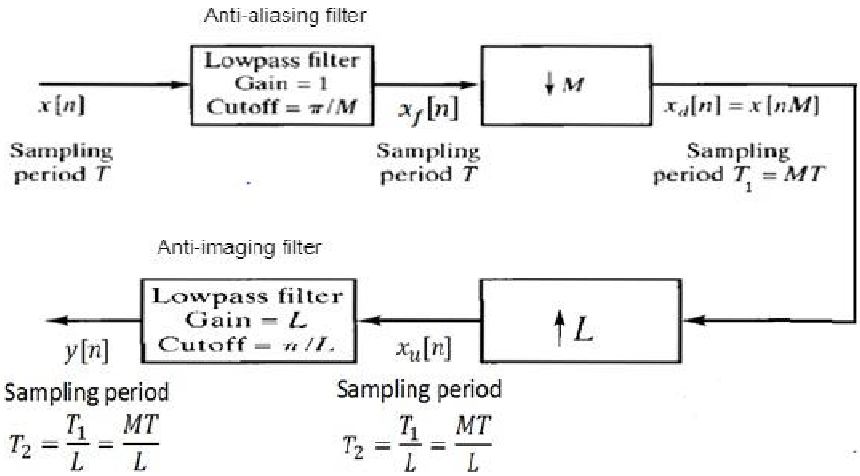
EE2801/EE5802: DSP Lab

Assignment 2

Problem:

Implementation of decimation and interpolation.

Technical details:



Input:

$$x[n] = \sin(2\pi f_0 n/f_s) + 0.5\sin(2\pi f_1 n/f_s) + 0.6\sin(2\pi f_2 n/f_s)$$

where, $f_0 = 100Hz$, $f_1 = 200Hz$, $f_2 = 300Hz$, $f_s = 2400\ Hz$

Generate 96 samples of input, where n= 0 to 95.

Downsampler: $x_d[n] = x[Mn]$

Upsampler:

$$x_u[n] = \begin{cases} x_d[n/L], & \textit{if n is a multiple of L} \\ 0, & \textit{otherwise} \end{cases}$$

1. **Decimation and interpolation by factor 2 (M=L=2):**

   LPF(HBF) specifications

   - $Anti\ aliasing\ Gain = 1, Anti\ imaging\ Gain = L$
   - $Cutoff\ frequency\ (f_c) = 600\ Hz$
   - $Sampling\ frequency\ (f_s) = 2400\ Hz$
   - $Digital\ cutoff\ frequency\ (\omega_c) = \frac{\pi}{2}$
   - $Number\ of\ samples\ (N) = 101$

2. **Decimation and interpolation by factor 4 (M=L=4):**

   LPF specifications

   - $Anti\ aliasing\ Gain = 1, Anti\ imaging\ Gain = L$
   - $Cutoff\ frequency\ (f_c) = 300\ Hz$
   - $Sampling\ frequency\ (f_s) = 2400\ Hz$
   - $Digital\ cutoff\ frequency\ (\omega_c) = \frac{\pi}{4}$
   - $Number\ of\ samples\ (N) = 101$

3. **Decimation and interpolation by factor 8 (M=L=8):**

   LPF specifications

   - $Anti\ aliasing\ Gain = 1, Anti\ imaging\ Gain = L$
   - $Cutoff\ frequency\ (f_c) = 150\ Hz$
   - $Sampling\ frequency\ (f_s) = 2400\ Hz$
   - $Digital\ cutoff\ frequency\ (\omega_c) = \frac{\pi}{8}$
   - $Number\ of\ samples\ (N) = 101$

