Homework 9

Course Title: Digital Signal Processing I (Spring 2020)

Course Number: ECE53800

Instructor: Dr. Li Tan

Author: **Zhankun Luo**

Problems

11.1, 11.4, 11.7, 11.8, 11.11, 11.13, 11.16, 11.17, 11.20, 11.21

MATLAB

11.24, 11.25, 11.26

Problems

Problem 11.1

For a single-stage decimator with the following specifications:

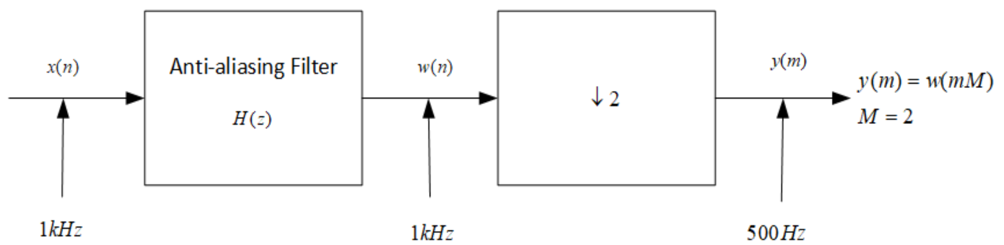
- Original sampling rate=1 kHz
- Decimation factor M=2
- Frequency of interest=0–100 Hz
- Passband ripple=0.015 dB
- Stopband attenuation=40 dB,

(a) Draw the block diagram for the decimator;

(b) Determine the window type, filter length, and cutoff frequency if the window method is used for the anti-aliasing FIR filter design

solution

(a) Draw the block diagram for the decimator;



(b) Determine the window type, filter length, and cutoff frequency

TABLE 7.7 FIR filter length estimation using window functions (Normalized transition width $\Delta f = |f_{stop} - f_{pass}| / f_s$).

Window type	Window function $w(n)$, $-M \le n \le M$	Window length N	Passband ripple (dB)	Stopband attenuation (dB)
Rectangular	1	$N = 0.9/\Delta f$	0.7416	21
Hanning	$0.5 + 0.5\cos\left(\frac{\pi n}{M}\right)$	$N = 3.1/\Delta f$	0.0546	44
Hamming	$0.54 + 0.46\cos\left(\frac{\pi n}{M}\right)$	$N = 3.3/\Delta f$	0.0194	53
Blackman	$0.42 + 0.5\cos\left(\frac{n\pi}{M}\right) + 0.08\cos\left(\frac{2n\pi}{M}\right)$	$N = 5.5/\Delta f$	0.0017	74

From table, Passband ripple<0.015 dB Stopband attenuation>40 dB,

window type: **Blackman**

filter length,

$$f_{pass} = 100Hz, f_{stop} = \frac{f_s/M}{2} = 250Hz$$

$$\Delta f = \frac{(f_{stop} - f_{pass})}{f_s} = 150/1000 = 0.15$$

$$N = \frac{5.5}{\Delta f} = 36.67$$

select the closest odd number $N = 37$

cutoff frequency

$$f_c = \frac{f_{pass} + f_{stop}}{2} = 175Hz$$

Problem 11.4

For a single-stage interpolator with the following specifications:

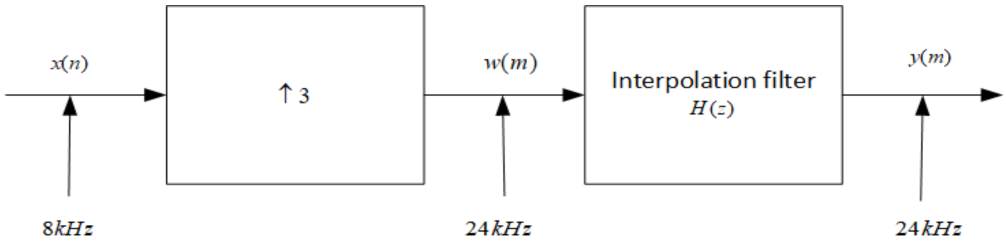
- Original sampling rate=8 kHz.
- Interpolation factor L=3
- Frequency of interest=0–3400 Hz
- Passband ripple=0.02 dB
- Stopband attenuation=46 dB,

(a) Draw the block diagram for the interpolator;

(b) Determine the window type, filter length, and cutoff frequency if the window method is used for the anti-image FIR filter design.

solution

(a) Draw the block diagram for the interpolator;



(b) Determine the window type, filter length, and cutoff frequency

From table, Passband ripple<0.02 dB Stopband attenuation>46 dB,

window type: **Hamming**

filter length,

$$f_{pass} = 3.4kHz, f_{stop} = \frac{f_s}{2} = 4kHz$$
$$\Delta f = \frac{(f_{stop} - f_{pass})}{f_s \times L} = 0.6/24 = 0.025$$
$$N = \frac{3.3}{\Delta f} = 132$$

select the closest odd number $N = 133$

cutoff frequency

$$f_c = \frac{f_{pass} + f_{stop}}{2} = 3.7kHz$$

Problem 11.7

For the sampling conversion from 6 to 8 kHz with the following specifications:

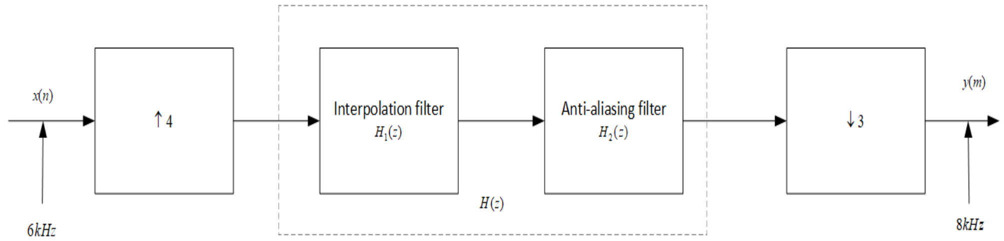
- Original sampling rate=6 kHz
- Interpolation factor L=4
- Decimation factor M=3
- Frequency of interest=0–2400 Hz
- Passband ripple=0.02 dB
- Stopband attenuation=46 dB,

(a) Draw the block diagram for the processor;

(b) Determine the window type, filter length, and cutoff frequency if the window method is used for the combined FIR filter H(z).

solution

(a) Draw the block diagram for the processor;



(b) Determine the window type, filter length, and cutoff frequency

For interpolation filter:

$$f_{stop} = \frac{f_s}{2} = 3kHz$$

For Anti-aliasing filter:

$$f_{stop} = \frac{(f_s \times L)/M}{2} = 4kHz$$

Because 3 kHz < 4 kHz, we choose $f_{stop} = \min(3, 4) = 3$ kHz

From table, Passband ripple<0.02 dB Stopband attenuation>46 dB,

window type: **Hamming**

$$f_{pass} = 2.4kHz, f_{stop} = 3kHz$$
$$\Delta f = \frac{f_{stop} - f_{pass}}{f_s \times L} = 0.6/24 = 0.025$$
$$N = \frac{3.3}{\Delta f} = 132$$

select the closest odd number $N = 133$

cutoff frequency

$$f_c = \frac{f_{pass} + f_{stop}}{2} = 2.7kHz$$

Problem 11.8

For the design of a two-stage decimator with the following specifications:

- Original sampling rate=320 kHz
- Frequency of interest=0–3400 Hz
- Passband ripple=0.05 (absolute)
- Stopband attenuation=0.005 (absolute)
- Final sampling rate=8000 Hz

(a) Draw the decimation block diagram;

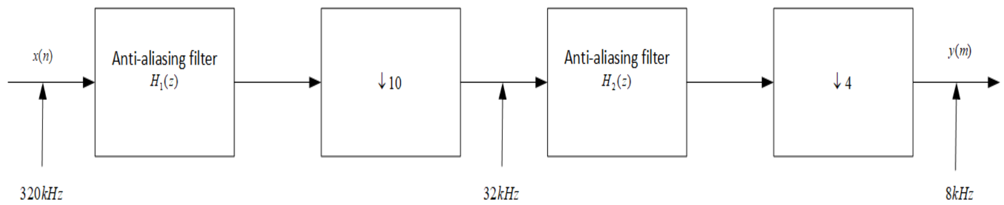
(b) Specify the sampling rate for each stage;

(c) determine the window type, filter length, and cutoff frequency for the first stage if the window method is used for anti-aliasing FIR filter design (H1(z));

(d) determine the window type, filter length, and cutoff frequency for the second stage if the window method is used for anti-aliasing FIR filter design (H2(z)).

solution

(a) Draw the decimation block diagram;



(b) Specify the sampling rate for each stage;

$$\frac{320kHz}{8kHz} = 40 = 10 \times 4 = M_1 \times M_2$$

Here we select the sampling rate $M_1 = 10$ for stage 1

the sampling rate $M_2 = 4$ for stage 2.

