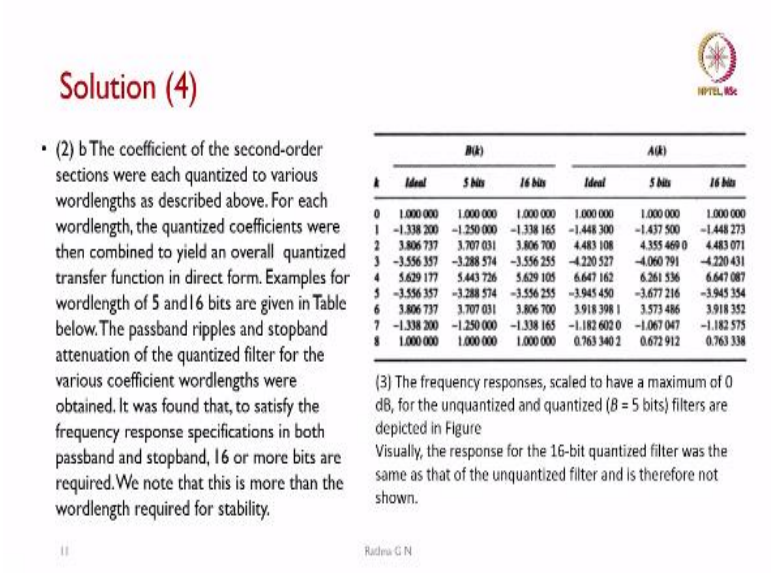
(Refer Slide Time: 20:37)



So, with this you can go back and then calculate your r and theta value and see that what we are going to get is 84.99° what we are getting it so, what it says is the all quantized coefficients and polar coordinates were computed using an analysis program and if for any coefficient word length the pole radial distance of a filter section is equal to or greater than unity, then there is potential instability that is what the literature gives. And then it was found that all the filter sections as few as $B = 5$ bits are required to maintain our stability.

So, in general if the pole of an unquantized second order section is at radius less than r, $r < 0.9$. So, what it says is instability is unlikely. So, if it is ≥ 0.9 you may say that the system may become unstable then you have to take care of designing the proper stable filter. So, as with the word length 8 bits we see that it is 0.96875 what we have is 0.9843. So, is more used in this case.

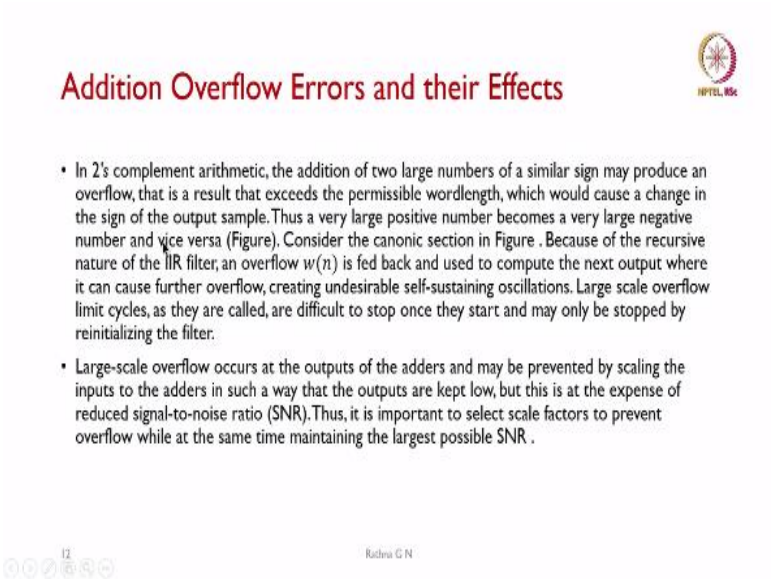
(Refer Slide Time: 22:15)



So, to see that b coefficient of second order section were each quantized to various word lengths, so, you have to use your MATLAB code to do these things. And then you can vary your word length 5 to 16 bits, and then how they are going to be represented is given in the table here, this is my poles that is A coefficients and this represent zeros that is a B coefficient for 5 bits, what we want is ideally is this one and then with 16 bits, how almost closer what we are as you can see that most of the 16 bit is closer to the original one what we say it.