Solution for above Problem

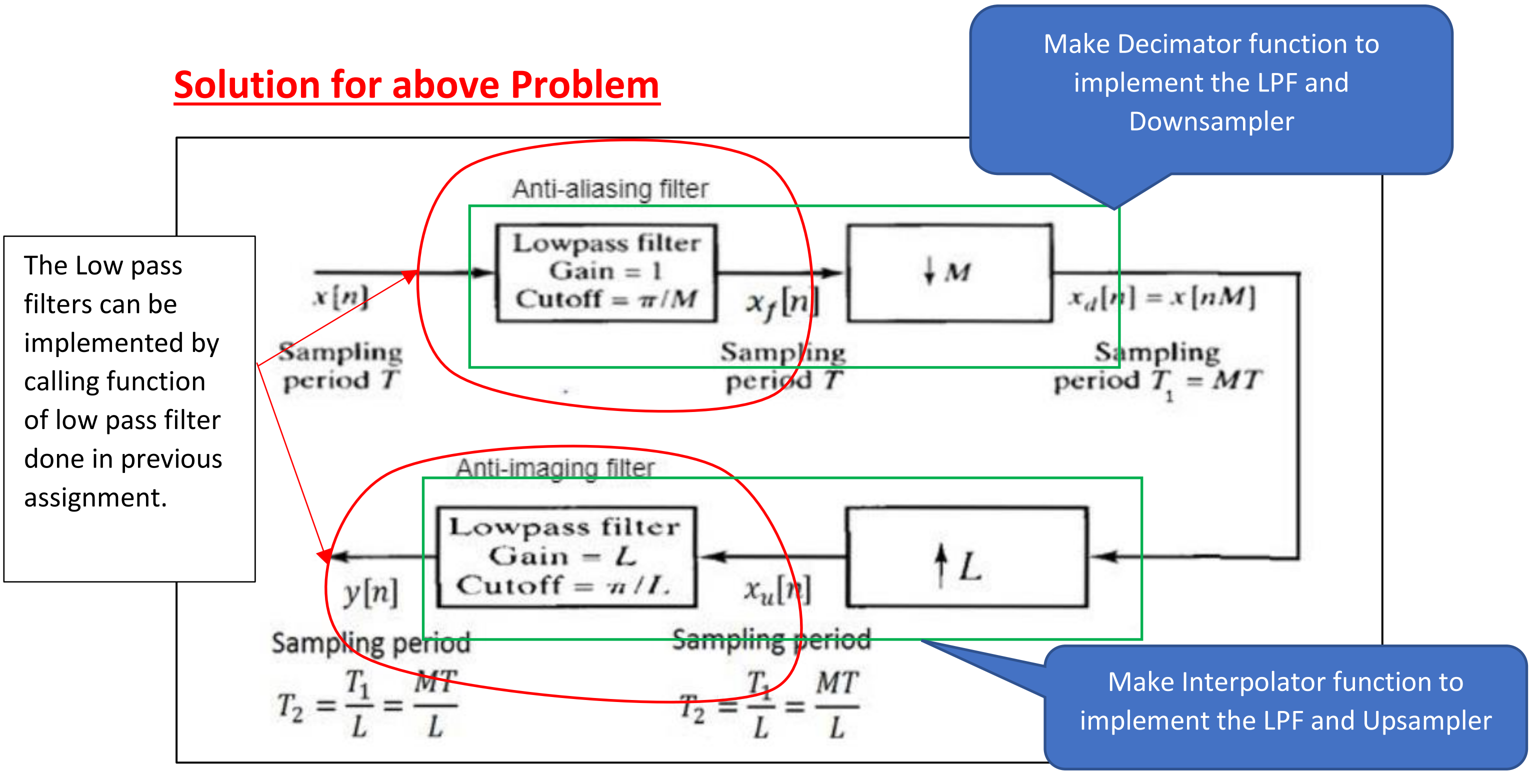


Figure 1. Setup Diagram to implement the Interpolator – Decimator

1. Use the above diagram as a reference point architecture of the Interpolator – Decimator code.
2. First, we will include the function of the low pass filter that we created in the previous assignment.
3. The Input to the function of the filter will be the Fc(Cut off frequency), Fs (Sampling Frequency) and N(No. of samples).
4. The data given for the filter is same for the anti-aliasing filter and imaging filter.
5. Create the input signal as combination of sinusoids as given in the function. The loop will go from 0 to 95 to create all the 96 data points.

<u>Implement the Interpolator function</u>

6. Use the LPF function to create the Anti-Aliasing filter.
7. Filter data points and the input signal will be convolved to get output of the filter.

8. Since the number of samples at the output of the filter will increase from desired data points (Due to convolution), we will make the output signal of the desired length.
9. Now pass this through a down-sampler to get the decimated output.

Implementation of Interpolator

10. The use of interpolate function consist of two parts. One part is up-sampler after which it is to be passed with the anti-imaging filter.
11. Anti-imaging filter will convolute the sequence with the filter's data points. These needs resizing in order to meet the number of samples as in the original signal.
12. After resizing next task is to compensate for the gain of 1/L. Which can be done by scalar multiplication of the gain.

