Digital Signal Processing

1 Signal Generation

Consider the continuous-time signal $\sin(\Omega t)$, where $\Omega = 2\pi f$. The range of f: $-\infty < f < \infty$. Ω is known as the analog frequency.To convert it into a discrete-time signal, put $t = nT_s$, where $T_s = \frac{1}{f_s}$ is the sampling interval and f_s is the sampling frequency.

$$\sin(2\pi f n T_s) = \sin(2\pi \frac{f}{f_s}) \tag{1}$$

$$= \sin(\omega n) \tag{2}$$

where ω is the digital frequency.

Analog frequencies are represented by Ω, while digital frequencies are represented by ω. $\omega = \Omega T_s$, where Ω lies in $(-\infty, \infty)$, and ω ranges from $[-\pi, \pi]$.

2 Frequency Analysis

The input signal is analyzed in the frequency domain using the Discrete Fourier Transform (DFT). The DFT equation is given by:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j2\pi \frac{kn}{N}}$$

3 Filtering

Filtering the input signal using a predefined filter impulse response h. This is achieved through convolution. Convolution can be mathematically expressed as follows:

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

Where:

- $x[n]$ is the input signal.
- $h[n]$ is the filter impulse response.
- $y[n]$ is the filtered output signal.

The process of filtering can be mathematically represented as follows:

$$y[n] = x[n] * h[n] \tag{3}$$

3.1 Types of Digital Filters

There are two main types of digital filters:

1. FIR (Finite Impulse Response) Filters
2. IIR (Infinite Impulse Response) Filters

FIR filters are generally easier to design in discrete time compared to IIR filters. FIR filters can exhibit either linear or non-linear phase responses. The window method is one of the simplest methods for FIR filter design. In this method, the goal is often to design a linear phase FIR filter to avoid phase distortions.

3.2 Window Method for FIR Filter Design

In the window method for FIR filter design, the process involves the following steps:

1. Define the desired ideal frequency response $H_d(e^{j\omega})$.
2. Compute the inverse discrete Fourier transform (IFFT) of $H_d(e^{j\omega})$ to obtain the impulse response $h_d[n]$.
3. Since $h_d[n]$ is of infinite length, it needs to be truncated using a finite-length window function $w[n]$ to obtain the practical impulse response $h[n]$.

$$h[n] = h_d[n] \times w[n] \tag{4}$$

3.3 Commonly Used Window Functions

Several commonly used window functions include:

- Rectangular window

  The rectangular window function, also known as the boxcar window, is defined as:

$$w[n] = \begin{cases} 1, & \text{for } 0 \leq n < N \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

- Bartlett (Triangular) Window

  The Bartlett window, also known as the triangular window, is defined as:

$$w[n] = \begin{cases} 1 - \frac{|n-(N-1)/2|}{(N-1)/2}, & \text{for } 0 \leq n < N \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

- Hamming Window

  The Hamming window is defined as:

$$w[n] = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right), \quad \text{for } 0 \leq n < N \tag{7}$$

- Hanning (Hann) Window

  The Hanning window, also known as the Hann window, is defined as:

$$w[n] = 0.5 - 0.5\cos\left(\frac{2\pi n}{N-1}\right), \quad \text{for } 0 \leq n < N \tag{8}$$

- Blackman Window

  The Blackman window is defined as:

$$w[n] = 0.42 - 0.5\cos\left(\frac{2\pi n}{N-1}\right) + 0.08\cos\left(\frac{4\pi n}{N-1}\right), \quad \text{for } 0 \leq n < N \tag{9}$$

3.4 Low-Pass Filter

The impulse response of a Low-Pass Filter (LPF) is characterized by the expression below, where $h_d[n]$ represents the filter's response at discrete time indices n.

$$h_d[n]\Big|_{n=0} = \lim_{n\to 0} h_d[n] \tag{10}$$

$$\tag{11}$$

For the LPF, we can express $h_d[n]$ as:

$$h_d[n] = \frac{\sin(\omega_c n)}{\pi \omega_c n} \quad \text{for} \quad -\frac{N-1}{2} \leq n \leq \frac{N-1}{2}$$
$$\text{where } \omega_c = \frac{\omega_c}{\pi} \quad \text{and} \quad \omega_c = \frac{2\pi f_c}{F_s}. \tag{12}$$

