5

EXPERIMENT

Overlap Save method for linear convolution

AIM

Write a MATLAB program to perform linear convolution through overlap save 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-Save Method:

Here the size of the input data blocks is $N = L + M - 1$ and the sizes of DFTs and IDFTs are also N.

The impulse response of the FIR filter is increased in length by appending $L - 1$ zeros and its N-point DFT, $H[k]$, is computed once and stored.

To begin the processing, the first $M - 1$ points of the first input block are set to zero. Then we will append the first L points from $x[n]$ to it to obtain the N length input subsequence $x_1[n]$.

For subsequent input blocks, to avoid loss of data due to aliasing, the last $M - 1$ points of the previous input data segment are repeated as the first $M - 1$ points of the next data. The remaining L points are new points taken from $x[n]$.

Thus we produce the N-length subsequences $x_m[n]$; $m = 1, 2, \ldots$..

$$
\begin{aligned}
x_1(n) &= \{\underbrace{0, \quad 0, \quad \ldots \quad 0,}_{M-1 \text{ zeros}} \; x(0), x(1), \ldots, x(L-1)\} \\
x_2(n) &= \{\underbrace{x(L-M+1), \ldots x(L-1)}_{\text{last } M-1 \text{ points from } x_1(n)}, x(L), \ldots, x(2L-1)\} \\
x_3(n) &= \{\underbrace{x(2L-M+1), \ldots x(2L-1)}_{\text{last } M-1 \text{ points from } x_2(n)}, x(2L), \ldots, x(3L-1)\}
\end{aligned}
$$

Figure 5.1: Formation of input subsequences: Overlap-Save Method

We take each input subsequence, $x_m[n]$, and compute its N-point DFT, $X_m[k]$.

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 $y_m[n]$.
The first $M-1$ points of each $y_m[n]$ are discarded due to aliasing.
The remaining L points of each $y_m[n]$ are fitted together to obtain the desired result from linear convolution, $y[n]$.