Results of the above said operation

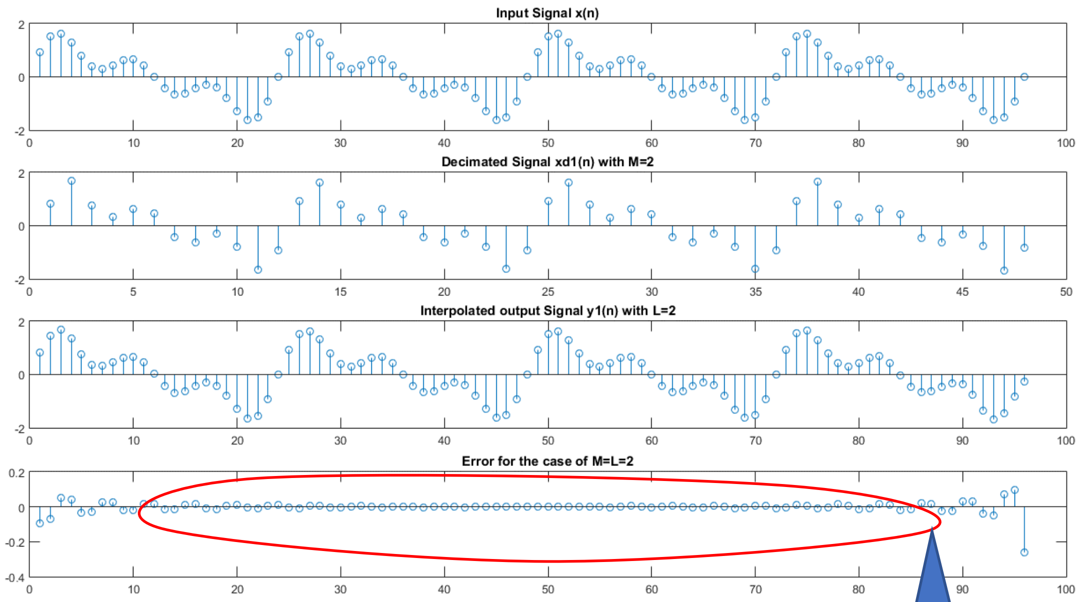
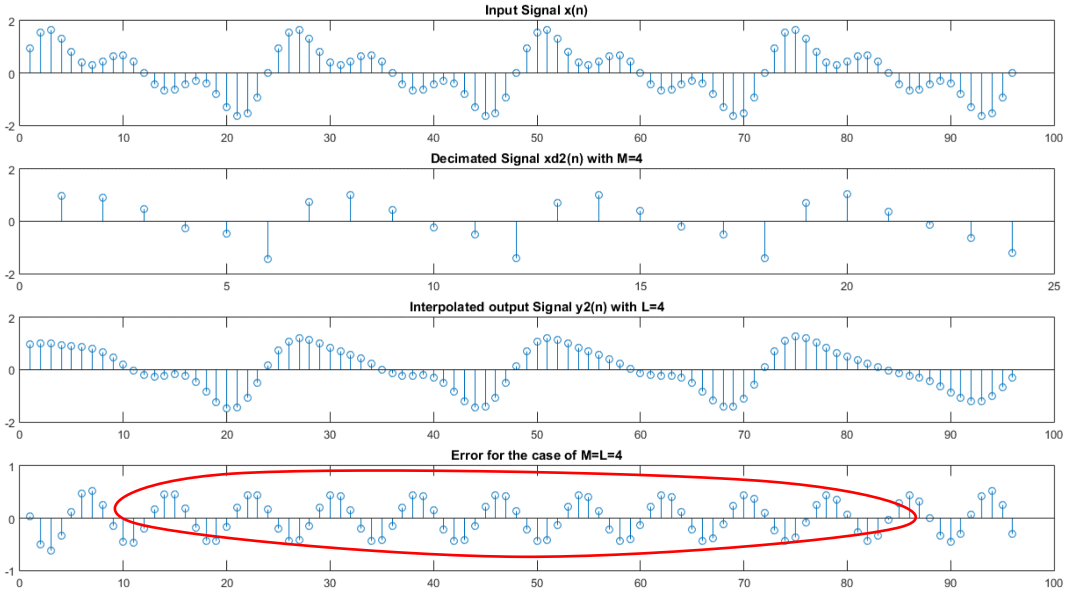


Figure 2 Interpolate and Decimation function when M=L=2

Least error occurred in case of M=L=2



We can see that there is subsequent increase in the error portion.

