Experiment – 8

DSP – LAB (EE2801)

Name – Aman Kumar

Roll no – EE22BTECH11006.

Aim: To understand and implement interpolation and decimation techniques in digital signal processing.

Theory:

1. **Interpolation:**
    - This involves increasing the sampling rate of a signal by inserting additional samples between existing samples.
    - This is typically achieved through up sampling followed by a low pass filter to remove the high frequency components introduced in the up-sampling process.
    - So, when we up sample the signal by L-fold i.e. we are inserting (L-1) zero valued samples for each input sample.
    - Thus, the sampling rate increases from Fs to LFs.

The mathematical definition of L-fold interpolation:

$$y[n] = L \sum_{k=-\infty}^{\infty} h[k]w[n-k]$$

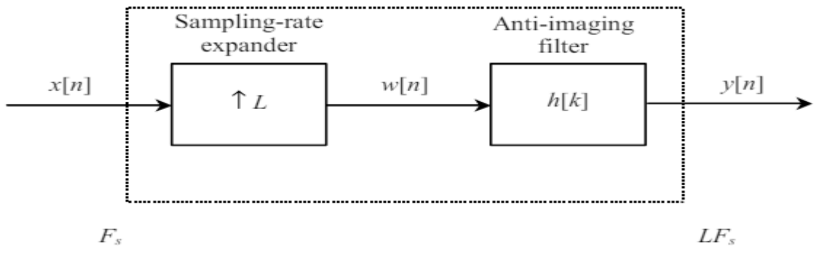


Figure 9.3: Block diagram notation of interpolation, by a factor of L.

2.**Decimation:**

- Decimation is exact opposite of Interpolation; it involves reducing the sampling rate of a signal by discarding samples.
- This is achieved by a low pass filter followed by a down-sample, where every Mth sample is retained.
- The low pass filter in decimation prevents aliasing by attenuating high frequency components before down sampling.
- In decimation the sampling rate is reduced from Fs to Fs/M by discarding the M-1 samples for every M sample in the original signal.

The formal definition of M-fold decimation.

$$y[n] = \sum_{k=-\infty}^{\infty} h[k]x[nM-k]$$

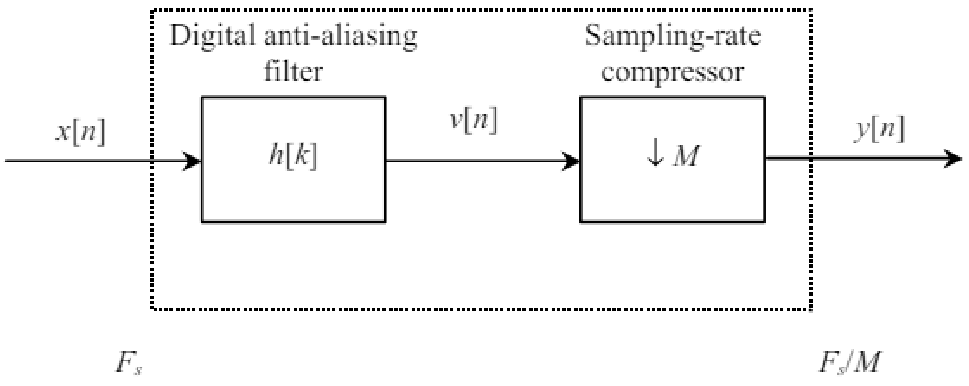


Figure 9.1: Block diagram notation of decimation, by a factor of M.

- **Deciding filter cutoffs:**

Interpolation

Let the original signal has a bandwidth of π i.e. [-π/2,π/2], So after up sampling it will become from [-π/4, π/4] i.e. in the spectrum there will be unwanted image so for that we need to apply a low pass filter of cutoff as π/L.

Decimation

In this case if the original signal bandwidth is π i.e. [-π/2, π/2] then after down sampling the range is [-π, π], now this can cause aliasing if we down sample it by more factor, So the cutoff frequency is to kept at π/M as the bandwidth should be less than 2π/M so that the cut-off frequency is π/M which ensures that no aliasing will occur.