So, this is how we will be calculating and then fixing our number of bits that is the, something should be triggering in your mind all DSP processor, most of them are 16 bit defined.

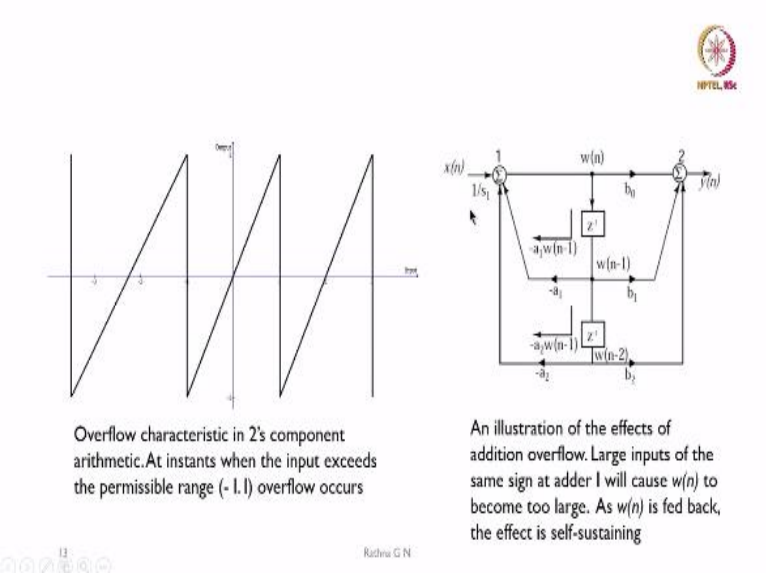
(Refer Slide Time: 23:18)



And then we have to this is one of the distinct quantization of our coefficients how it is going to vary the thing. Next is, we have to take care of overflow errors and their effects based on it what we have to decide on it. So, we know that in 2's complement arithmetic, the addition of 2 large numbers of a similar sign may produce an overflow, if it is beyond our representation, we are going to have an overflow so that it exceeds the permissible word length.

And then very large negative number if you are adding them negative numbers you may have an underflow in that. So, these are the 2 things what you have to look at it what is that?

(Refer Slide Time: 24:08)



Overflow characteristic in 2's component arithmetic. At instants when the input exceeds the permissible range (- 1, 1) overflow occurs

An illustration of the effects of addition overflow. Large inputs of the same sign at adder 1 will cause $w(n)$ to become too large. As $w(n)$ is fed back, the effect is self-sustaining

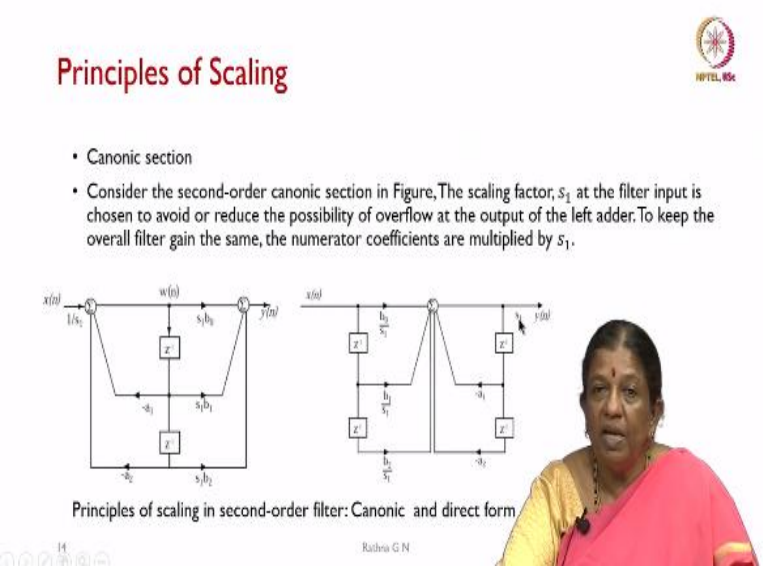
We will see in the figure, what is the thing happened? So, the value is getting added here, from here to here, it has gone to the positive value we will assume and then what happens. So, when my number of bits are not sufficient immediately drops down to the negative value. And then again it starts building up once it reached the peak value it is going to drop down. So, this is how it will be oscillating between -1 and then 1.