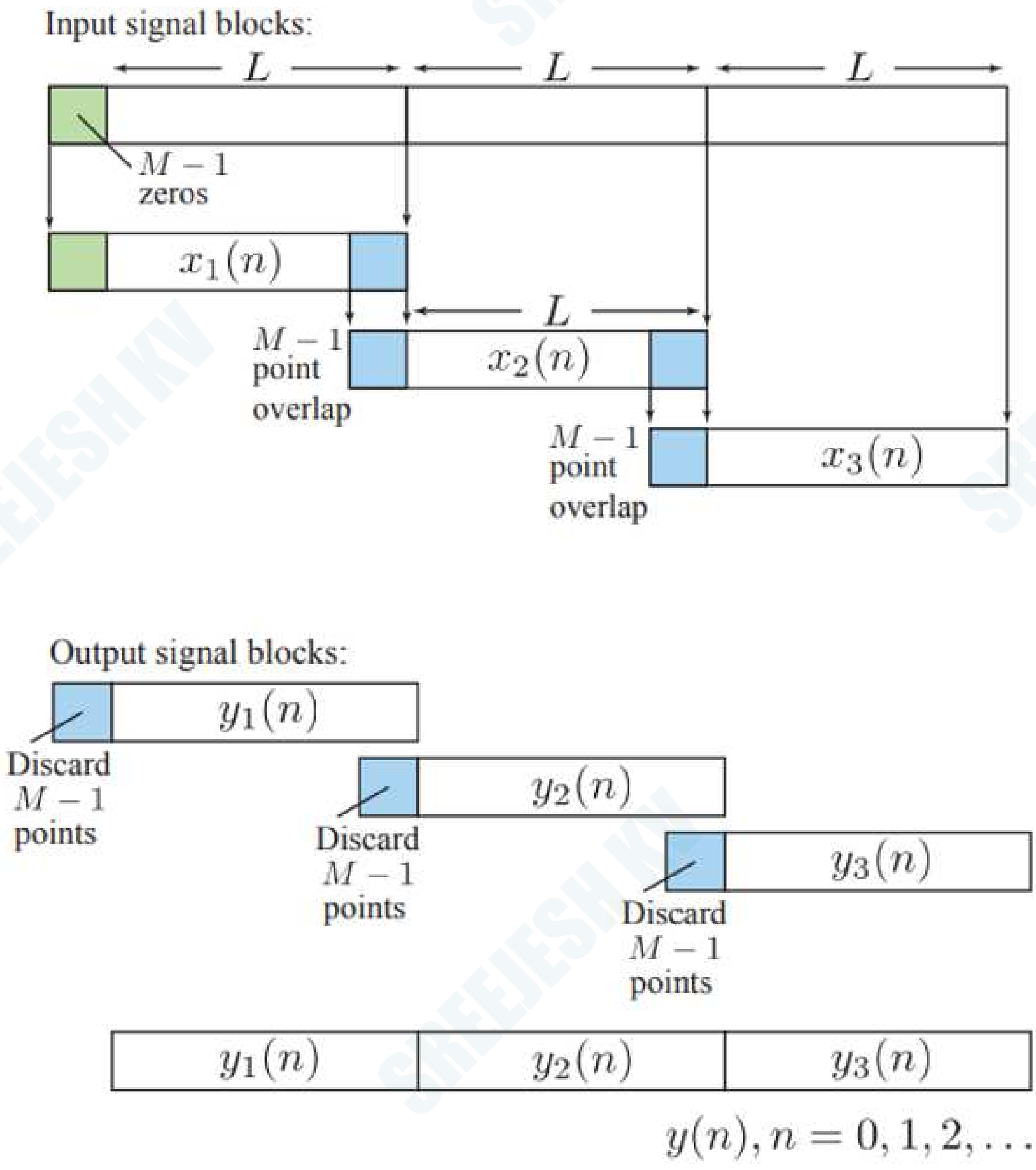


Figure 5.2: Overlap-Save Method

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 second next multiple of L after $N1$.

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

Step 5. Create subsequences of length L from $x[n]$ and precede each subsequence by the last $M - 1$ values of the previous subsequence. For the first block, precede with $M - 1$ zeros.

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

Step 7. Obtain the output sequence $y[n]$ by fitting the output subsequence after discarding the first $M - 1$ values from each subsequence.

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

Step 9. Stop

PROGRAM

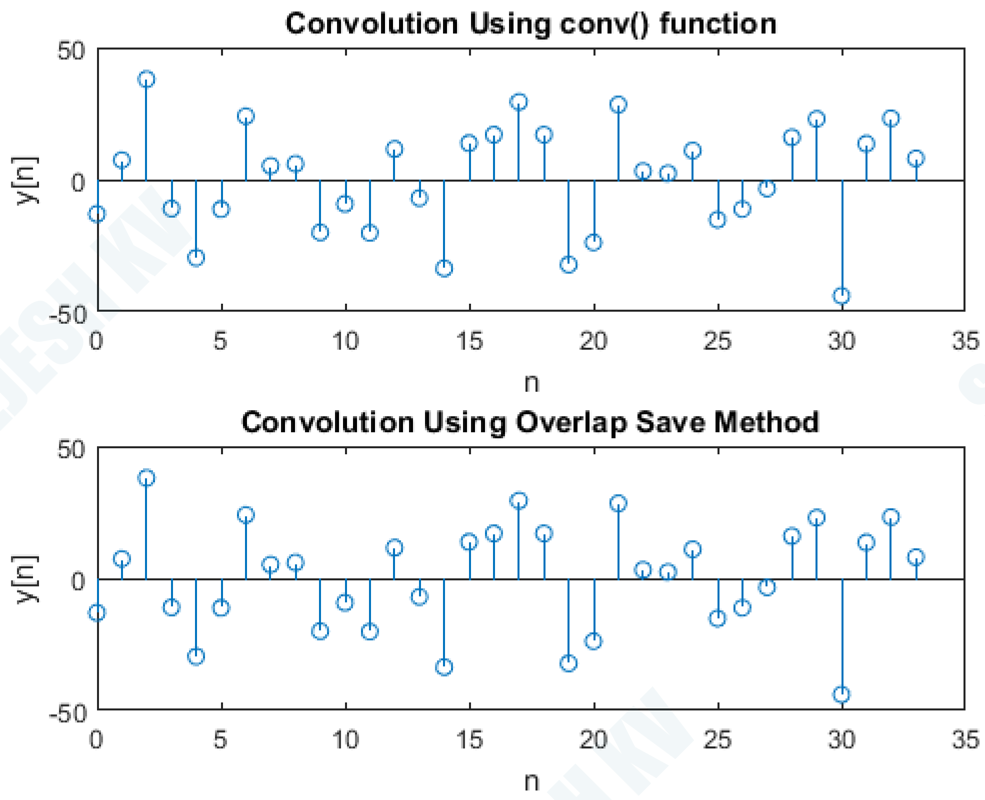
```
%Title: Program to Perform linear convolution through overlap ...
    save method
%and verify the results using the builtin function - conv

%Author: Sreejesh K V, Dept. of ECE, GCEK
%Date: 01-10-2022

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 new 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 Save method for computing the linear convolution -- %
x=[x zeros(1,mod(-N1,L)) zeros(1,L)];%zero pad x to the length ...
    which is the second next multiple of L

N2=length(x);%length after zero padding; 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 to take
index=1:L;%index of first set of L values to be taken from x[n]
y=[];%the output sequence initialized as empty
for stage=1:S
if stage==1  % first input sub seq : M-1 zeros followed by first ...
    L values from x[n]
xm=[zeros(1,M-1) x(index)];
else
xm=x(index);
end
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]

index2=M:N;%index of non-aliased values in ym
ym=ym(index2);%discarding first M-1 (aliased) Samples
y=[y ym];%appending the non-aliased values to the o/p sequence

index=(((stage)*L)-M+2):((stage+1)*L);  % next stage ...
    index(Previous M-1 values followed by L new values)
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 Save Method')
xlabel('n');
ylabel('y[n]');
```

OUTPUT & OBSERVATIONS

Figure Window Output:



RESULTS

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