(c) determine the window type, filter length, and cutoff frequency for the first stage H1(z);

$$20\log_{10}(1/0.005) = 46.02dB$$

From table, Passband ripple<0.05 dB Stopband attenuation>46.02 dB,

window type: **Hamming**

filter length,

$$f_{pass} = 3.4kHz, f_{stop} = \frac{f_s/M_1}{2} = 16kHz$$

$$\Delta f = \frac{(f_{stop} - f_{pass})}{f_s} = 12.6/320 = 0.039375$$

$$N = \frac{3.3}{\Delta f} = 83.81$$

select the closest odd number $N = 85$

cutoff frequency

$$f_c = \frac{f_{pass} + f_{stop}}{2} = 9.7kHz$$

(d) determine the window type, filter length, and cutoff frequency for the second stage H2(z)

From table, Passband ripple<0.05 dB Stopband attenuation>46.02 dB,

window type: **Hamming**

filter length,

$$f_{pass} = 3.4kHz, f_{stop} = \frac{f_s/(M_1 M_2)}{2} = 4kHz$$
$$\Delta f = \frac{(f_{stop} - f_{pass})}{f_s/M_1} = 0.6/32 = 0.01875$$
$$N = \frac{3.3}{\Delta f} = 176$$

select the closest odd number $N = 177$

cutoff frequency

$$f_c = \frac{f_{pass} + f_{stop}}{2} = 3.7kHz$$

Problem 11.11

(a) Given an interpolator filter as

$$H(z) = 0.25 + 0.4z^{-1} + 0.5z^{-2} + 0.6z^{-3} + 0.7z^{-4} + 0.6z^{-5}$$

draw the block diagram for interpolation polyphase filter implementation for the case of L = 4.

(b) Given a decimation filter as

$$H(z) = 0.25 + 0.4z^{-1} + 0.5z^{-2} + 0.6z^{-3} + 0.5z^{-3} + 0.4z^{-4}$$

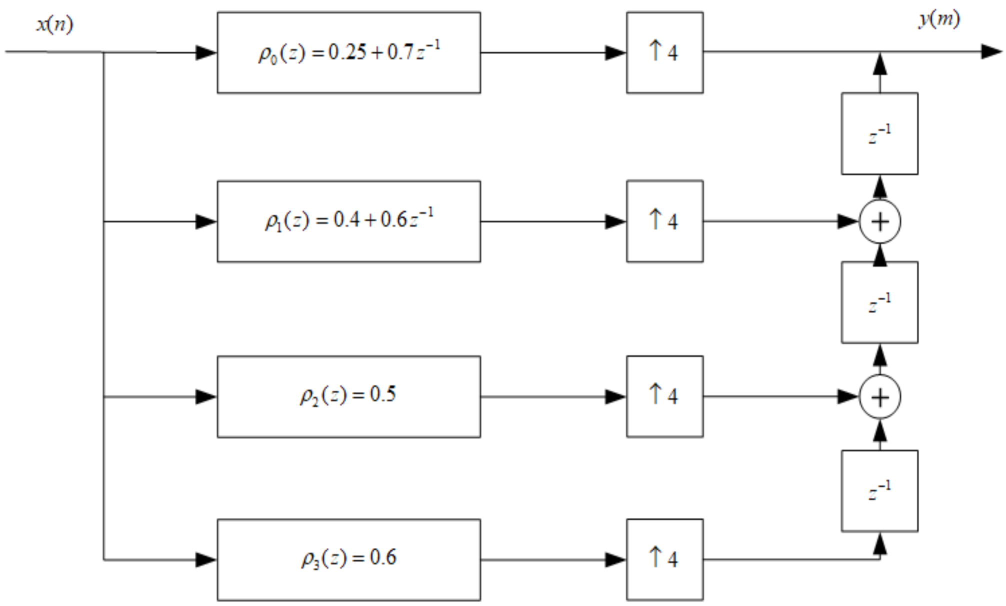
It should be

$$H(z) = 0.25 + 0.4z^{-1} + 0.5z^{-2} + 0.6z^{-3} + 0.5z^{-4} + 0.4z^{-5}$$

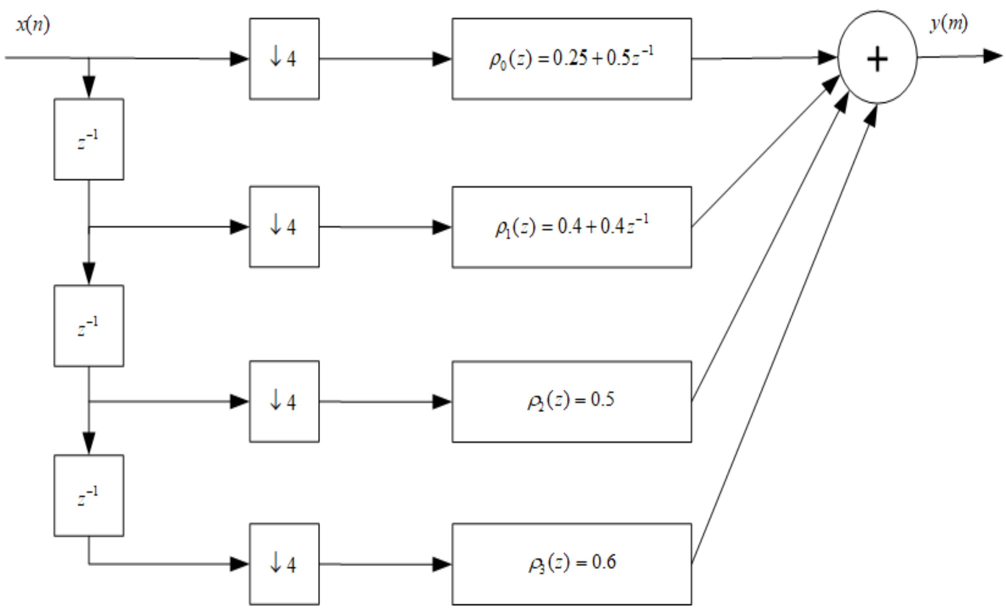
draw the block diagram for decimation polyphase filter implementation for the case of M=4.

solution

(a) the block diagram for the interpolation polyphase filter



(b) the block diagram for decimation polyphase filter



Problem 11.13

Given a speech system with the following specifications:

- Speech input frequency range: 0–4 kHz.
- ADC resolution=16 bits.
- Current sampling rate=8 kHz,

(a) Determine the oversampling rate if a 12-bit ADC chip is used to replace the speech system;

(b) Draw the block diagram.

solution

(a) Determine the oversampling rate;

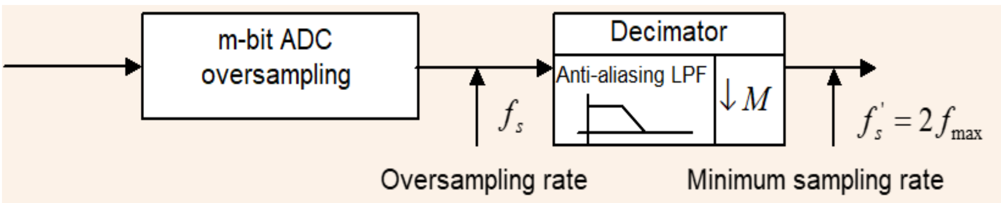
Noise power to be the same, n=16, m=12, f_max = 4 kHz

$$\frac{A^2}{12} \times 2^{-2n} = \frac{2f_{\max}}{f_s}\frac{A^2}{12} \times 2^{-2m}$$

Thus, the oversampling rate f_s is 2.048 MHz

$$f_s = (2f_{\max}) \times 2^{2(n-m)} = 8 \times 2^8 = 2048kHz = 2.048MHz$$

(b) Draw the block diagram.



where $f_s =, f_s' = 2f_{\max} = 8kHz$

Problem 11.16

Given an audio system with the following specifications:

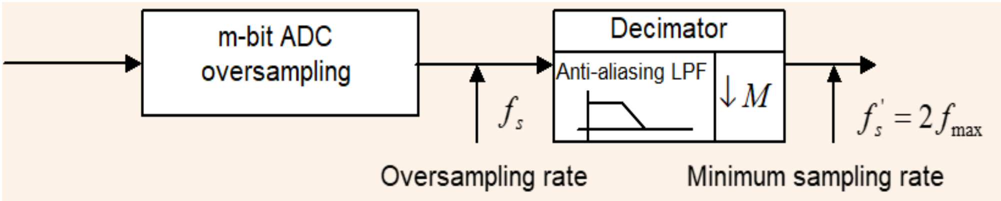
- Audio input frequency range: 0–15 kHz.
- ADC resolution=6 bits.
- Oversampling rate=45MHz,

(a) Draw the block diagram;

(b) Determine the actual effective ADC resolution (number of bits per sample)

solution

(a) Draw the block diagram;



where $f_s = 45MHz, f_s' = 2f_{\max} = 30kHz$

(b) Determine the actual effective ADC resolution

Noise power to be the same

$$\frac{A^2}{12} \times 2^{-2n} = \frac{2f_{\max}}{f_s}\frac{A^2}{12} \times 2^{-2m}$$

Thus, the actual effective ADC resolution

$$n = 0.5\log_2(\frac{f_s}{2f_{\max}}) + m = 0.5\log_2(\frac{45000}{30}) + 6 = 11.275 \approx 11$$

Problem 11.17

Given the following specifications of an oversampling DSP system:

