Real-Time Digital Signal Processing

Prof. Rathna G N
Department of Electrical Engineering
Indian Institute of Science - Bengaluru

Lecture - 30
Overlap Add and Save Method Using MATLB

(Video Starts: 00:23)

Namaskara, So, welcome back to real time digital signal processing lab this time. So, we have seen in theory overlap add and then save method is with an example. So, we will use MATLAB for code to see that how it is going to work in real time, and then we will be seeing on the board also. So, as you can see here, some of the codes able to show you in the last class for DFT and FFT. So, I had not explained you overlap although run some of it.

So, here it is going to run on audio signal. So, you will be a generating what is it timit2 dot asc file that is the speech sampled at 8 kilohertz with 16 bits, what it is going to now it is already stored actually. So, if you want you can record your own speech signal and then load it in this area. So, it can be if you have sampled at 8 kilohertz. So, you can do it or you can change your sampling frequency and then record your own signal.

And use that particular sampling frequency in running this code. So, first we will be running the original speech and then you will be seeing that soundsc thus timit2 what file it has it and then it is getting sample fs frequency and 16 bits what it is used so, you will be playing it first then what is it now? So, your length of speech file as you will be seeing it for one minute or whatever you will be there however long your stored it that you will be using it here.

So, first you have to take the length of this signal and then your $\sum_{n=0}^{N-1} \square$ and then f what we are going to introduce is 1 kilohertz so frequency of sinewave is noise in this so that will be using the filter in overlap add method. So, we will be taking $\omega = 2\pi \frac{f}{f_s}$ and sinewave created with that amplitude is 2000 and then create this with $sin(\omega)$ into number of samples is n that is throughout your speech signal this frequency is going to be present.

So, now our next one is what is it. So, you will be combining with the speech signal sinewave signal what you have generated that is basically corrupt speech by 1 kilohertz tone. So, if you want you can have any other noise, random noise or white noise you can add it and then see that you are going to eliminate using your filter basically. So, we will be displaying this and then you will be hearing the noisy speech signal with this command then you can have a display of it how it is going to look like.

So, as we have discussed in the theory you can calculate the spectrogram using a MATLAB function, here you will be using kaiser window and then number of points what you will be putting it is 200 order and your 256 points and then fs what you will be calculating and then if you are using this is the latest MATLAB what it says you can select if you give a help on spectrogram you will be getting what are the parameters you have to use it.

So, earlier one what it says is older MATLAB we will be using this only number of points and then sampling frequency what you have to give it along with the input signal. So, you can now give title to that noisy signal. And then you can display it the spectrogram and then pause it and then until you press the next key so you can view your spectrogram there. Now you can define the window because sinewave what we have created is 1000 hertz that is 1 kilohertz.

So, I want to eliminate it. So, your filter has to be bandstop filter or a notch filter what you can design use the filter design tool box or you can use here in command window also. So, the stop band region is 1000 so, you have given 900 to 1100 as your transition band and the sampling frequency is fs by 2 which is 4 kilohertz in this or 4000 hertz, then you compute the filter coefficients b. You can use the FIR 1 filter with what is the order of the filter.

You are going to provide 128th order filter what your designing it wn is whatever you have specified here stopband and then your calling it as stopband filter what you want to design. So, then you will be FIR filtering using overlap add method what you will be calling directly fftfilt will be using the overlap add method. So, you will be giving yn, fs and then 16 b is your filter length and then xn is the input and then you will be seeing that when you pass it filtered signal through your soundsc command and you can plot your spectrogram also with respect to this.

So, what we will do is we will put a breakpoint here. So, then run the code. So, the previous one you will be seeing it is overlap with different frequencies what you are seeing it sine and cos function. So, it has come to overlap of 2 spectral lines due to frequency. So, you will be seeing it 60 and 61 what we had done, this is the spectrum of the speech signal as you can see the thing.