If this overflow or underflow is not taken care off. So, what it says is large scale overflow occurs at the outputs of the adders and may be prevented by scaling the inputs to the adders in such a way that outputs are kept low, but this is at the expense of reduced signal to noise ratio, because you are bringing down your amplitude of the signal, then my signal to noise ratio is going to

have a effect on it. So, that is how it is important to select scale factors to prevent overflow while at the same time maintaining my largest possible signal to noise ratio.

So, coming to the thing you will be seeing that overflow illustration is shown in this figure. So, you have b_0, b_1, b_2 are the forward coefficients and then a_1 and a_2 are the feedback coefficients. So, those both $-a_1$ and $-a_2$ and what is the scaling factor we will be providing it one of the section if I am correct, taking it up, it is going to be scaled by 1 by s_1 at the input there are different places where you can do the scaling whether at the input or at the output or if you are providing any of these sections, you have to integrate into all the arms basically, to take care of that you have scaled everything fine. So, as you we will be seeing using the MATLAB when you are doing it the scale factor is given in the beginning itself. So, that has to be used and then your filter has to be designed.

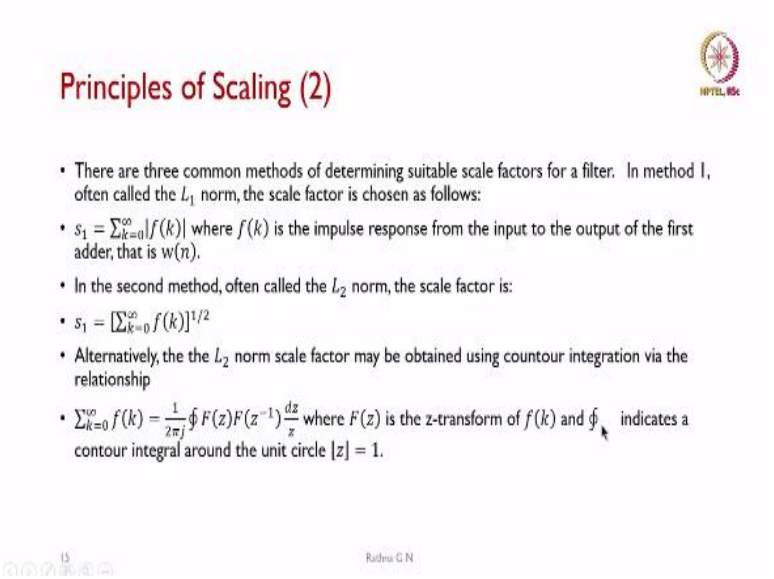
(Refer Slide Time: 26:29)



So, what is the principle of scaling? So, first we will consider the canonic section because this is the one most widely used in all hardware implementation that is what we are seeing it as I said that it is scaled by 1 by s_1. And when we want to get back of our $y(n)$ either I can scale up here or provide at this legs. So, if I take the scaling factor inside, as I was mentioning in the original thing, this is going to be $\frac{b_0}{s_1}$ and $\frac{b_1}{s_1}$ and then $\frac{b_2}{s_1}$ when we are loading our coefficients itself, if we want we can scale them if their power of 2, if they are not then you will be having a little computation

involvement in designing your IIR filter. So, you will be seeing that in the end what you can do is I can scale the output by s_1 value.

(Refer Slide Time: 27:45)