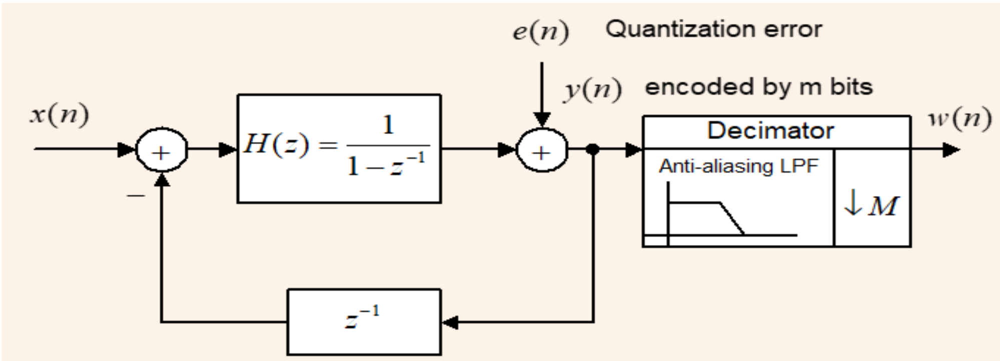
- Audio input frequency range: 0–4 kHz
- First-order SDM with a sampling rate of 128 kHz
- ADC resolution in SDM=1 bit,

(a) draw the block diagram using the DSP model;

(b) determine the equivalent (effective) ADC resolution

solution

(a) draw the block diagram using the DSP model;



Where $M = \frac{f_s}{2f_{\max}} = 16$

(b) determine the equivalent (effective) ADC resolution

Noise power to be the same

$$\frac{A^2}{12} \times 2^{-2n} = (\frac{2f_{\max}}{f_s})^3 \frac{\pi^2}{3} \times \frac{A^2}{12} \times 2^{-2m}$$

Thus, the actual effective ADC resolution, here $m = 1, f_{\max} = 4kHz, f_s = 128kHz$

$$n = m + 0.5\log_2(\frac{3}{\pi^2} \times (\frac{f_s}{2f_{\max}})^3)$$

$$= 1 + 0.5 \times 3 \times \log_2(\frac{f_s}{2f_{\max}}) - 0.5\log_2(\frac{\pi^2}{3})$$

$$= 1 + 0.5 \times 3 \times \log_2(\frac{128}{2 \times 4}) - 0.5\log_2(\frac{\pi^2}{3}) \approx 6.14 \approx 6$$

the equivalent (effective) ADC resolution is 6

Problem 11.20

Given the following specifications of an oversampling DSP system:

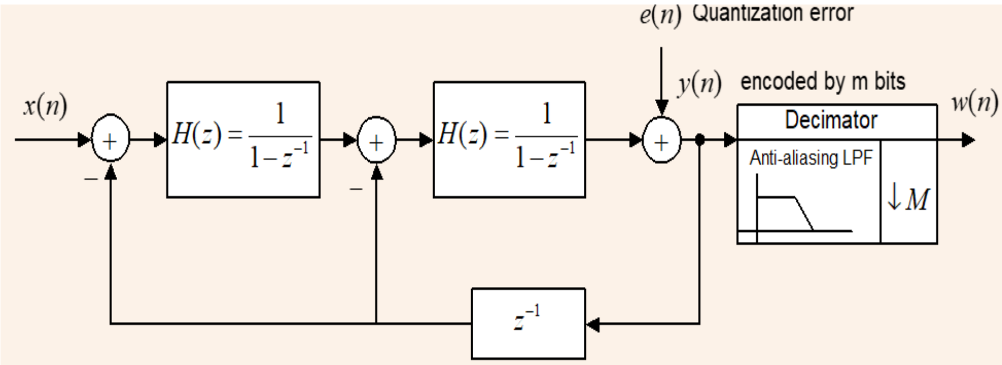
- Signal input frequency range: 0–500 Hz
- Second-order SDM with a sampling rate of 16 kHz
- ADC resolution in SDM=8 bits,

(a) Draw the block diagram using the DSP model;

(b) Determine the equivalent (effective) ADC resolution

solution

(a) Draw the block diagram using the DSP model;



Where $M = \frac{f_s}{2f_{\max}} = 16$

(b) Determine the equivalent (effective) ADC resolution

$$\frac{A^2}{12} \times 2^{-2n} = (\frac{2f_{\max}}{f_s})^{2K+1} \frac{\pi^{2K+1}}{\pi(2K+1)} \times \frac{A^2}{12} \times 2^{-2m}$$

Thus

$$n = m + 0.5 \times (2K+1)\log_2(\frac{f_s}{2f_{\max}}) - 0.5\log_2(\frac{\pi^{2K}}{2k+1})$$

Here, K =2, m= 8, $f_{\max} = 0.5kHz$, $f_s = 16kHz$, so

$$n \approx 8 + 2.5\log_2(16) - 2.142 = 15.858 \approx 16$$

the equivalent (effective) ADC resolution is **16**

Problem 11.21

Given a bandpass signal with its spectrum shown in Fig. 11.46,

and assuming the bandwidth B=5 kHz, select the sampling rate, and sketch the sampled spectrum ranging from 0 Hz to the carrier frequency for each of the following carrier frequencies:

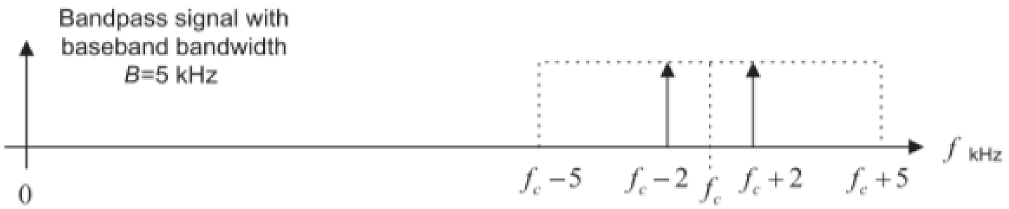


FIG. 11.46

Spectrum of the bandpass signal in Problem 11.21.

(a) fc=30 kHz

(b) fc=25 kHz

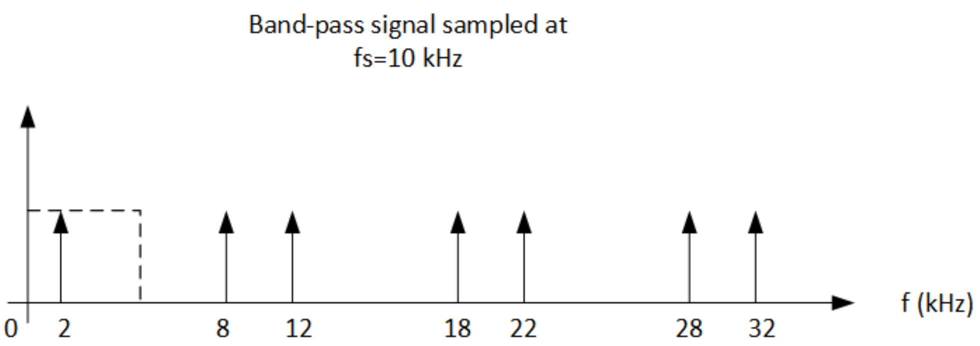
(c) fc=33 kHz

solution

(a) fc=30 kHz, select the sampling rate, sketch the sampled spectrum 0 Hz - the carrier frequency

$f_c/B = 6$ is an even number

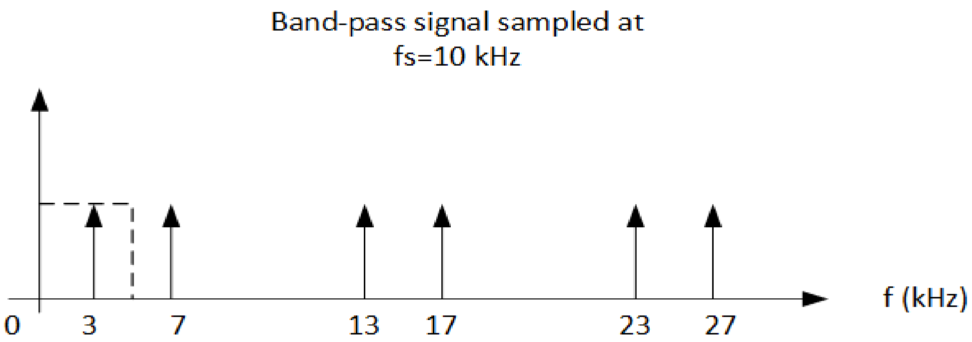
$f_s = 2B = 10$ kHz



(b) fc=25 kHz, select the sampling rate, sketch the sampled spectrum 0 Hz - the carrier frequency