Therefore, for $n = 0$, we have:

$$\begin{aligned} h_d[n]\Big|_{n=0} &= \lim_{n\to 0} \omega_c \frac{\cos(\omega_c n)}{\pi n} \\ &= \lim_{n\to 0} \frac{\omega_c n}{\pi n} = \frac{\omega_c}{\pi} \end{aligned} \tag{13}$$

For the LPF, the impulse response $h_d[n]$ is given by:

$$h_d[n] = \begin{cases} \frac{\sin(\omega_c n)}{\pi n}, & -\frac{N-1}{2} \leq n \leq \frac{N-1}{2}, \\ \\ \frac{\omega_c}{\pi}, & n = 0, \end{cases}$$

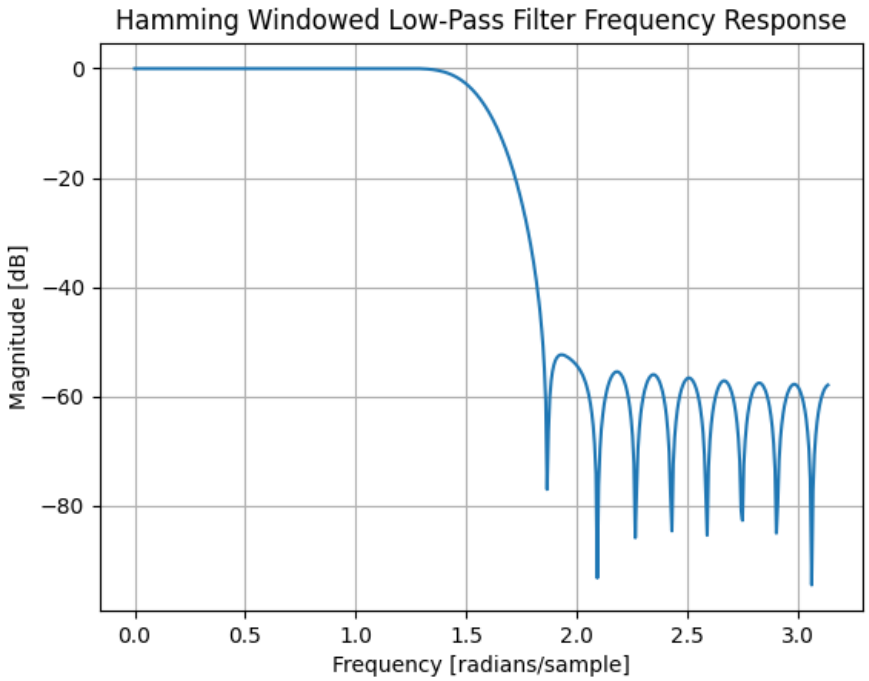


Figure 1: Frequency Response

/codes/lpf.py

3.5 Band-Pass Filter (BPF)

To design a Band-Pass Filter (BPF), the impulse response $h_d[n]$ is constructed as follows using the Inverse Fast Fourier Transform (IFFT):

$$h_d[n] = \frac{\sin(\omega_{c2} n)}{\pi n} - \frac{\sin(\omega_{c1} n)}{\pi n} \tag{14}$$

Where:

- ω_{c1} and ω_{c2} are the normalized cutoff frequencies of the filter.

- f_{c1} and f_{c2} are the corresponding actual cutoff frequencies.
- n is the discrete time index.

The idea behind designing a Band-Pass Filter (BPF) involves subtracting the magnitude response of a Low-Pass Filter (LPF) with cutoff frequency $\omega_{c1}$from another LPF magnitude response with cutoff frequency ω_{c2}.

For the BPF, the impulse response $h_d[n]$ is given by:

$$h_d[n] = \begin{cases} \frac{\sin(\omega_{c2}n)}{\pi n} - \frac{\sin(\omega_{c1}n)}{\pi n}., & \text{for } -\frac{N-1}{2} \leq n \leq \frac{N-1}{2}, \\ \frac{\omega_{c2}-\omega_{c1}}{\pi}, n = 0, \end{cases} \tag{15}$$

