Experiment No:08 Date: 03-10-2024

OVERLAP ADD AND OVERLAP SAVE METHOD

Aim

To simulate linear convolution of two signals using overlap add and overlap save methods

Theory

1. Overlap-Add Method

This procedure cuts the signal up into equal length segments with no overlap. Then it zero-pads the segments and takes the DFT of the segments. Part of the convolution result corresponds to the circular convolution. The tails that do not correspond to the circular convolution are added to the adjoining tail of the previous and subsequent sequence. This addition results in the aliasing that occurs in circular convolution.

2. Overlap-Save Method

This procedure cuts the signal up into equal length segments with some overlap. Then it takes the DFT of the segments and saves the parts of the convolution that correspond to the circular convolution. Because there are overlapping sections, it is like the input is copied therefore there is not lost information in throwing away parts of the linear convolution.

Program

1) Overlap add method

```
clc;
clear all;
close all;
% Step 1: Define the Input Signal and Filter
Xn =input('enter the elements of xn: ');
Hn = input('enter the elements of hn: ');
L = 5;
NXn = length(Xn);
M = length(Hn);
```

```
M1 = M - 1;
R = rem(NXn, L);
N = L + M1;
% zero padding
Xn = [Xn, zeros(1, L - R)];
Hn = [Hn, zeros(1, N - M)];
%overlap add method
K = floor(length(Xn) / L);
y = zeros(1, length(Xn) + M - 1);
z = zeros(1, M1);

for k = 0:K-1
    startIndex = L * k + 1;
    endIndex = min(startIndex + L - 1, length(Xn));
    Xnp = Xn(startIndex:endIndex);

    Xnk = [Xnp, z];
    convSegment = mycirconv(Xnk, Hn);

    % Add the current segment to the output
    outputStart = startIndex;
    outputEnd = outputStart + N - 1;
    y(outputStart:outputEnd) = y(outputStart:outputEnd) +
convSegment(1:N);
end
disp('Input Signal:');
disp(Xn);
disp('Filter:');
disp(Hn);
disp('Output Signal (Convolved using Overlap-Add Method):');
```

```
disp(y);
% Function for Circular Convolution
function y = mycirconv(x, h)
    % Compute the circular convolution using FFT
    N = max(length(x), length(h));
    y = ifft(fft(x, N) .* fft(h, N)); % FFT-based circular
convolution
end
```

2) Overlap save method

```
clc;
clear all;
close all;
x = input('enter the elements of x: ');
h = input('enter the elements of h: ');
N = 5;
lx = length(x);
lh = length(h);
m = lh - 1;
x = [zeros(1, m), x, zeros(1, N-1)];
h = [h, zeros(1, N - lh)];
L = N - lh + 1;
k = floor((length(x) - m) / L);
p = [];
for i = 0:k-1
    y = x(i * L + 1 : i * L + N);
    q = mycirconv1(y, h);
    p = [p, q(lh:N)];
end
disp("Output Signal (Convolved using Overlap-Save Method):");
```

```
disp(p);
function y = mycircconv1(x, h)
    N = length(x);
    y = ifft(fft(x, N) .* fft(h, N));
end
```

Result

Performed overlap add and overlap save method using MATLAB and verified the output.

Observation

1)Overlap add method

```
Input signal

Xn =[2,-2,8,-2,-2,-3,-2,1,-1,9,1,3];

Hn = [1,2,3];


output Signal (Convolved using Overlap-Add Method)


2.0000    2.0000   10.0000    8.0000   18.0000  -13.0000  -14.0000
-12.0000   -5.0000   10.0000   16.0000   32.0000    9.0000    9.0000
```

2)Overlap save method

```
Input signal

x = [1 2 3 4 5 6 7 8 9 10];

h = [1 1 1];


Output Signal (Convolved using Overlap-Save Method):


1.0000    3.0000    6.0000    9.0000   12.0000   15.0000   18.0000
21.0000   24.0000   27.0000   19.0000   10.0000
```