$f_c/B = 5$ is an odd number

$f_s = 2B = 10$ kHz

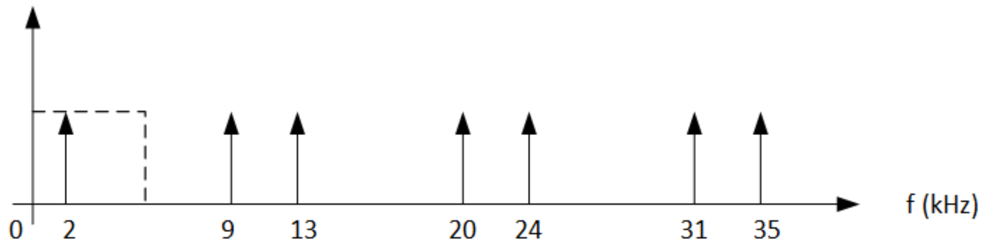


(c) fc=33 kHz, select the sampling rate, sketch the sampled spectrum 0 Hz - the carrier frequency

$f_c/B = 6.6$ is not a integer, $\overline{B} = \frac{f_c}{6} = 5.5$ kHz

$f_s = 2\overline{B} = 11$ kHz

Band-pass signal sampled at
fs=11 kHz



MATLAB

Problem 11.24

Generate a sinusoid with a 1000 Hz for 0.05 s using a sampling rate of 8 kHz,

(a) Design a decimator to change the sampling rate to 4 kHz with specifications below:

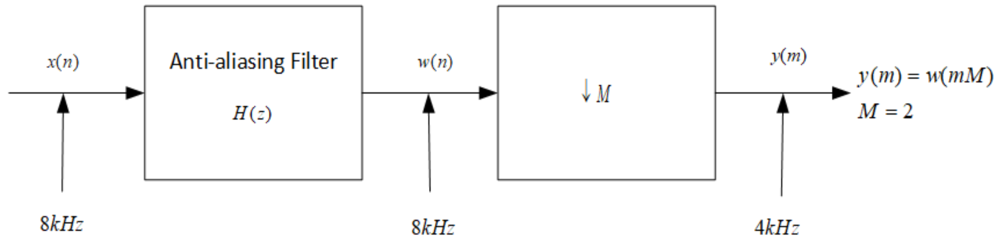
- Signal frequency range: 0–1800 Hz.
- Hamming window required for FIR filter design

(b) Write a MATLAB program to implement the down-sampling scheme,

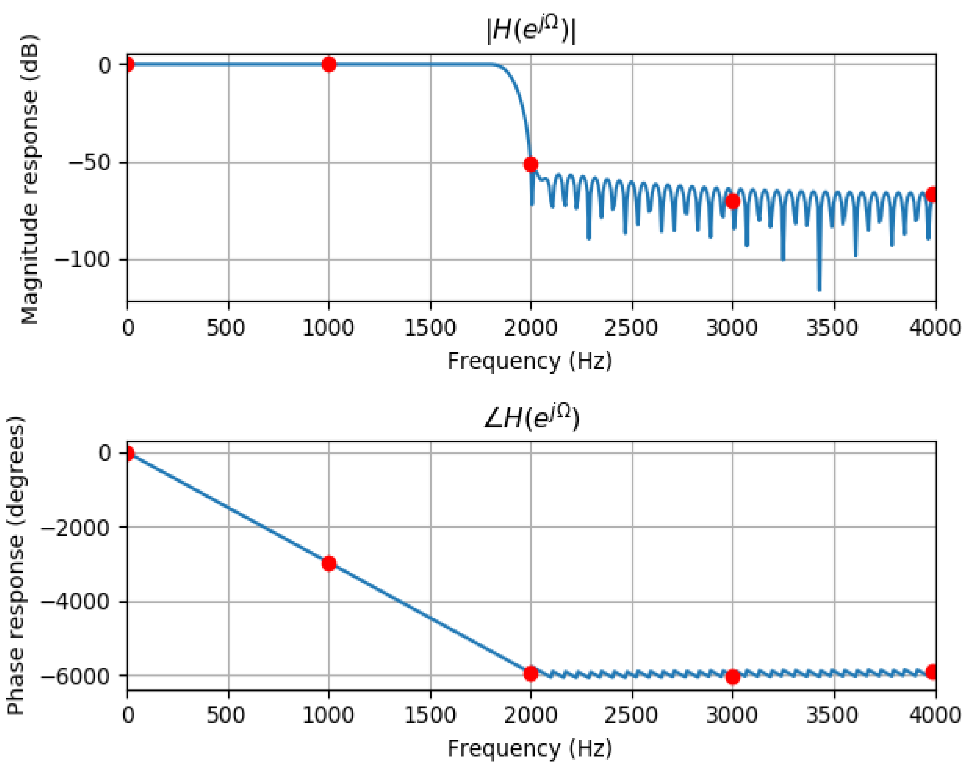
and plot the original signal and the down-sampled signal versus the sample number, respectively.

solution

(a) Design a decimator to change the sampling rate to 4 kHz



The anti-aliasing filter, filter length = **133**, $f_c = 1900$ Hz

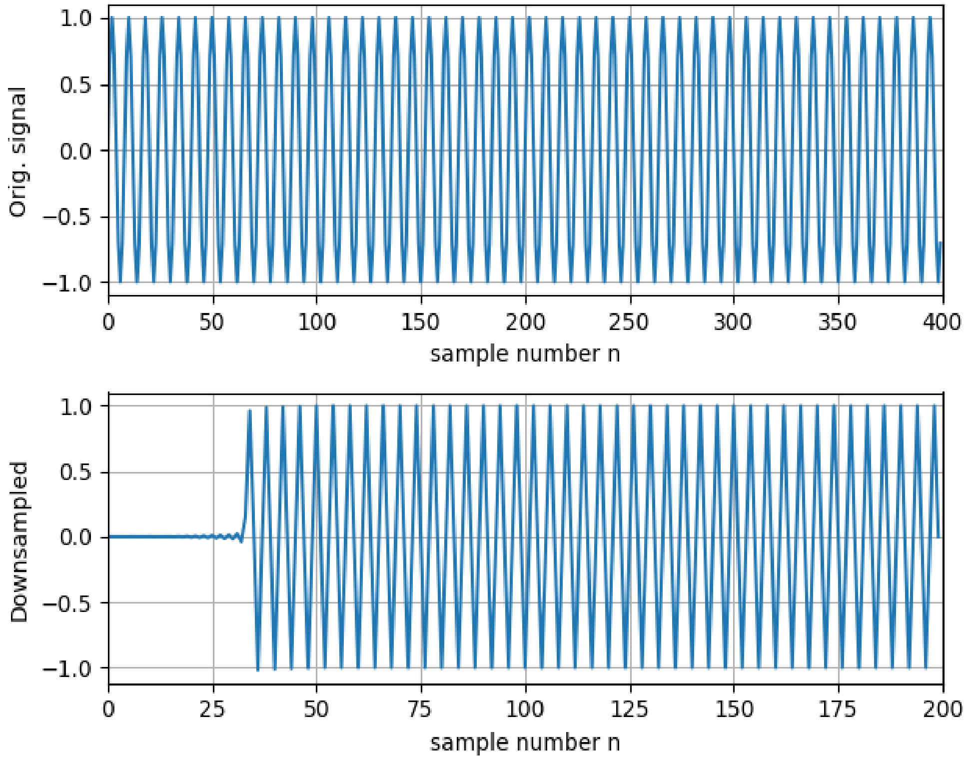


```
h(n) = \
[-0.0043, 0.0019, 0.0047, -0.0012, -0.0051, 0.0004, 0.0053, 0.0004, -0.0054,
-0.0013, 0.0054, 0.0022, -0.0053, -0.0031, 0.005, 0.0041, -0.0045, -0.0049,
0.0039, 0.0058, -0.0031, -0.0065, 0.0022, 0.0072, -0.0012, -0.0077, 0.0, 0.0081,
0.0013, -0.0084, -0.0027, 0.0084, 0.0043, -0.0082, -0.0058, 0.0078, 0.0075,
-0.0071, -0.0092, 0.0062, 0.0109, -0.0049, -0.0126, 0.0032, 0.0143,
-0.0012, -0.0159, -0.0013, 0.0175, 0.0044, -0.0189, -0.0081, 0.0203, 0.0128,
-0.0215, -0.0188, 0.0225, 0.0269, -0.0234, -0.0388, 0.0241, 0.0588, -0.0246,
-0.1032, 0.0249, 0.3173, 0.475, 0.3173, 0.0249, -0.1032, -0.0246, 0.0588,
0.0241, -0.0388, -0.0234, 0.0269, 0.0225, -0.0188, -0.0215, 0.0128, 0.0203,
-0.0081, -0.0189, 0.0044, 0.0175, -0.0013, -0.0159, -0.0012, 0.0143, 0.0032,
-0.0126, -0.0049, 0.0109, 0.0062, -0.0092, -0.0071, 0.0075, 0.0078, -0.0058,
-0.0082, 0.0043, 0.0084, -0.0027, -0.0084, 0.0013, 0.0081, 0.0, -0.0077,
-0.0012, 0.0072, 0.0022, -0.0065, -0.0031, 0.0058, 0.0039, -0.0049, -0.0045,
0.0041, 0.005, -0.0031, -0.0053, 0.0022, 0.0054, -0.0013, -0.0054, 0.0004,
0.0053, 0.0004, -0.0051, -0.0012, 0.0047, 0.0019, -0.0043]

h_w(n) = \
[-0.0003, 0.0002, 0.0004, -0.0001, -0.0004, 0.0, 0.0005, 0.0, -0.0006, -0.0002,
0.0007, 0.0003, -0.0008, -0.0005, 0.0009, 0.0008, -0.0009, -0.0011, 0.0009,
0.0015, -0.0009, -0.0019, 0.0007, 0.0024, -0.0004, -0.0029, 0.0, 0.0033, 0.0006,
-0.0038, -0.0013, 0.0042, 0.0022, -0.0044, -0.0033, 0.0046, 0.0045, -0.0045,
-0.006, 0.0041, 0.0075, -0.0035, -0.0092, 0.0024, 0.011, -0.0009, -0.0128,
-0.0011, 0.0147, 0.0037, -0.0165, -0.0072, 0.0183, 0.0117, -0.0199, -0.0176,
0.0214, 0.0258, -0.0226, -0.0378, 0.0236, 0.0581, -0.0244, -0.1027, 0.0248,
0.3172, 0.475, 0.3172, 0.0248, -0.1027, -0.0244, 0.0581, 0.0236, -0.0378,
-0.0226, 0.0258, 0.0214, -0.0176, -0.0199, 0.0117, 0.0183, -0.0072, -0.0165,
0.0037, 0.0147, -0.0011, -0.0128, -0.0009, 0.011, 0.0024, -0.0092, -0.0035,
0.0075, 0.0041, -0.006, -0.0045, 0.0045, 0.0046, -0.0033, -0.0044, 0.0022,
0.0042, -0.0013, -0.0038, 0.0006, 0.0033, 0.0, -0.0029, -0.0004, 0.0024, 0.0007,
-0.0019, -0.0009, 0.0015, 0.0009, -0.0011, -0.0009, 0.0008, 0.0009, -0.0005,
-0.0008, 0.0003, 0.0007, -0.0002, -0.0006, 0.0, 0.0005, 0.0, -0.0004, -0.0001,
0.0004, 0.0002, -0.0003]
```