For practical implementation, the impulse response of the BPF is often windowed by a window function $w[n]$, resulting in the practical impulse response

$$h[n] = h_d[n] \times w[n] \tag{16}$$

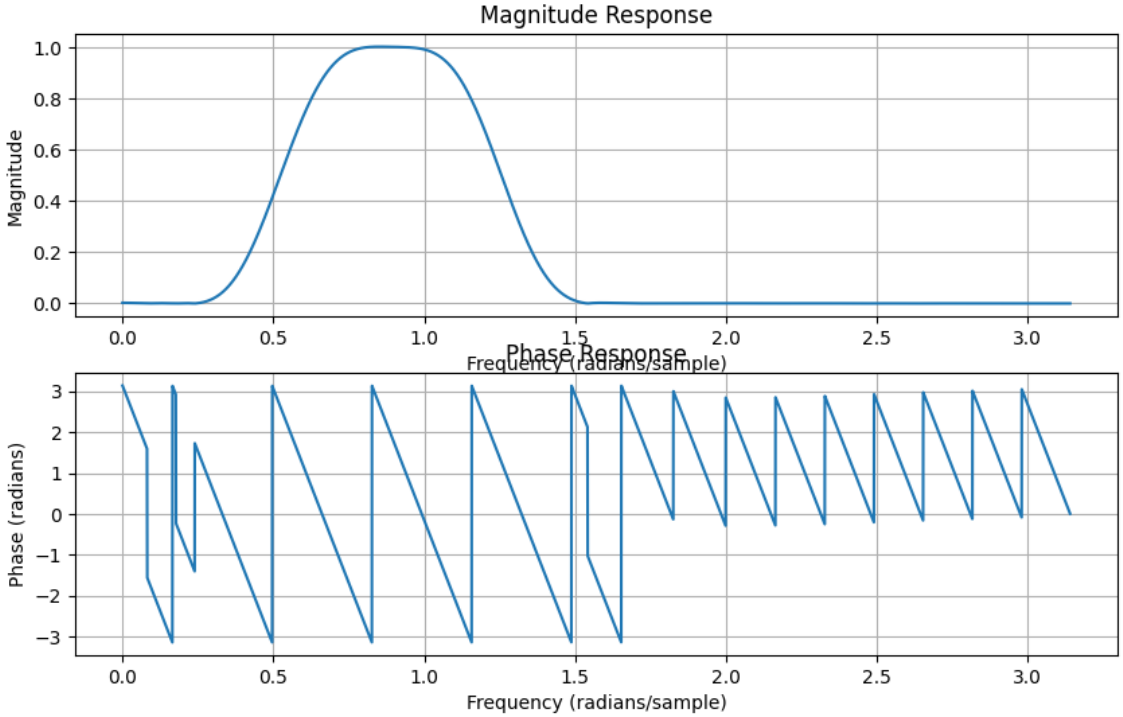


Figure 2: Magnitude and Phase Responses of the Hamming Windowed FIR Filter

/codes/bandpass.py

This Python script generates and plots the magnitude and phase responses of a Hamming windowed FIR filter, as shown in 2

3.6 Half Band Filter (HBF)

The Half Band Filter (HBF) is a special case of a Low-Pass Filter (LPF) with a cutoff frequency $f_c = \frac{f_s}{4}$, where f_s is the sampling frequency.

The impulse response of the half band filter is given by:

$$h(2n) = \begin{cases} \frac{1}{2}, & n = 0 \\ 0, & n \neq 0 \end{cases}$$

Where:

- n is the discrete time index.

The HBF has a unique property where it passes half of the frequency spectrum while attenuating the other half. Its impulse response consists of a central value of $\frac{1}{2}$ at $n = 0$ and zeros for all other discrete time indices.

3.7 M Band Filter

The M Band Filter is a generalization of the Half Band Filter (HBF), designed to allow for the selective passage of specific frequency bands within the spectrum.

The impulse response of the M Band Filter is given by:

$$h(Mn) = \begin{cases} c, & n = 0 \\ 0, & n \neq 0 \end{cases}$$

Where: n is the discrete time index and c is a constant scaling factor.