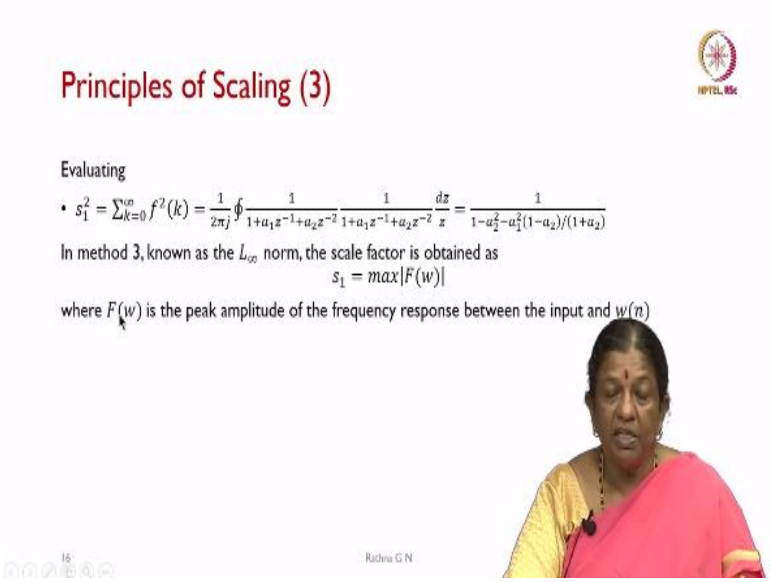
So, you will be seeing that what are the principles to use the scaling? So, most of you would have heard of what we have norms basically, it can be L_1 norm, L_2 norm or L_∞ are the 3 norms what we have in literature. So, what is that first we will see the L_1 norm we call it a scaling by as s_1. So, here it says $k = 0$ to infinity. So, I am going to take the frequency response of my input and take the magnitude of it and then calculate the summation of all of them.

So, as it says $f(k)$ is the impulse response from input to the output of the first adder, that is w of n in our figure here, this is what I will be taking the impulse response of that and then in the second method, often we usually call it as L_2 norm the scale factor is calculated this way. So, if you want you can call it as s 2 here or we can have it as s_1 is the scaling factor what notation what we are using it. So, in this case, you will take the impulse response, but you will be taking the square root of the value what you have some value what you have calculated.

So that is what, what it says scale factor may be obtained using contour integration via the relationship. The last one L_∞ norm what you are going to do is $k = 0$ to infinity, what I have to calculate f of k impulse response, so which is given as $1/2\pi j$, this is the L_2 norm what we are doing with the contour integration as you can see it integral of $F(z)F(z^{-1})$ this is a complex

conjugate what I have taken the thing $\frac{dz}{z}$, where we are $F(z)$ is the z transform of $f(k)$ impulse response. And this is a represents our contour integral on around the unit circle $|z| = 1$ basically.

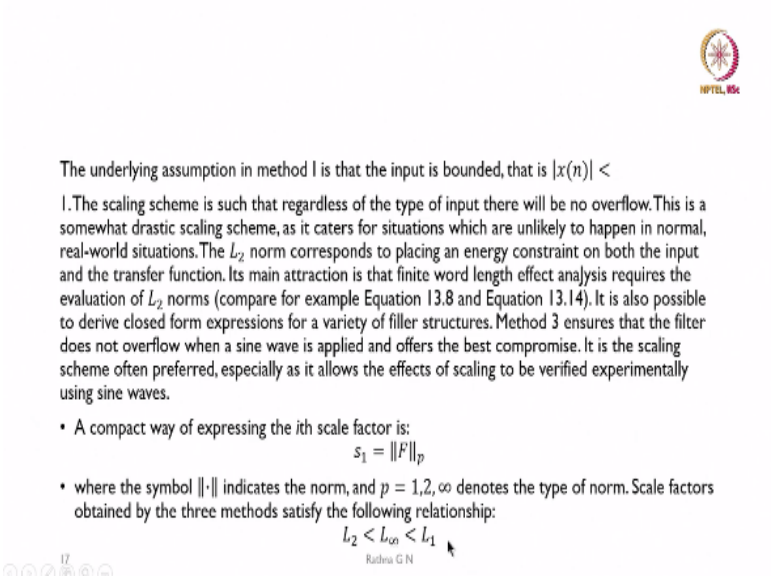
(Refer Slide Time: 30:08)



So, evaluating this is much easier. So, if you want you can go to the book and then refer to the steps involved in deriving this final equation in terms of your poles basically what it is calculated, so, you will be calculating the contour integral of your poles in conjugate form $\frac{dz}{z}$. So, if you do the simplification of it, what then simplest one you will be getting it is $1 - a_2^2 - a_1^2(1 - a_2)/(1 + a_2)$. So, this is how one can manually calculate our L_2 norm using this.