(b) Write a MATLAB program to implement the down-sampling scheme,

and plot the original signal and the down-sampled signal versus the sample number, respectively.



Python script:

```
from fir_filter.choose_window_type import choose_window_type
from fir_filter.calc_window_len import calc_window_len
from fir_filter.calc_mag_angle import calc_mag_angle
from iir_filter.calc_mag_angle import plot_mag_angle_freq
from fir_filter.calc_freq_cutoff import calc_freq_cutoff
from fir_filter.fir_filter import print_approx, fir_filter
from fir_filter.window import window
from fir_filter.filter import filter


def plot_down_sample(list_origin, list_downsample, f_sample, M, interval,
path_fig="./test.png"):
    fig = plt.figure()
    plt.subplot(2, 1, 1)
    num_origin = ceil(interval * f_sample)
    plt.plot(list(range(num_origin)), list_origin[:num_origin])
    plt.xlim([0, interval * f_sample])
    plt.xlabel("sample number n")
    plt.ylabel("Orig. signal")
    plt.grid()
    plt.subplot(2, 1, 2)
    num_downsample = ceil(interval * f_sample/M)
    plt.plot(list(range(num_downsample)), list_downsample[:num_downsample])
    plt.xlim([0, interval * f_sample/M])
    plt.xlabel("sample number n")
    plt.ylabel("Downsampled")
    plt.grid()
    plt.tight_layout()
    fig.savefig(path_fig)
    plt.show()
```

```
# passband_ripple = 0.02
# stopband_attenuation = 60
# str_window_type = choose_window_type(passband_ripple, stopband_attenuation)
# print(str_window_type)
str_window_type = "Hamming"
f_s, M = 8000, 2
f_pass, f_stop = 1800, f_s /(2*M)
list_transient_band = [ [f_pass, f_stop] ]
filter_len = calc_window_len(str_window_type, list_transient_band, f_sample=f_s)
print(filter_len)
list_freq_cutoff = calc_freq_cutoff(list_transient_band)
print(list_freq_cutoff)
list_filter = fir_filter(list_freq_cutoff, f_s, filter_len,
str_filter_type="low_pass")
print_approx(list_filter)
# Hamming window function.
path_fig = "../p11_24_H(z).png"
list_filter_window = window(list_filter, str_window_type=str_window_type)
print_approx(list_filter_window)
list_mag, list_angle, list_omega = calc_mag_angle(list_filter_window)
plot_mag_angle_freq(list_mag, list_angle, list_omega, f_s, path_fig=path_fig)
# down sample
from math import sin, pi, ceil
import matplotlib.pyplot as plt
interval = 0.05
list_x = [sin(2*pi * 1000*ind / f_s) for ind in range(round(0.05*f_s))]
list_anti = filter(list_x, list_filter_window) # anti-aliasing filter
list_downsample = [elem for ind, elem in enumerate(list_anti) if ind % M == 0] #
down sample
plot_down_sample(list_x, list_downsample, f_s, M, interval=0.05,
path_fig="../p11_24_point.png")
```

Problem 11.25

Generate a sinusoid with a 1000 Hz for 0.05 s using a sampling rate of 8 kHz,

(a) Design an interpolator to change the sampling rate to 16 kHz with following specifications:

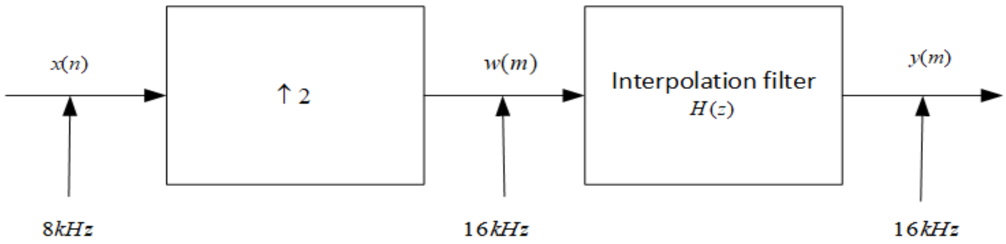
- Signal frequency range: 0–3600 Hz
- Hamming window required for FIR filter design

(b) Write a MATLAB program to implement the up-sampling scheme,

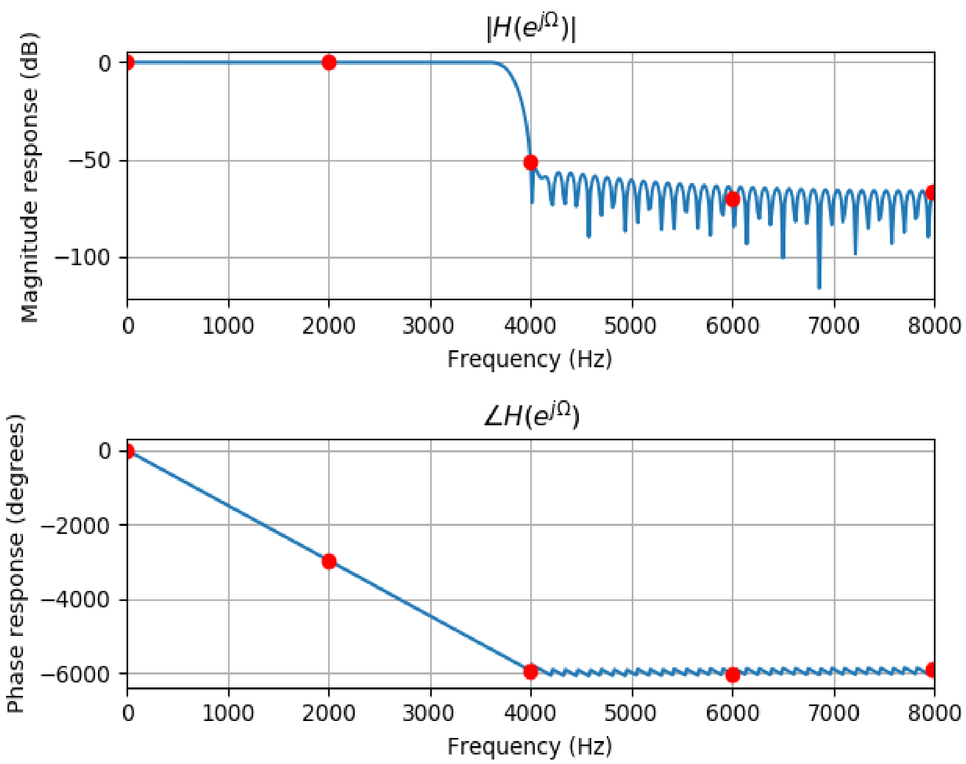
and plot the original signal and the up-sampled signal versus the sample number, respectively.