- For the Half Band Filter, $c = \frac{1}{2}$, and cut-off frequency :$\omega_c = \frac{\pi}{2}$
- For the M Band Filter, $c = \frac{1}{M}$, and cut-off frequency :$\omega_c = \frac{\pi}{M}$

4 Downsampling

Downsampling is the process of reducing the number of samples in a signal. Given an input signal $x[n]$, downsampling by a factor of M produces a new signal

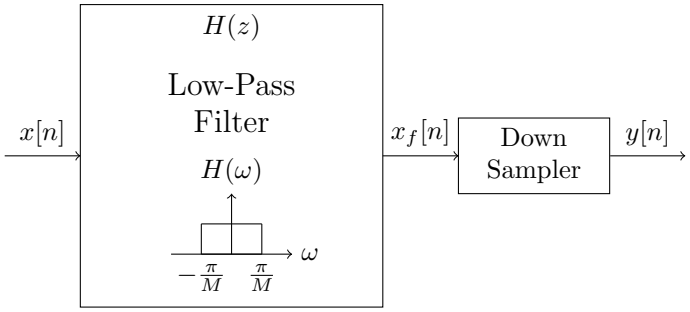
$$y[n] = x[Mn]. \tag{17}$$

Time Varying (LTV) System The system described in the code can be characterized as a Linear Time Varying (LTV) system. In the frequency domain, there exists a relation between the input and output of the system.

For a given value of M, the frequency domain relation between the input $X(e^{j\omega})$ and the output $Y(e^{j\omega})$ can be expressed as:

$$Y(e^{j\omega}) = \frac{1}{M} \sum_{k=0}^{M-1} X\left(e^{j(\omega - 2\pi k)/M}\right) \tag{18}$$

5 Decimation



- LPF followed by downsampler is known as a decimator.
- The primary role of the LPF is to prevent aliasing, earning it the name anti-aliasing filter.
- The cutoff frequency for the LPF is set to π/M.
- When $M = 2$, the LPF becomes a Half Band Filter (HBF) with a cutoff frequency of $\pi/2$.

The Python code below is accompanied by corresponding plots shown in

/codes/decimation.py

In Figure 3, the input signal composed of multiple sinusoidal components is shown. The output after applying a low-pass filter is depicted in Figure 4. The effect of decimation is visible in Figure 5, and the outcome of downsampling without a low-pass filter is illustrated in Figure 6.

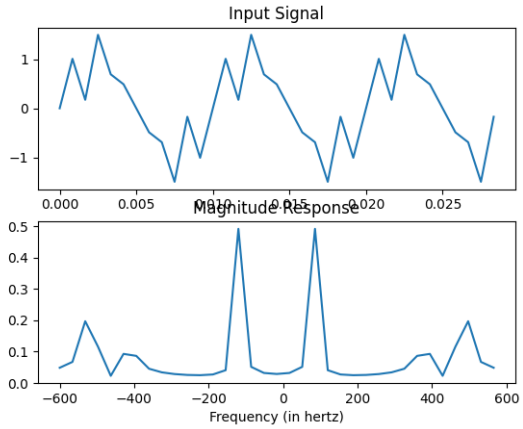


Figure 3: Input Signal

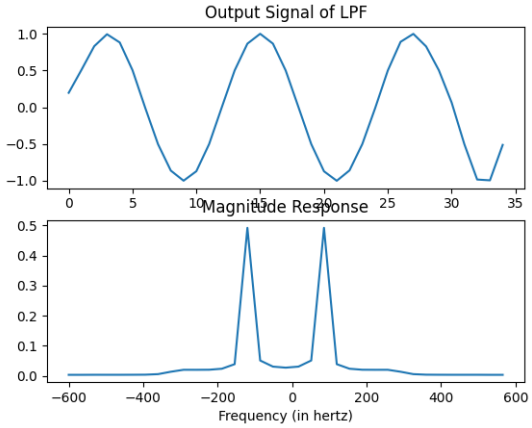


Figure 4: Output Signal of Low-Pass Filter

Upsampler (Expander)

- **Time domain relation between input and output:**

$$y[n] = \begin{cases} x[n/L], & \text{if } n \text{ is a multiple of } L \\ 0, & \text{otherwise} \end{cases}$$

- **Frequency domain relation between input and output:**

$$Y(e^{j\omega}) = X(e^{j(\omega/L)}), \quad \text{for } 0 \leq \omega < 2\pi$$

6 Interpolation

- Upsampler followed by LPF is known as an interpolator.
- The job of LPF is to remove the unwanted image of $Xe^{j\omega}$. Hence, it is known as an anti-imaging filter.
- Cutoff frequency is $\frac{\pi}{L}$.
- When $L = 2$, then the LPF is also known as a Half Band Filter (HBF) with a cutoff frequency of $\frac{\pi}{2}$.