So, now, it is going to do the filtering as you can see the thing overlap add technique what it is going to use to there it was a previous selected speech signal now also you have selected the normal speech. Now we will be seeing that it is waiting for any key to be pressed, as previously shown that is you have a sine wave constant your seeing that at 1 kilohertz what you have it.

So, and then these are the speech. So, this is how you have overlapped with the speech signal we have noisy sine wave. So, now, it is going to waiting for one more time to do filtering. So, you. So, you have heard the all though there is a little sine tone is left but most of them have been eliminated. So, now, you can generate your cos and then take this thing power spectrum what you can calculate to find the power spectral density and other things.

So, if we run the complete code, you will be seeing that your power signal what it is going to be the complete signal why how it is it? So, this is one sided power pass spectrum of your y and then you have the signal here. So, once I did pass spectrum of yy what you have it and then signal here what you have recovered. So, you will be seeing that how both of them are look like and then you will be seeing in the output power signal y what dB word length what it has it.

And yy what is it here you will be seeing that your finding use the periodogram function with rectangular window that is what you have used it in this case previous one was the kaiser window and then trying to do that. Here there is no overlap is going to have so then what are the power signal what you will be achieving it is shown in this. So, now well go with the resolution just now we discussed about it. So, what is the thing is going to happen with N = 8.

And then the sampling frequency chosen as 8000 we saw in that theory, so, well look at it how it is going to look like. So, you will be having zeros padded here and then you will be computing sinewave and then you will be taking the FFT of the signal which have been combined here. So, well run the thing. So, you will be seeing that for 8 bit, this is what the resolution is going to be of your coefficients. So, if I increase it to 16, so, you will be seeing the magnitude spectrum how it is going to look like.

So, one can use the sub plot to plot one over the other and then see that how both for 16 point and then 8 point how it is going to look like. So, the next one, what it does is you can have a quantization effect built into this. So, you will be seeing that what is it your sampling frequency is 1 into e power 6 and then your fx is 50 into e power 3 and then your Afs magnitude is 1 and we are telling N is 2 to the power of 11 point FFT what you are going to have.

And then you will be taking the input signal is a cos basically multiplied with magnitude 1 and then this is the normalized position what you can look at it. And plot the signal and then you will be calculating Nfft as N point and then your s what you are going to define is 1 by Nfft into absolute of FFT of y, Nfft and then your frequency you will be seeing the line space one sided power spectrum of y and what you will be getting is you can see the periodogram with the rectangular window what you are going to use for the y signal.

And signal 2 what you will be taking is here you will be defining B = 2 and delta is 2 by 2 to the power of B and then your fs is this and cycles what you will be choosing is 67 and your yy signal whatever you are going to get it is this one. So, then compute your power spectrum from the thing and you can do the quantization how it is going to look like. So, we will run this and see you will be seeing the same thing what we had caught it in the previous one.

This is our yy signal and then y 1 what is the output this is the quantized one is the you can see that with number of quantized points what you have given the thing and this represents your original signal and then error between the original signal and then quantized signal you will be seeing that this is how the error is going to creep in. So, you are seeing to the original one sided power to the quantized one you will be seeing there are little peaks what it is going to appear.

So, how much you can tolerate after quantization to fixed point what you can have a look at it in the case. So, coming to we have discussed about overlap add method already. Usually students what did I give them is to design both overlap add and then save. So, in this case, I will be showing you what they have designed. So, this is a sampling frequency we assume it is 8 kilohertz in the thing and then whatever you have to generate for 40 samples in one period.

So that is 8000 divided by 40 is the signal what you will be generating is 200 hertz signal in this case. So, they will be using as we discussed in the IIR filter. So, to do our FFT calculation we use sine generation using IR in the oscillatory mode. So, to generate 200 hertz, these are the values what it has to be generated. So, your y 1 = 0, y 1 of 2 is given with these values and then for k = 3 to 10,000 so, you will be generating your sine wave y 1 of k with respect to as you can see this is the IIR filter this all is your amplitude.