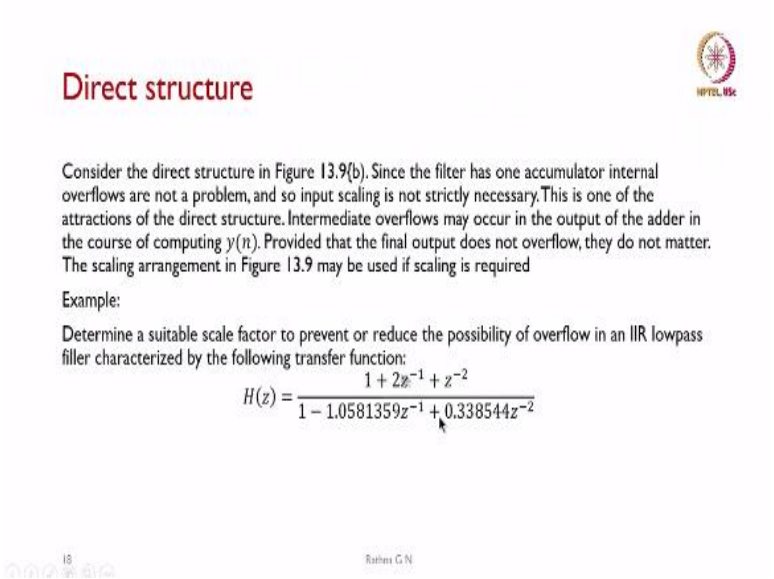
So, in method 3, we calculate peak amplitude of the frequency response between the input and then $w(n)$, that is we will be taking the Fourier transform, and then the peak amplitude whatever it has it, so, we will be giving that as our scaling factor.

(Refer Slide Time: 31:11)



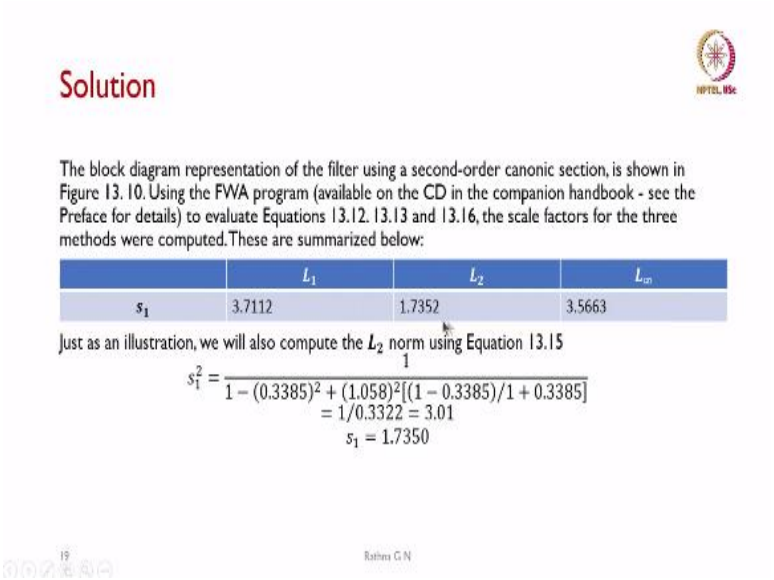
So, these are the underlying methods what we have it. So, you can read the theory compact way of expressing the scale factor what we call it as s_1 is our L_1 norm what we have it with p. So, you will be seeing that this is represented as norm and the method what we are going to have it is L_1 norm, L_2 norm and in L_∞ norm. And then how the scaling factors are going to get themselves aligned or compare what we will be seeing it. So, L_2 norm is the minimum and then next comes the L_∞ and then L_1 is the maximum what you would have scaled. So, the value of L_2 is less than your L_∞ which is less than L_1.

(Refer Slide Time: 32:05)



So, we will take up an example and then see how we are going to calculate this. So, in the book itself where it says that the figure whatever in 13.9 is being used, and then we have this is our second order section for the example.