The Python code demonstrates upsampling and interpolation of a discrete input signal using a half-band filter. It generates plots illustrating the signal processing steps and visualizes the magnitude response in the frequency domain.

/codes/interpolation.py

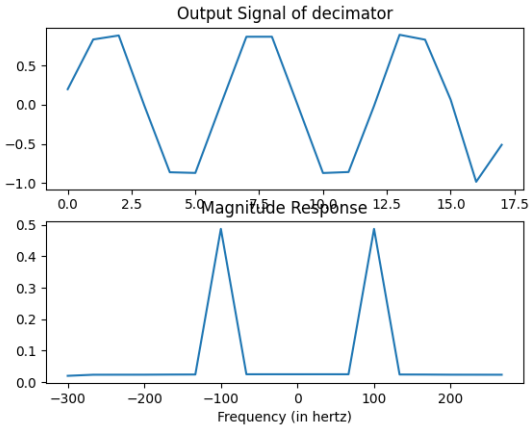


Figure 5: Output Signal of Decimator

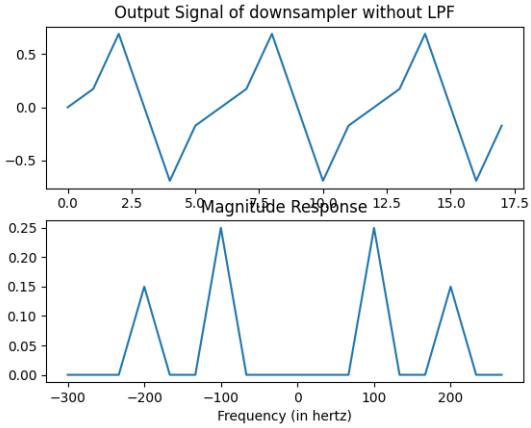


Figure 6: Output Signal of Downsampler Without LPF

In the results presented in Figure 8, we can observe the original input signal. The interpolated output signal and its corresponding frequency response are illustrated in Figure 9. Similarly, Figure 10 provides insight into the upsampled input signal and its frequency response.

