EXPERIMENT 4

Overlap Add method for linear convolution

AIM

Write a MATLAB program to perform linear convolution through overlap add method and verify the result through direct convolution using the MATLAB builtin function - `conv`

THEORY

Linear filtering methods based on DFT

Suppose a finite duration sequence $x[n]$ of length L is applied as the input to an FIR filter of length M. The output of the filter in time domain can be expressed as the linear convolution of $x[n]$ & $h[n]$ as,

$$y[n] = \sum_{k=0}^{M-1} h[k]x[n-k]$$

The length of the linear convolution of $x[n]$ & $h[n]$ will be $L+M-1$
We know that the IDFT of the product $X[k]H[k]$ will give us the circular convolution of $x[n]$ & $h[n]$.
We can ensure that this circular convolution has the effect of linear convolution by padding both $x[n]$ & $h[n]$ with enough zeros to make each sequence have a length of $L+M-1$.
Thus we can get the filtered output sequence $y[n]$ using DFT-IDFT method to compute the circular convolution of the zero-padded $x[n]$ & $h[n]$

Filtering of long data sequences

The input sequence $x[n]$ is often very long especially in real-time signal monitoring applications. For linear filtering via the DFT, the signal must be limited in size due to memory requirements. To solve this issues, we use a strategy which involves:

- Segmenting the input signal into fixed-size blocks prior to processing
- Computing the DFT-based linear filtering of each block separately via the FFT
- Fitting the output blocks together in such a way that the overall output is equivalent to linear filtering $x[n]$ directly

The main advantage of this strategy is that samples of the output $y[n]$ will be available in real-time on a block-by-block basis.

Assume that the input sequence is segmented into blocks of length L & M is the length of the FIR filter and $L >> M$.

There are two methods utilizing this strategy:

- Overlap-Add Method
- Overlap-Save Method

Overlap-Add Method:

Here, we segment the long input sequence into fixed size input data blocks of length L.
To each data block, we append $M-1$ zeros to produce the N-length subsequences $x_m[n]$; $m = 1,2,.....$

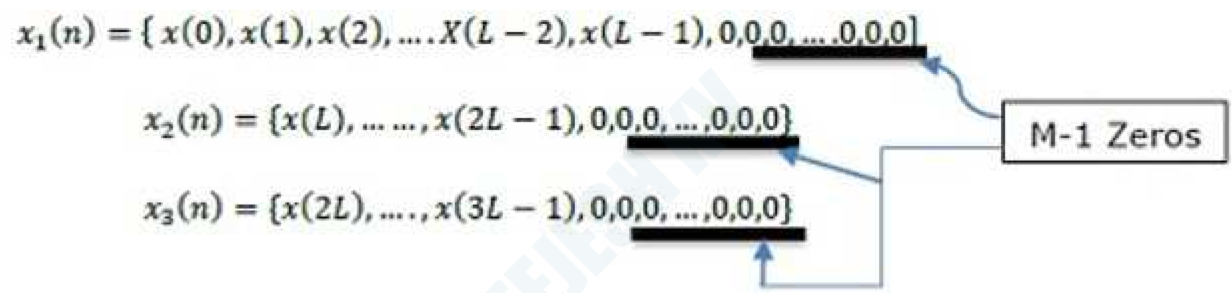


Figure 4.1: Formation of input subsequences: Overlap-Add Method

The lengths of DFTs &IDFTs used in this method are $N = L + M - 1$.
We take each subsequence, $x_m[n]$, and compute its N-point DFT, $X_m[k]$.
The impulse response of the FIR filter is increased in length by appending $L-1$ zeros and an N-point DFT, $H[k]$, is computed once and stored.
For each subsequence $x_m[n]$, we multiply the two N-point DFTs together to form,

$$Y_m[k] = H[k]X_m[k]; \; k = 0,1,\ldots,N-1$$

Taking the N-point IDFT of this result, yields the N-length output data block $y_m[n]$ which is free of aliasing.
The last $M-1$ blocks from each output block must be overlapped and added to the first $M-1$ points of the succeeding block to get the final output sequence $y[n]$ as shown in the figure below.