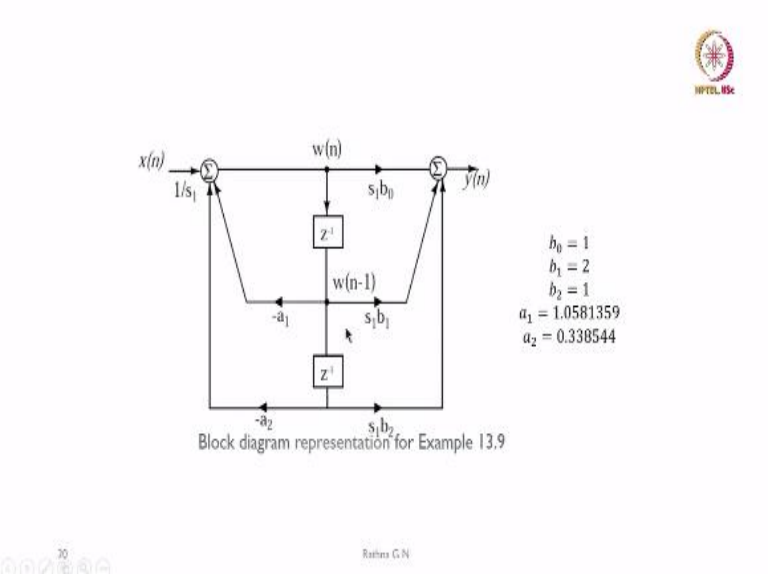
(Refer Slide Time: 32:28)



Then, what is the thing is going to happen. So, either using flow diagram what you can flow analysis, you can use the thing or you can use the book if you are using the thing CD which is a companion which is going to have it so, you will be getting these codes for this program and then you can run it. So, you can evaluate all the equations in your book and then get the scale factors for 3 methods, which it is computed and then you will be seeing it. So, for the L_1 norm it is 3.7112 and for L_2 norm it is 1.7352 and then L_∞ 3.5863 what you will be getting it.

So, one of the thing what we can compute we know that we have the equation to calculate our L_2 norm. So, which is given as $s_1^2 = 1$ by this you are a_1 and then a_2 what you will be substituting and calculate. So, when you calculate it s_1 is coming as 1.7350 so, which is closer to whatever the software has calculated the thing so, this is how you can do that.

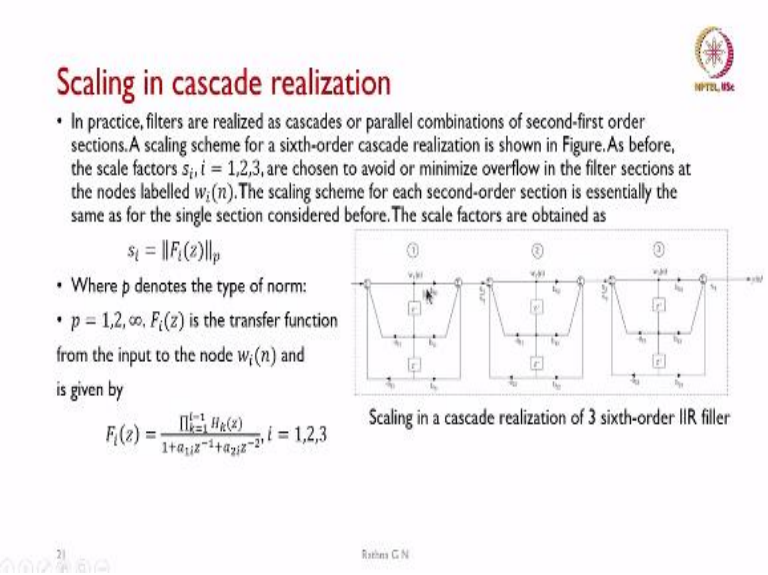
(Refer Slide Time: 33:48)



Block diagram representation for Example 13.9

So, the thing is given us your b_0, b_1, b_2 and then this is the diagram what you have it for the example there and then see go and then look at them.