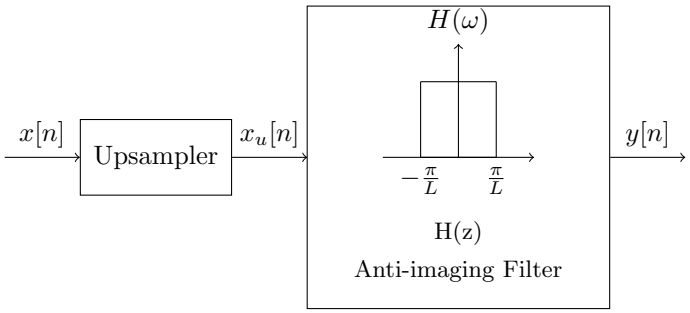


Figure 7: Interpolation and Filtering Diagram

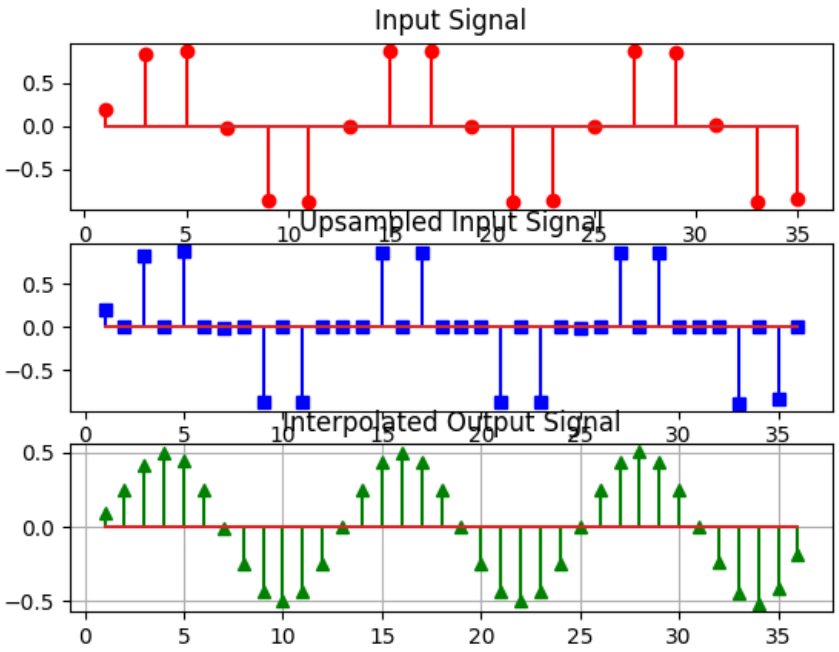


Figure 8: Input Signal

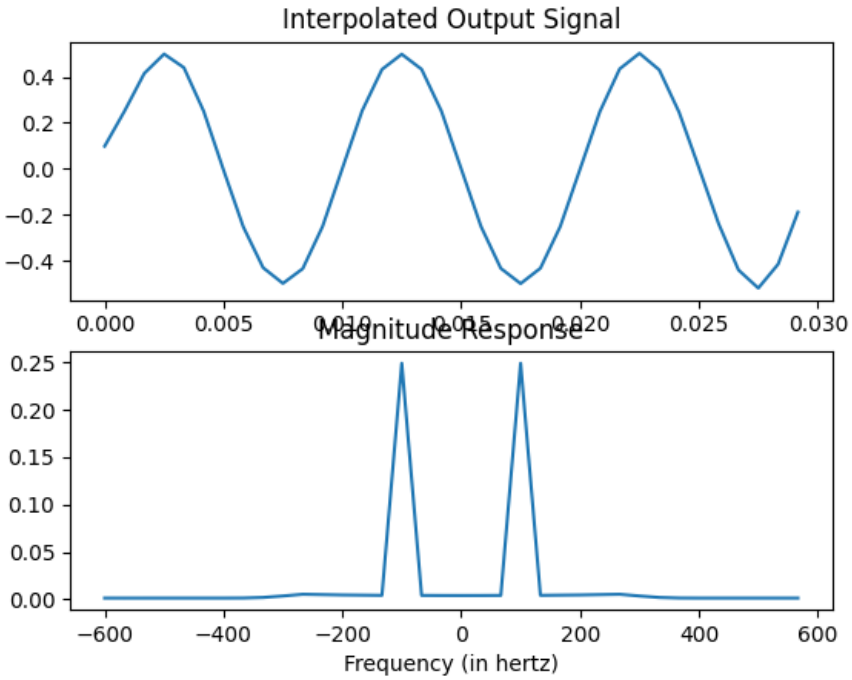


Figure 9: Interpolated Output Signal and Frequency Response

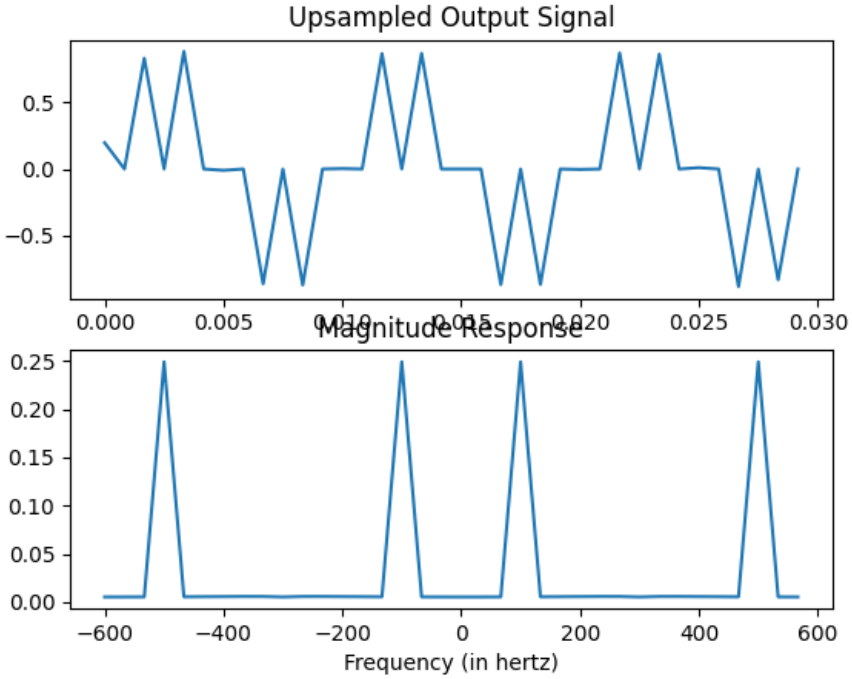


Figure 10: Upsampled Input Signal and Frequency Response