solution

(a) Design an interpolator to change the sampling rate to 16 kHz



The interpolation filter, filter length = **133**, $f_c = 3800$ Hz



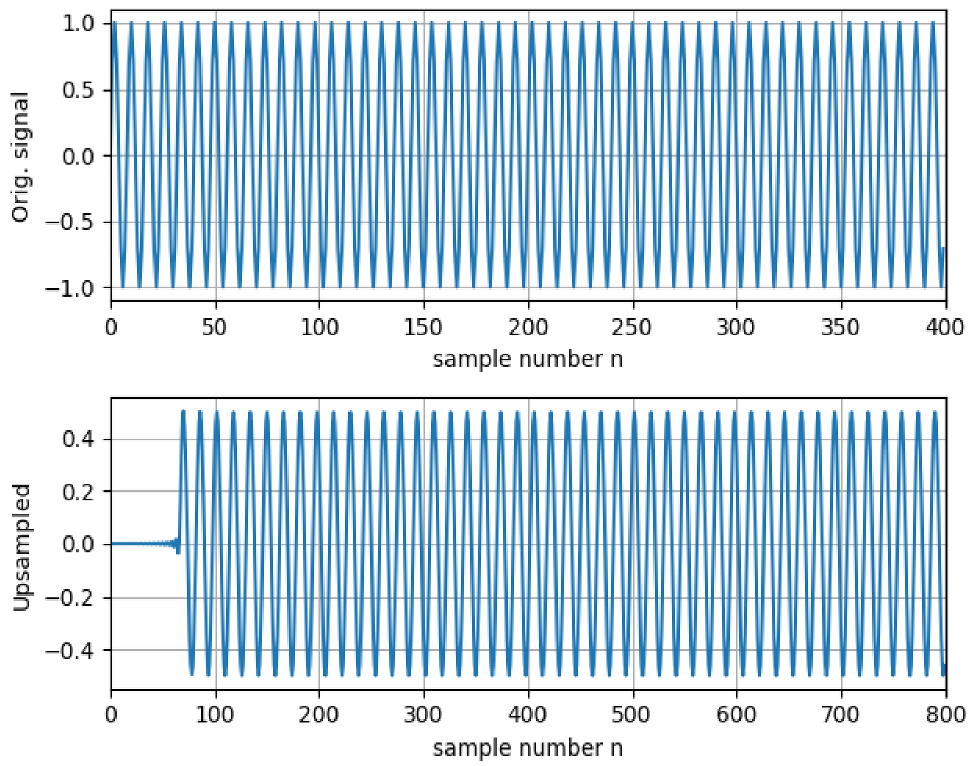
```
h(n) =\
[-0.0043, 0.0019, 0.0047, -0.0012, -0.0051, 0.0004, 0.0053, 0.0004, -0.0054,
-0.0013, 0.0054, 0.0022, -0.0053, -0.0031, 0.005, 0.0041, -0.0045, -0.0049,
0.0039, 0.0058, -0.0031, -0.0065, 0.0022, 0.0072, -0.0012, -0.0077, 0.0, 0.0081,
0.0013, -0.0084, -0.0027, 0.0084, 0.0043, -0.0082, -0.0058, 0.0078, 0.0075,
-0.0071, -0.0092, 0.0062, 0.0109, -0.0049, -0.0126, 0.0032, 0.0143,
-0.0012, -0.0159, -0.0013, 0.0175, 0.0044, -0.0189, -0.0081, 0.0203, 0.0128,
-0.0215, -0.0188, 0.0225, 0.0269, -0.0234, -0.0388, 0.0241, 0.0588, -0.0246,
-0.1032, 0.0249, 0.3173, 0.475, 0.3173, 0.0249, -0.1032, -0.0246, 0.0588,
0.0241, -0.0388, -0.0234, 0.0269, 0.0225, -0.0188, -0.0215, 0.0128, 0.0203,
-0.0081, -0.0189, 0.0044, 0.0175, -0.0013, -0.0159, -0.0012, 0.0143, 0.0032,
-0.0126, -0.0049, 0.0109, 0.0062, -0.0092, -0.0071, 0.0075, 0.0078, -0.0058,
-0.0082, 0.0043, 0.0084, -0.0027, -0.0084, 0.0013, 0.0081, 0.0, -0.0077,
-0.0012, 0.0072, 0.0022, -0.0065, -0.0031, 0.0058, 0.0039, -0.0049, -0.0045,
0.0041, 0.005, -0.0031, -0.0053, 0.0022, 0.0054, -0.0013, -0.0054, 0.0004,
0.0053, 0.0004, -0.0051, -0.0012, 0.0047, 0.0019, -0.0043]

h_w(n) = \
[-0.0003, 0.0002, 0.0004, -0.0001, -0.0004, 0.0, 0.0005, 0.0, -0.0006, -0.0002,
0.0007, 0.0003, -0.0008, -0.0005, 0.0009, 0.0008, -0.0009, -0.0011, 0.0009,
0.0015, -0.0009, -0.0019, 0.0007, 0.0024, -0.0004, -0.0029, 0.0, 0.0033, 0.0006,
-0.0038, -0.0013, 0.0042, 0.0022, -0.0044, -0.0033, 0.0046, 0.0045, -0.0045,
-0.006, 0.0041, 0.0075, -0.0035, -0.0092, 0.0024, 0.011, -0.0009, -0.0128,
-0.0011, 0.0147, 0.0037, -0.0165, -0.0072, 0.0183, 0.0117, -0.0199, -0.0176,
0.0214, 0.0258, -0.0226, -0.0378, 0.0236, 0.0581, -0.0244, -0.1027, 0.0248,
0.3172, 0.475, 0.3172, 0.0248, -0.1027, -0.0244, 0.0581, 0.0236, -0.0378,
-0.0226, 0.0258, 0.0214, -0.0176, -0.0199, 0.0117, 0.0183, -0.0072, -0.0165,
0.0037, 0.0147, -0.0011, -0.0128, -0.0009, 0.011, 0.0024, -0.0092, -0.0035,
0.0075, 0.0041, -0.006, -0.0045, 0.0045, 0.0046, -0.0033, -0.0044, 0.0022,
0.0042, -0.0013, -0.0038, 0.0006, 0.0033, 0.0, -0.0029, -0.0004, 0.0024, 0.0007,
-0.0019, -0.0009, 0.0015, 0.0009, -0.0011, -0.0009, 0.0008, 0.0009, -0.0005,
-0.0008, 0.0003, 0.0007, -0.0002, -0.0006, 0.0, 0.0005, 0.0, -0.0004, -0.0001,
0.0004, 0.0002, -0.0003]
```

(b) Write a MATLAB program to implement the up-sampling scheme,

and plot the original signal and the up-sampled signal versus the sample number, respectively.

Up-sampled signal = $\frac{1}{L}$ **original signal**, (L=2)




```
from fir_filter.choose_window_type import choose_window_type
from fir_filter.calc_window_len import calc_window_len
from fir_filter.calc_mag_angle import calc_mag_angle
from iir_filter.calc_mag_angle import plot_mag_angle_freq
from fir_filter.calc_freq_cutoff import calc_freq_cutoff
from fir_filter.fir_filter import print_approx, fir_filter
from fir_filter.window import window
from fir_filter.filter import filter


def plot_up_sample(list_origin, list_upample, f_sample, L, interval,
path_fig="./test.png"):
    fig = plt.figure()
    plt.subplot(2, 1, 1)
    num_origin = ceil(interval * f_sample)
    plt.plot(list(range(num_origin)), list_origin[:num_origin])
    plt.xlim([0, interval * f_sample])
    plt.xlabel("sample number n")
    plt.ylabel("Orig. signal")
    plt.grid()
    plt.subplot(2, 1, 2)
    num_upsample = ceil(interval * f_sample * L)
    plt.plot(list(range(num_upsample)), list_upsample[:num_upsample])
    plt.xlim([0, interval * f_sample * L])
    plt.xlabel("sample number n")
    plt.ylabel("Upsampled")
    plt.grid()
    plt.tight_layout()
    fig.savefig(path_fig)
    plt.show()

# passband_ripple = 0.02
# stopband_attenuation = 60
# str_window_type = choose_window_type(passband_ripple, stopband_attenuation)
```

```python
# print(str_window_type)
str_window_type = "Hamming"
f_s, L = 8000, 2
f_sL = f_s * L
f_pass, f_stop = 3600, f_s /2
list_transient_band = [ [f_pass, f_stop] ]
filter_len = calc_window_len(str_window_type, list_transient_band,
f_sample=f_sL)
print(filter_len)
list_freq_cutoff = calc_freq_cutoff(list_transient_band)
print(list_freq_cutoff)
list_filter = fir_filter(list_freq_cutoff, f_sL, filter_len,
str_filter_type="low_pass")
print_approx(list_filter)
# Hamming window function.
path_fig = "../p11_25_H(z).png"
list_filter_window = window(list_filter, str_window_type=str_window_type)
print_approx(list_filter_window)
list_mag, list_angle, list_omega = calc_mag_angle(list_filter_window)
plot_mag_angle_freq(list_mag, list_angle, list_omega, f_sL, path_fig=path_fig)
# down sample
from math import sin, pi, ceil
import matplotlib.pyplot as plt
import itertools
interval = 0.05
list_x = [sin(2*pi * 1000*ind / f_s) for ind in range(round(0.05*f_s))]
list_zeros = [[elem] + [0] * (L-1) for elem in list_x]
list_zeros = list(itertools.chain.from_iterable(list_zeros))
list_upsample = filter(list_zeros, list_filter_window) # anti-aliasing filter
plot_up_sample(list_x, list_upsample, f_s, L, interval=0.05,
path_fig="../p11_25_point.png")
```

Problem 11.26

Generate a sinusoid with a frequency of 500 Hz for 0.1 s using a sampling rate of 8 kHz,