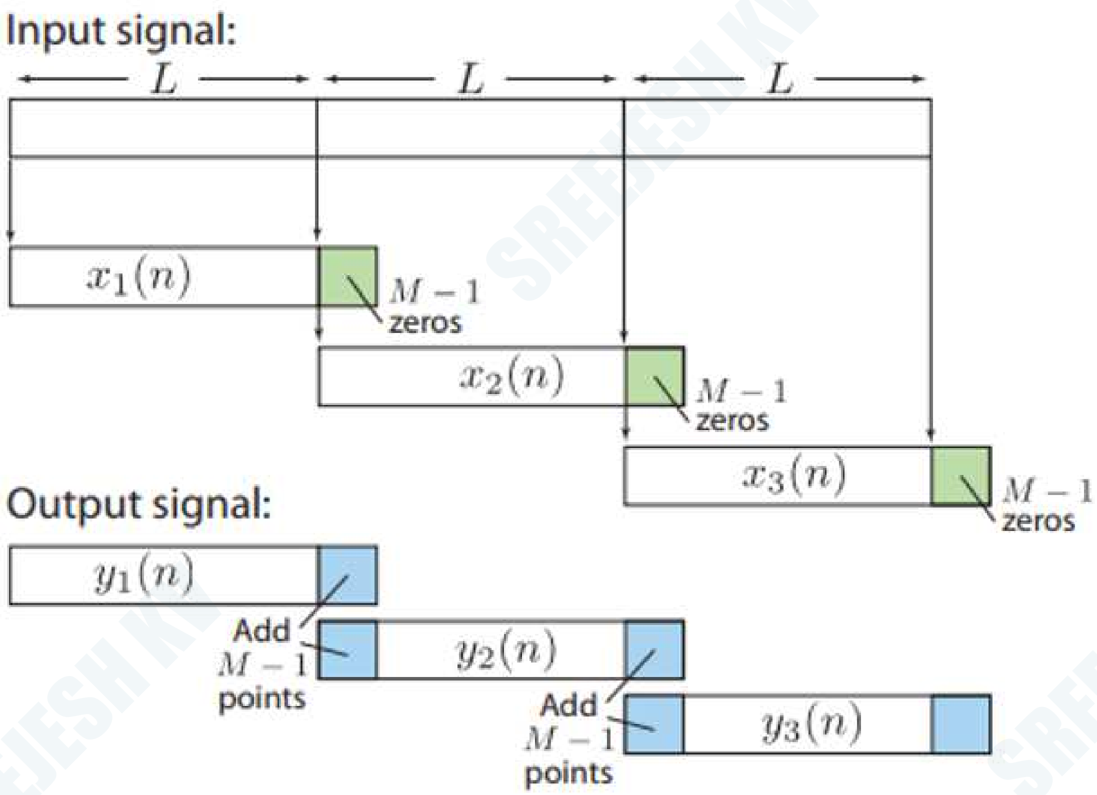


Figure 4.2: Overlap-Add Method

MATLAB FUNCTIONS USED

randi

Pseudorandom integers from a uniform discrete distribution

```
R = randi([IMIN,IMAX],[M,N]) returns an M by N array
containing integer values drawn from the discrete uniform
distribution on IMIN:IMAX.
```

ALGORITHM

Step 1. Start

Step 2. Define the input sequence $x[n]$, its length $N1$, the filter coefficients $h[n]$, its length M, block length L, DFT length $N = L + M - 1$

Step 3. Zero pad $x[n]$ to the length $N2$ which is the next higher multiple of L after $N1$.

Step 4. Zero-pad the sequence $h[n]$ to $M + L - 1$ -length and compute its N point DFT $H[k]$

Step 5. Create non-overlapping subsequences of length L from $x[n]$ and follow each subsequence by $M-1$ zeros.

Step 6. Find the N point DFT of each subsequence, multiply it with $H[k]$ and find the inverse DFT to get $y_m[n]$

Step 7. Obtain the output sequence $y[n]$ by fitting each output subsequences in such a way that last $M-1$ values of an output- subsequence are overlapped and added with the first $M-1$ values from the next output-subsequence.