And then y 1 of k - 1 is your feedback signal and then y 1 of k - 2 is the second feedback signal what you will be providing to oscillate the filter. So, then, what is it here one of the frequency is here it is 200 hertz, so, you have to generate 3 sine waves usually I asked them to do the thing in this case they have taken x 1 is 30 samples, which is going to give us 267 hertz and in the same way, third one is 400 hertz, what it is generated using our IIR filter in the oscillatory mode, then combine all these 3 y 1, y 2 and y 3.

So, you will be getting 3 frequencies which you can plot it. So, take the absolute FFT defined by the function in this case and then plot their magnitude. Now, you will be generating FIR band pass filter of 237 to 297 hertz to get the second frequency of order 350. So, you can see that FIR 1 is a 350 order. So, something should be striking in your mind to eliminate in a short this thing frequency. So, what they have given is that is 237 to 297 is their transition band, but it is being used.

So, this is the high order to eliminate one of the frequency from the 3 frequencies what it has been generated. So, this will be passed through you will be seeing that filter basically what you can do the thing. So, length going to implement direct convolution first what you will be doing at length of y and then you are doing the convolution of it and generating the remaining length minus length plus 1 elements of output what you will be doing again the second convolution.

So, this is how your overlap add method has been done. So, here first one is overlap save method what it is going to be, so, FFT length is defined as 512 and then the length of whatever the filter is what you have assuming it and then size of each signal segment is 162. Then you will be padding with zeros and then your FFT of h is nothing but FFT of your h that is you are taking the FFT of your response of the filter.

And then you will be output array for final output what it is defined here and you will be having for k = 1 what you will do it so, FFT of the first signal what you will do it so, you have zeros and then why 1 to s length and first you are padding with zeros. And then y you have taken the thing and then you will be doing the as you can see it is a convolution basically FFT of h dot into FFT of s that is basically in the DFT domain, you are multiplying 2 DFT signals here both h and then s then you will be outputting the IFFT of this output.

So, this you will be plotting it and you will be doing for first one you are done it for k = 1 and then 2 and then 3 and then afterwards you will be going up to the length by 2. So, this is how to do overlap save method and the last one you will be doing it manually. So, the thing you will be seeing that the first one is overlap save method. So, what we will do is we will put a breakpoint here and then run our overlap save method.

So, you will be seeing that, this is how we have generated 3 this thing frequencies as you can see that it is 240 hertz 297 and then 400 hertz if I am correct, these are the 3 frequencies what it was generated and then you will be seeing the fs was 8000. So, you will be seeing the mirror image of the 3 frequencies on the left side also. So, this is the input to overlap save method the first one and then after it is using a band pass filter as you can see, so, it has eliminated the other 2 frequencies and then your, can you tell me what kind of filter it is.
In the previous case we saw it was a bandstop filter, here it is going to be a bandpass filter. So, you are using only one frequency to be passed. So, if you check the thing value what you will be getting it as 267 is passed the other 2 frequencies have been eliminated and that is using your overlap save method, this was the direct convolution using direct convolution the same thing. So, using the overlap save method also you have got the same thing.

Now it is going to be using overlap add method. So, we have discussed in theory that how it is going to work, this is how this has been implemented as you will be seeing that how the zeros have to be introduced and then you will be adding the previous length and then doing it so you will be seeing somewhere down the line the addition of it plus one, so, all the your x 1 x 2 x 3 till the length of it what you will be adding them.

So, then you will be calculating your inverse FFT and then adding them here out a what you will be seeing it here concatenation is going to happen here and you will output so, you can see that here also you will be seeing that only using the overlap add method, this frequency what is it basically, your 267 like overlap save method, this frequency is left out. So, this shows that how our overlap add and save method to run different ways of it one we saw it with the audio file.

And then we have seen with the frequency generated using IIR filter in the oscillatory mode to eliminate the 2 frequencies only passed only one of the frequency using this method here using MATLAB so we will take a break and then come back for CCS. Thank you.

(Video Ends: 23:14)