(a) Design an interpolation and decimation processing algorithm to change the sampling rate to 22 kHz

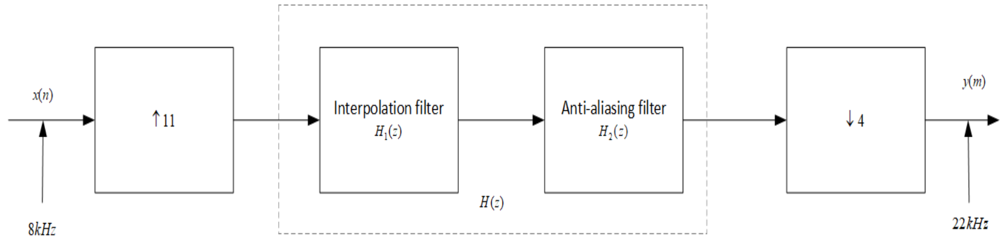
- Signal frequency range: 0–3400 Hz.
- Hamming window required for FIR filter design

(b) Write a MATLAB program to implement the scheme,

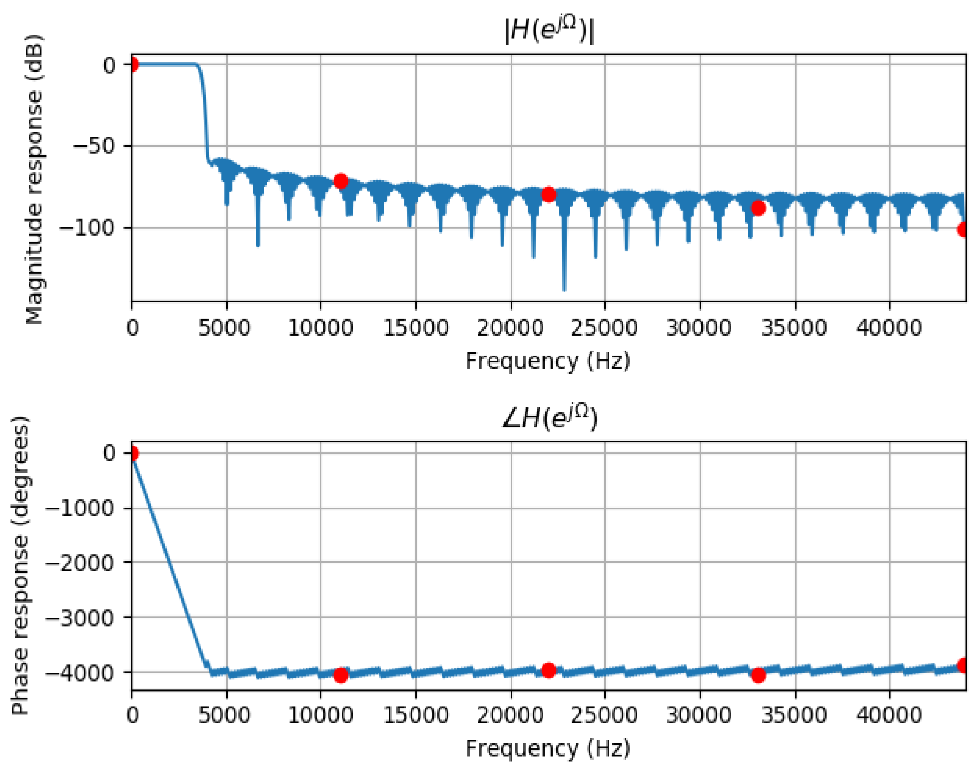
and plot the original signal and the sampled signal at the rate of 22 kHz versus the sample number, respectively.

solution

(a) Design an interpolation and decimation processing algorithm to change the sampling rate to 22 kHz