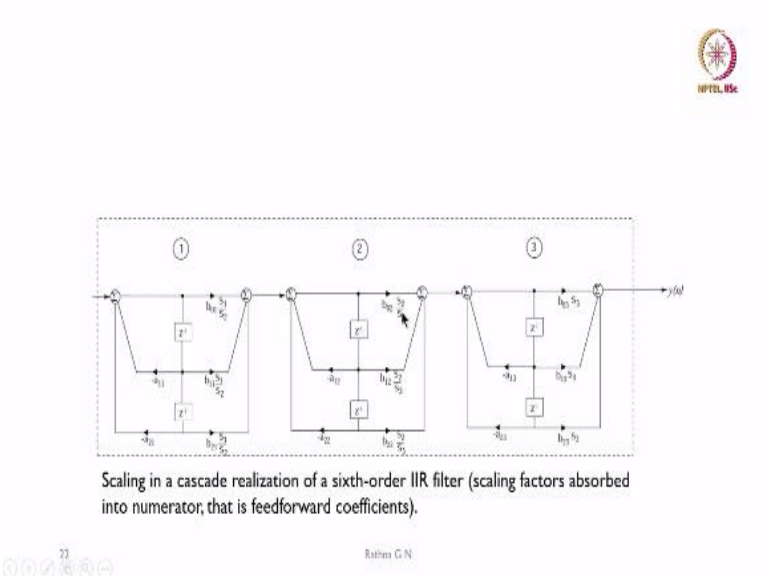
(Refer Slide Time: 34:03)



Scaling in a cascade realization of 3 sixth-order IIR filler

So, if you are realizing the scaling factor for cascade realization, so, you will be seeing that each one has to be calculated and then have you will be at rescaling back in the in between what you can see it one of the norms what you can use that.

(Refer Slide Time: 34:23)



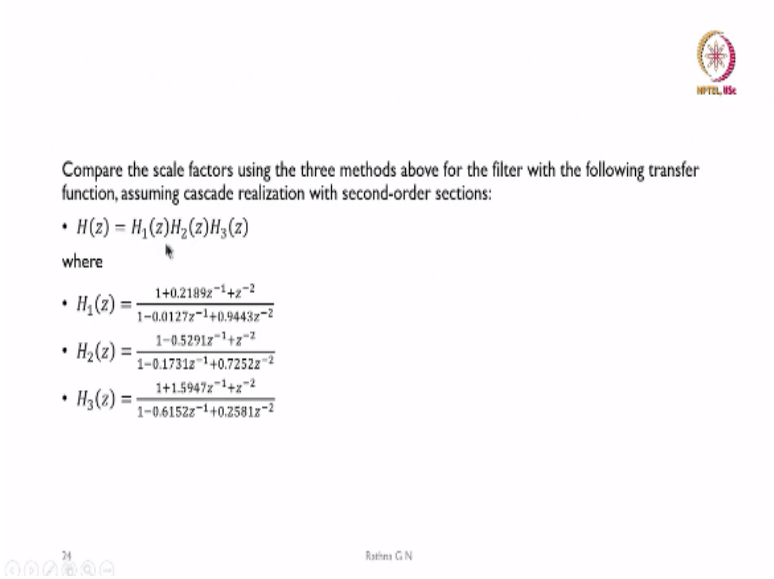
Scaling in a cascade realization of a sixth-order IIR filter (scaling factors absorbed into numerator, that is feedforward coefficients).

So, this is how we will be selecting it. So, one of the way is either you can have the division done, when you are storing your after multiplication before summation you can scale them and then do the addition that way you will be avoiding your overflow instead of adding them and if there is a overflow and then dividing it later. So, you can provide the scaling factor at this itself. So, this shows for the 6th order IIR filter, how you will be doing the scaling part of it and then the final one you will be multiplying only by s_3 whatever scaling you have done. So, you will be accounting for s_3 and then you will be sending the output $y(n)$.

(Refer Slide Time: 35:05)

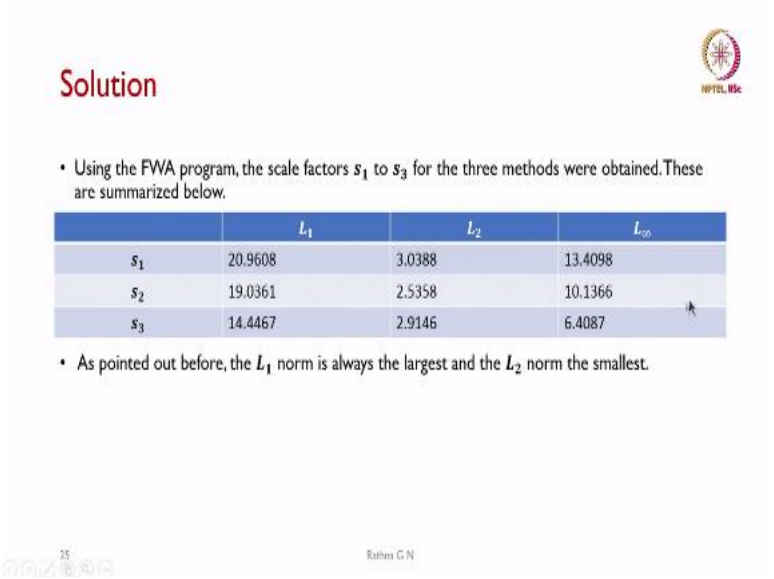
- For the cascade realization, it is common practice to absorb the input scaling factor s_1/s_2 into the numerator of the first stage, s_2/s_3 into that of the second and so on. Thus the scaling factors in Figure 13. 11 can be rearranged as shown in Figure 13 .12 It should be noted that the transfer function of the filter after scaling as discussed above is the same as that of the unscaled filter (theoretically at least)