Figure 3 Interpolate and Decimation function when M=L=4

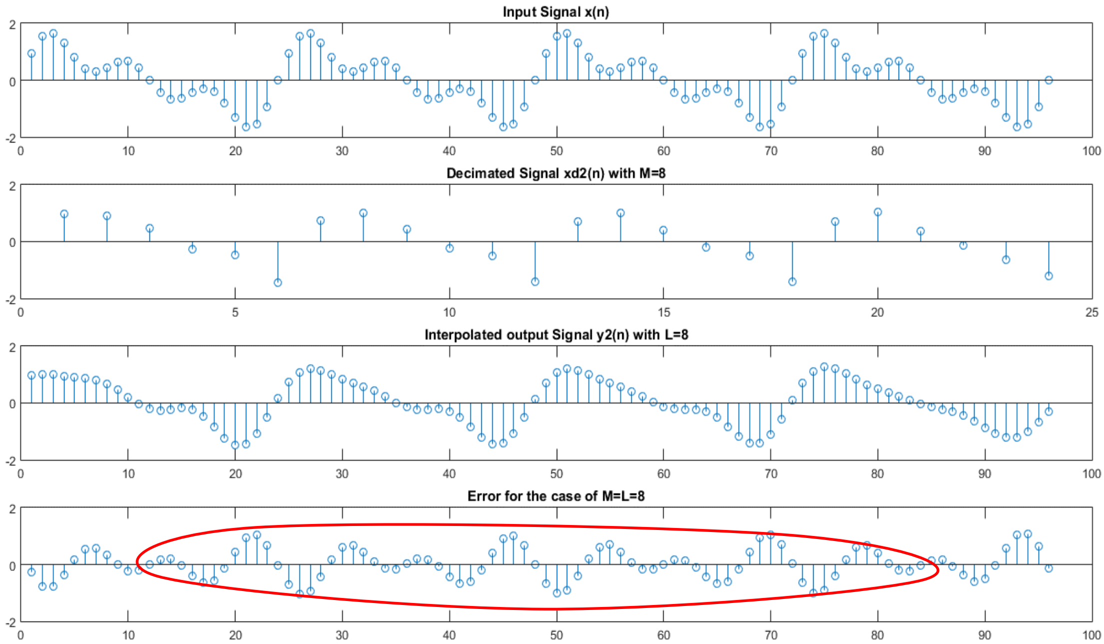
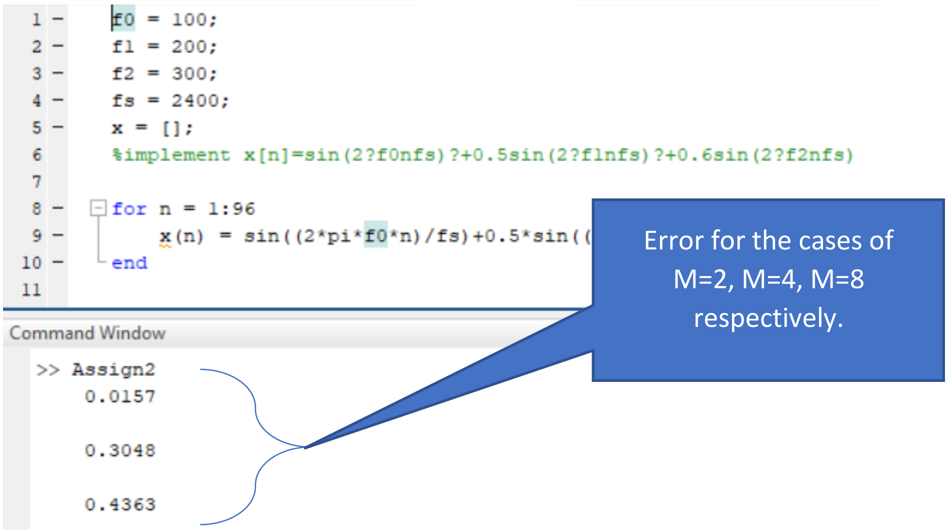


Figure 4 Interpolate and Decimation function when M=L=8

Above said observations are supported by the below calculated mean error of the coding done. Please see below image.



Conclusion and Learning

- The above function and results help us understand how the samples are increased and decreased.
- The importance of Anti-imaging and Anti-Aliasing filter.
- Error in cases of various up/down sampling cases.
- Building function for processing signals.