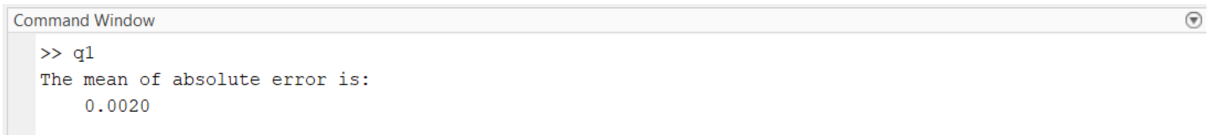
MATLAB CODE:

```
f1 = 500;
f2 = 1000;
f3 = 700;
fs =8000;
wc_1 = pi/2;
wc_2 = pi/2;
```

```matlab
N = 39;
t = 0:1/fs:1;

x_t = sin(2*pi*f1*t)+2*sin(2*pi*f2*t)+1.5*sin(2*pi*f3*t);

L = 2;
M = 2;

x_1 = u_sample(x_t,L);
x_i = L*lowpassfilter(x_1,wc_1,N);%output of interpolation is x_i
x_3 = lowpassfilter(x_i,wc_2,N);
x_4 =d_sample(x_3,M);
x_4 = x_4((N+1)/2:fs+(N+1)/2);

err = 0;
for i = 1:length(x_4)
    err = err+abs( x_4(i)-x_t(i));

end
disp("The mean of absolute error is: ");
disp(err/length(x_t));

function up_sample=u_sample(x,f)
    m=length(x);
    up_sample=zeros(1,m*f);
    for i=1:m
        up_sample((i-1)*f+1)=x(i);
    end
end

function [y] = lowpassfilter(x, wc, N)

    h_d = zeros(1,N);
    for n = -(N - 1) / 2 : (N - 1) / 2
        if n == 0
            h_d(n + (N - 1) / 2 + 1) = wc/pi;
        else
            h_d(n + (N - 1) / 2 + 1) = sin(wc*n)/(pi*n);
        end
    end

    w_h = zeros(1,N);
    for k= 0:N-1
        w_h(k+1) = 0.54-0.46*cos(2*pi*k/(N-1));
    end
    h_n = h_d .* w_h;
    y = conv_calculator(x, h_n);
end

function ans_conv = conv_calculator(x, h)
    n = length(x);
    m = length(h);
    N = [x, zeros(1, m - n - 1)];
    M = [h, zeros(1, n - 1)];
    Y = zeros(1, m + n - 1);

    for i = 1:m + n - 1
        for j = 1:n
            if (i - j + 1 > 0 && i - j + 1 <= m)
```

```
                Y(i) = Y(i) + N(j) * M(i - j + 1);
            end
        end
    end
    ans_conv = Y;
end

function down_sample=d_sample(x,f)
    m= length(x);
    down_sample=zeros(1,floor(m/f));
    for i=1:f:m
        down_sample((i+f-1)/f)=x(i);
    end
end
```

Output:



```
>> q1
The mean of absolute error is:
    0.0020
```

Observation:

The given MATLAB code demonstrates the process of interpolation and decimation on an input signal.

- The input signal x_t is a sum of three sinusoidal signals with frequencies 500 Hz, 1000 Hz, and 700 Hz, respectively.
- The input signal is first interpolated (up sampled) by a factor of 2 using a low-pass filter with a cutoff frequency of (π/2).
- After interpolation, the signal is then decimated (down sampled) by a factor of 2 using another low-pass filter with the same cutoff frequency.
- The final signal x_4 is compared to the original signal x_t to calculate the mean absolute error. The mean absolute error is very small (0.002), indicating that the processed signal is close to the original signal.

Conclusion:

- The code successfully demonstrates the process of interpolating and decimating an input signal using MATLAB.
- The mean absolute error is a measure of the similarity between the original signal and the processed signal. The low value of the error suggests that the interpolation and decimation processes were implemented effectively, retaining the key characteristics of the original signal.
- The use of a low-pass filter during both interpolation and decimation helps in removing high-frequency components and avoiding aliasing, resulting in a smooth and accurate signal transformation.
- The use of specific cutoff frequencies (π /2) is essential in ensuring that the signal remains within the desired frequency range and prevents aliasing.