The combined filter, filter length = **485**, $f_c = 3700$ Hz



```
h(n) = \
[0.0012, 0.001, 0.0007, 0.0004, 0.0001, -0.0003, -0.0006, -0.0009, -0.0012,
-0.0013, -0.0014, -0.0013, -0.0012, -0.001, -0.0007, -0.0004, -0.0, 0.0003,
0.0007, 0.001, 0.0012, 0.0014, 0.0014, 0.0014, 0.0013, 0.001, 0.0007, 0.0004,
-0.0, -0.0004, -0.0008, -0.0011, -0.0013, -0.0015, -0.0015, -0.0015, -0.0013,
-0.0011, -0.0007, -0.0003, 0.0001, 0.0005, 0.0009, 0.0012, 0.0014, 0.0016,
0.0016, 0.0015, 0.0014, 0.0011, 0.0007, 0.0003, -0.0001, -0.0006, -0.001,
-0.0013, -0.0015, -0.0017, -0.0017, -0.0016, -0.0014, -0.0011, -0.0007, -0.0003,
0.0002, 0.0006, 0.0011, 0.0014, 0.0017, 0.0018, 0.0018, 0.0017, 0.0015, 0.0012,
0.0007, 0.0003, -0.0002, -0.0007, -0.0012, -0.0016, -0.0018, -0.002, -0.002,
-0.0018, -0.0016, -0.0012, -0.0007, -0.0002, 0.0003, 0.0009,
0.0013, 0.0017, 0.002, 0.0021, 0.0021, 0.002, 0.0017, 0.0013, 0.0007, 0.0002,
-0.0004, -0.001, -0.0015, -0.0019, -0.0022, -0.0023, -0.0023, -0.0021, -0.0018,
-0.0013, -0.0007, -0.0001, 0.0005, 0.0011, 0.0017, 0.0021, 0.0024, 0.0025,
0.0025, 0.0023, 0.0019, 0.0014, 0.0007, 0.0001, -0.0006, -0.0013, -0.0019,
-0.0024, -0.0027, -0.0028, -0.0027, -0.0025, -0.002, -0.0015, -0.0007, 0.0,
0.0008, 0.0015, 0.0022, 0.0027, 0.003, 0.0032, 0.0031, 0.0027, 0.0022, 0.0016,
0.0008, -0.0001, -0.001, -0.0018, -0.0025, -0.0031, -0.0035, -0.0036, -0.0034,
-0.0031, -0.0025, -0.0017, -0.0008, 0.0002, 0.0013, 0.0022, 0.003, 0.0036,
0.004, 0.0041, 0.0039, 0.0035, 0.0028, 0.0018, 0.0008, -0.0004, -0.0016,
-0.0027, -0.0036, -0.0043, -0.0048, -0.0049, -0.0046, -0.0041, -0.0032, -0.0021,
-0.0008, 0.0007, 0.0021, 0.0034, 0.0045, 0.0053, 0.0058, 0.006, 0.0056, 0.0049,
0.0038, 0.0024, 0.0008, -0.001, -0.0028, -0.0044, -0.0059, -0.0069, -0.0075,
-0.0077, -0.0072, -0.0063, -0.0048, -0.0029, -0.0008, 0.0016, 0.004, 0.0063,
0.0082, 0.0097, 0.0106, 0.0108, 0.0102, 0.0089, 0.0068, 0.004, 0.0008, -0.0028,
-0.0066, -0.0102, -0.0134, -0.016, -0.0177, -0.0183, -0.0176, -0.0155, -0.012,
-0.0071, -0.0008, 0.0068, 0.0153, 0.0245, 0.0341, 0.0437, 0.053, 0.0617, 0.0693,
0.0756, 0.0802, 0.0831, 0.0841, 0.0831, 0.0802, 0.0756, 0.0693, 0.0617, 0.053,
0.0437, 0.0341, 0.0245, 0.0153, 0.0068, -0.0008, -0.0071, -0.012, -0.0155,
-0.0176, -0.0183, -0.0177, -0.016, -0.0134, -0.0102, -0.0066, -0.0028, 0.0008,
0.004, 0.0068, 0.0089, 0.0102, 0.0108, 0.0106, 0.0097, 0.0082, 0.0063, 0.004,
0.0016, -0.0008, -0.0029, -0.0048, -0.0063, -0.0072, -0.0077, -0.0075, -0.0069,
-0.0059, -0.0044, -0.0028, -0.001, 0.0008, 0.0024, 0.0038, 0.0049, 0.0056,
0.006, 0.0058, 0.0053, 0.0045, 0.0034, 0.0021, 0.0007, -0.0008, -0.0021,
-0.0032, -0.0041, -0.0046, -0.0049, -0.0048, -0.0043, -0.0036, -0.0027, -0.0016,
-0.0004, 0.0008, 0.0018,
0.0028, 0.0035, 0.0039, 0.0041, 0.004, 0.0036, 0.003, 0.0022, 0.0013, 0.0002,
-0.0008, -0.0017, -0.0025, -0.0031, -0.0034, -0.0036, -0.0035, -0.0031, -0.0025,
-0.0018, -0.001, -0.0001, 0.0008, 0.0016, 0.0022, 0.0027, 0.0031, 0.0032, 0.003,
0.0027, 0.0022, 0.0015, 0.0008, 0.0, -0.0007, -0.0015, -0.002, -0.0025, -0.0027,
-0.0028, -0.0027, -0.0024, -0.0019, -0.0013, -0.0006, 0.0001, 0.0007, 0.0014,
0.0019, 0.0023, 0.0025, 0.0025, 0.0024, 0.0021, 0.0017, 0.0011, 0.0005, -0.0001,
-0.0007, -0.0013, -0.0018, -0.0021, -0.0023, -0.0023, -0.0022, -0.0019, -0.0015,
-0.001, -0.0004, 0.0002, 0.0007, 0.0013, 0.0017, 0.002, 0.0021, 0.0021, 0.002,
0.0017, 0.0013, 0.0009, 0.0003, -0.0002, -0.0007, -0.0012, -0.0016, -0.0018,
-0.002, -0.002, -0.0018, -0.0016, -0.0012, -0.0007, -0.0002, 0.0003, 0.0007,
0.0012, 0.0015, 0.0017, 0.0018, 0.0018, 0.0017, 0.0014, 0.0011, 0.0006, 0.0002,
-0.0003, -0.0007, -0.0011, -0.0014, -0.0016, -0.0017, -0.0017, -0.0015, -0.0013,
-0.001, -0.0006, -0.0001, 0.0003, 0.0007, 0.0011, 0.0014, 0.0015, 0.0016,
0.0016, 0.0014, 0.0012, 0.0009, 0.0005, 0.0001, -0.0003, -0.0007, -0.0011,
-0.0013, -0.0015, -0.0015, -0.0015, -0.0013, -0.0011, -0.0008, -0.0004, -0.0,
0.0004, 0.0007, 0.001, 0.0013, 0.0014, 0.0014, 0.0014, 0.0012, 0.001, 0.0007,
0.0003, -0.0, -0.0004, -0.0007, -0.001, -0.0012, -0.0013, -0.0014, -0.0013,
-0.0012, -0.0009, -0.0006, -0.0003, 0.0001, 0.0004, 0.0007, 0.001, 0.0012]

h_w(n) = \
```

```
[0.0001, 0.0001, 0.0001, 0.0, 0.0, -0.0, -0.0001, -0.0001, -0.0001, -0.0001,
-0.0001, -0.0001, -0.0001, -0.0001, -0.0001, -0.0, -0.0, 0.0, 0.0001, 0.0001,
0.0001, 0.0001, 0.0001, 0.0001, 0.0001, 0.0001, 0.0001, 0.0, -0.0, -0.0,
-0.0001, -0.0001, -0.0002, -0.0002, -0.0002, -0.0002, -0.0002, -0.0001, -0.0001,
-0.0, 0.0, 0.0001, 0.0001, 0.0002, 0.0002, 0.0002, 0.0003, 0.0003, 0.0002,
0.0002, 0.0001, 0.0001, -0.0, -0.0001, -0.0002, -0.0002, -0.0003, -0.0003,
-0.0004, -0.0003, -0.0003, -0.0002, -0.0002,
-0.0001, 0.0, 0.0002, 0.0003, 0.0003, 0.0004, 0.0005, 0.0005, 0.0005, 0.0004,
0.0003, 0.0002, 0.0001, -0.0001, -0.0002, -0.0004, -0.0005, -0.0006, -0.0006,
-0.0006, -0.0006, -0.0005, -0.0004, -0.0003, -0.0001, 0.0001, 0.0003, 0.0005,
0.0006, 0.0007, 0.0008, 0.0008, 0.0008, 0.0007, 0.0005, 0.0003, 0.0001, -0.0002,
-0.0004, -0.0006, -0.0008, -0.001, -0.001, -0.001, -0.001, -0.0008, -0.0006,
-0.0004, -0.0001, 0.0003, 0.0006, 0.0008, 0.0011, 0.0012, 0.0013, 0.0013,
0.0012, 0.001, 0.0007, 0.0004, 0.0, -0.0004, -0.0007, -0.0011, -0.0014, -0.0016,
-0.0017, -0.0016, -0.0015, -0.0012, -0.0009, -0.0005, 0.0, 0.0005, 0.001,
0.0014, 0.0017, 0.002, 0.0021, 0.002, 0.0018, 0.0015, 0.0011, 0.0005, -0.0001,
-0.0007, -0.0013, -0.0018, -0.0022, -0.0025, -0.0026, -0.0025, -0.0023, -0.0018,
-0.0013, -0.0006, 0.0002, 0.001, 0.0017, 0.0023, 0.0028, 0.0031, 0.0033, 0.0031,
0.0028, 0.0022, 0.0015, 0.0006, -0.0003, -0.0013, -0.0022, -0.003, -0.0036,
-0.004, -0.0041, -0.0039, -0.0035, -0.0027, -0.0018, -0.0007, 0.0006,
0.0018, 0.003, 0.004, 0.0047, 0.0052, 0.0053, 0.0051, 0.0044, 0.0035, 0.0022,
0.0007, -0.0009, -0.0026, -0.0041, -0.0054, -0.0064, -0.007, -0.0072, -0.0068,
-0.0059, -0.0046, -0.0028, -0.0007, 0.0015, 0.0038, 0.006, 0.0079, 0.0093,
0.0102, 0.0104, 0.0099, 0.0086, 0.0066, 0.0039, 0.0007, -0.0028, -0.0064, -0.01,
-0.0132, -0.0157, -0.0174, -0.0181, -0.0174, -0.0154, -0.0119, -0.007, -0.0008,
0.0067, 0.0152, 0.0244, 0.034, 0.0436, 0.053, 0.0616, 0.0692, 0.0755, 0.0802,
0.0831, 0.0841, 0.0831, 0.0802, 0.0755, 0.0692, 0.0616, 0.053, 0.0436, 0.034,
0.0244, 0.0152, 0.0067, -0.0008, -0.007, -0.0119, -0.0154, -0.0174, -0.0181,
-0.0174, -0.0157, -0.0132, -0.01, -0.0064, -0.0028, 0.0007, 0.0039, 0.0066,
0.0086, 0.0099, 0.0104, 0.0102, 0.0093, 0.0079, 0.006, 0.0038, 0.0015, -0.0007,
-0.0028, -0.0046, -0.0059, -0.0068, -0.0072, -0.007, -0.0064, -0.0054, -0.0041,
-0.0026, -0.0009, 0.0007, 0.0022, 0.0035, 0.0044, 0.0051, 0.0053, 0.0052,
0.0047, 0.004, 0.003, 0.0018, 0.0006, -0.0007, -0.0018, -0.0027, -0.0035,
-0.0039, -0.0041, -0.004, -0.0036, -0.003, -0.0022, -0.0013, -0.0003, 0.0006,
0.0015, 0.0022, 0.0028, 0.0031, 0.0033, 0.0031, 0.0028, 0.0023, 0.0017, 0.001,
0.0002, -0.0006, -0.0013, -0.0018, -0.0023, -0.0025, -0.0026, -0.0025, -0.0022,
-0.0018, -0.0013, -0.0007, -0.0001, 0.0005, 0.0011, 0.0015, 0.0018, 0.002,
0.0021, 0.002, 0.0017, 0.0014, 0.001, 0.0005, 0.0, -0.0005, -0.0009, -0.0012,
-0.0015, -0.0016, -0.0017, -0.0016, -0.0014, -0.0011, -0.0007, -0.0004, 0.0,
0.0004, 0.0007, 0.001, 0.0012, 0.0013, 0.0013, 0.0012, 0.0011, 0.0008, 0.0006,
0.0003, -0.0001, -0.0004, -0.0006, -0.0008, -0.001, -0.001, -0.001, -0.001,
-0.0008, -0.0006, -0.0004, -0.0002, 0.0001, 0.0003, 0.0005, 0.0007, 0.0008,
0.0008, 0.0008, 0.0007, 0.0006, 0.0005, 0.0003, 0.0001,
-0.0001, -0.0003, -0.0004, -0.0005, -0.0006, -0.0006, -0.0006, -0.0006, -0.0005,
-0.0004, -0.0002, -0.0001, 0.0001, 0.0002, 0.0003, 0.0004, 0.0005, 0.0005,
0.0005, 0.0004, 0.0003, 0.0003, 0.0002, 0.0, -0.0001, -0.0002, -0.0002, -0.0003,
-0.0003, -0.0004, -0.0003, -0.0003, -0.0002, -0.0002, -0.0001, -0.0, 0.0001,
0.0001, 0.0002, 0.0002, 0.0003, 0.0003, 0.0002, 0.0002, 0.0002,
0.0001, 0.0001, 0.0, -0.0, -0.0001, -0.0001, -0.0002, -0.0002, -0.0002, -0.0002,
-0.0002, -0.0001, -0.0001, -0.0, -0.0, 0.0, 0.0001, 0.0001, 0.0001, 0.0001,
0.0001, 0.0001, 0.0001, 0.0001, 0.0001, 0.0, -0.0, -0.0, -0.0001, -0.0001,
-0.0001, -0.0001, -0.0001, -0.0001, -0.0001, -0.0001, -0.0001, -0.0, 0.0, 0.0,
0.0001, 0.0001, 0.0001]
```

(b) Write a MATLAB program to implement the scheme, and plot the original signal and the sampled signal at the rate of 22 kHz versus the sample number, respectively.

Up-sampled signal = $\frac{1}{L}$ **original signal**, (L=11)