Step 8. Verify the result obtained using MATLABs inbuilt `conv` function

Step 9. Stop

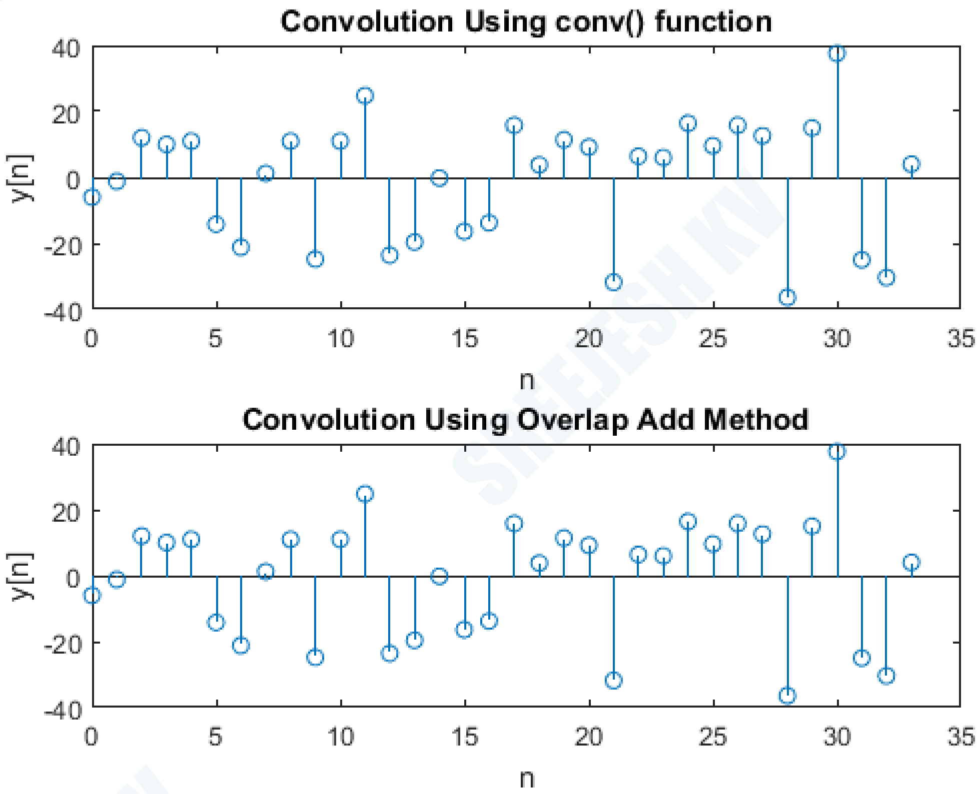
PROGRAM

```
%Title: Program to Perform linear convolution through overlap add ...
    method
%and verify the results using the builtin function - conv

%Author: Sreejesh K V, Dept. of ECE, GCEK
%Date: 25/09/2022

clc;
clear;
close all;

x=randi([-15 15],[1 32]);% Generating a random 32 length sequence ...
    of integers in the range -15 to 15
h=[1 0.2 -2];% FIR filter's impulse response

L=6;%number of nonzero values in each subsequence
N1=length(x);%input sequence length
M=length(h);%filter length
N=L+M-1; %DFT length
lclength=N1+M-1;% length of linear convolution sequence

% --direct linear convolution using inbuilt function for ...
    verification --%
lc=conv(x,h);

% -- Overlap Add method for computing the linear convolution -- %
x=[x zeros(1,mod(-N1,L))];%zero pad x to the length which is the ...
    next multiple of L
N2=length(x);% N2 will be a multiple of L
h=[h zeros(1,L-1)];%zero-padding the sequence h[n] to M+L-1 length
H=fft(h,N);%N=L+M-1 point DFT of h[n]

S=N2/L;%number of segments
index=1:L;%index of first set of L values to be taken from x[n]
```

```
y=zeros(1, M-1);%the output sequence-initialized with M-1 zeros
for stage=1:S
xm=[x(index) zeros(1,M-1)]; % Selecting the subsequence to ...
    process & proceed with M-1 zeros
Xm=fft(xm,N);%N point FFT of subsequence
Ym=Xm.*H;%multiplying subsequence DFT with filter DFT
ym=ifft(Ym,N);%taking IDFT- will give the N point circular convln ...
    of x_m[n]& h[n]

%Z is the M-1 point sequence obtained by Overlapping & adding ...
    first M-1 values of ym to last M-1 values of y
Z=y((length(y)-M+2):length(y))+ym(1:M-1);%

y=[y(1:(stage-1)*L) Z ym(M:M+L-1)];%concatenating the sequences
index=(stage*L)+1:(stage+1)*L;%set the index to next set of L ...
    values from x[n]
end
i=1:lclength;
y=y(i);%trimming the zero values at the end

% -- time values (values of n) for plotting -- %
n=0:lclength-1;%first value of the sequence corresponds to n=0

% -- Plotting the sequences -- %%
figure()
subplot(2,1,1)
stem(n,lc);
title('Convolution Using conv() function')
xlabel('n');
ylabel('y[n]');

subplot(2,1,2)
stem(n,y);
title('Convolution Using Overlap Add Method')
xlabel('n');
ylabel('y[n]');
```

OUTPUT & OBSERVATIONS

Figure Window Output:



RESULTS

A program to compute the linear convolution of two sequences using overlap-add method was written and executed in MATLAB and the result was verified using the inbuilt function `conv`