(Refer Slide Time: 35:11)



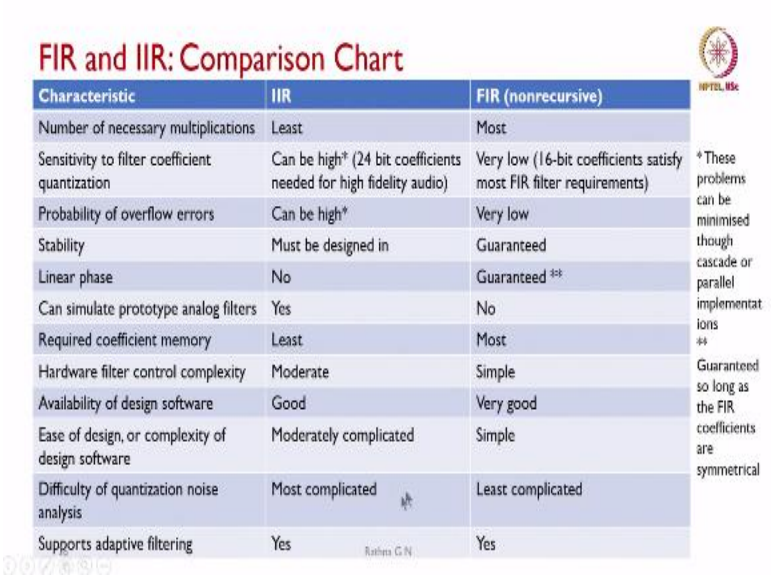
So, this is what the s_1 and s_3 for the figures you can compute. So, these are the cascades section $H_1H_2H_3$ using MATLAB you can calculate them and then you can get the thing.

(Refer Slide Time: 35:22)



So, if you use your FWA program, so, the solution you will be seeing that L_1, L_2, L_∞ for the 3 sections what it is calculated as given in the thing L_2 is going to give you the minimum this thing, scaling factor, and then the maximum is L_1, L_∞ lies in between. The simplest method to calculate is L_2. So, if your application is going to overflow and then you are going to get a results malfunctioning then better to go with one of these norms L_1 or L_∞ which is going to suit your application.

(Refer Slide Time: 36:03)



So, just the last slide to wind up our IIR filter so, we will be seeing the comparison between our FIR and IIR filter, few of the parameters what you can see the thing 1 or 2 I will specify and rest of it you can go through so, that is sensitivity to filter coefficient quantization. So, it can be as high it says in Motorola they use the 24 bit coefficients for high fidelity audio function in their processor. Otherwise, all ti are analog devices, they use 16 bit for their number representation.

So, in the case of we know that FIR filter it is very low, and it says 16 bit coefficients are safely represented or computed using that. And then probability of overflow it can be very high in this case, it is going to be very low. And coming with our linear phase we do not have IIR filter direct method to implement it, but nowadays over the IIR filter, you can try to impose the linear phase so using software techniques, so you can go through the MATLAB functions.

So, otherwise they do not provide the linearity of the phase whereas it is guaranteed and rest of the thing what you will be seeing it one of the examples will be supports adaptive filtering what it says. So, I have both of them support adaptive filter, so we will be considering it in the next after a few classes later.

(Refer Slide Time: 37:48)

So, coming to the end of it which finishes our filter design. So, we will be taking our DFT and then FFT in the next class. So, thank you for listening to this lecture and then happy learning through this media. Thank you.