Laboratory Exercise 5
DIGITAL PROCESSING OF CONTINUOUS-TIME SIGNALS

5.1 THE SAMPLING PROCESS IN THE TIME-DOMAIN

Project 5.1 Sampling of a Sinusoidal Signal

A copy of Program P5_1 is given below:

```matlab
% Program P5_1
% Illustration of the Sampling Process
% in the Time-Domain
clf;
t = 0:0.0005:1;
f = 13;
xa = cos(2*pi*f*t);
subplot(2,1,1)
plot(t,xa);grid
xlabel('Time, msec');ylabel('Amplitude');
title('Continuous-time signal x_{a}(t)');
axis([0 1 -1.2 1.2])
subplot(2,1,2);
T = 0.1;
n = 0:T:1;
xs = cos(2*pi*f*n);
k = 0:length(n)-1;
stem(k,xs);grid;
xlabel('Time index n');ylabel('Amplitude');
title('Discrete-time signal x[n]');
axis([0 (length(n)-1) -1.2 1.2])
```
**Answers:**

**Q5.1** The plots of the continuous-time signal and its sampled version generated by running Program P5_1 are shown below:

![Continuous-time signal and its sampled version](image_url)

**Q5.2** The frequency of the sinusoidal signal in Hz is \( f = 13 \text{ kHz} \). This is because the time scale shown in the upper graph is in msec. Thus, the sinusoid goes through 13 cycles in 1 msec, or 13,000 cycles in one sec.

The sampling period in seconds is \( T = 0.1 \text{ msec} \).

**Q5.3** The effects of the two `axis` commands are:
- The first `axis` command sets the minimum and maximum values for the x-axis and the y-axis in the upper plot.
- The second `axis` command does the same thing for the lower plot. In each `axis` command, the first two parameters are the minimum and maximum values for the x-axis. The third and fourth parameters are the minimum and maximum values for the y-axis.

**Q5.4** The plots of the continuous-time signal and its sampled version generated by running Program P5_1 for the following four values of the sampling period are shown below:
$T = 0.004 \text{ msec}$

Continuous-time signal $x_a(t)$

Discrete-time signal $x[n]$

$T = 0.02 \text{ msec}$

Continuous-time signal $x_a(t)$

Discrete-time signal $x[n]$
\[ T = 0.15 \text{ msec} \]

Continuous-time signal \( x_a(t) \)

\[ T = 0.2 \text{ msec} \]

Continuous-time signal \( x_a(t) \)

Discrete-time signal \( x[n] \)
Based on these results we make the following observations — Since the “analog” waveform goes through 13 cycles in the 1 msec shown in the graph, there will be aliasing unless we get at least two samples per cycle, or 26 samples total on the graph. 26 samples in 1 msec is a sampling rate of 26 kHz, which requires $T < 1/26 \approx 0.0385$ msec to avoid aliasing. With $T=0.004$ msec, there is no aliasing and the discrete-time waveform has an appearance that is very similar to that of the “analog” waveform. With $T=0.02$ msec, there is still no aliasing, but the sampling rate is much closer to being “critical,” i.e., much closer to the Nyquist rate. Consequently, the appearance of the discrete-time waveform is less similar to the “analog waveform,” although perfect reconstruction is still possible. With $T=0.15$ msec, there is significant aliasing and the discrete-time waveform has the appearance of an “analog” waveform of much lower frequency. Finally, with $T=0.2$ msec, there is again severe aliasing which causes the discrete-time waveform to have the appearance of an “analog” waveform of lower frequency.

Q5.5 The plots of the continuous-time sinusoidal signal of frequency 3 kHz and its sampled version generated by running a modified Program P5.1 are shown below: (NOTE: the book is in error about the frequency. Since the “t” in the program is in units of msec, the frequency here is 3 kHz, not 3 Hz.)
The plots of the continuous-time sinusoidal signal of frequency 7 kHz and its sampled version generated by running a modified Program P5.1 are shown below: (Again, the book is in error. For the same reason as above, the frequency here is 7 kHz, not 7 Hz.)

Based on these results we make the following observations – In all three cases, we have $T = 0.1$ msec. Originally, in Q5.1, we had $f = 13$ kHz. This gives a maximum sampling period of $1/26 \approx 0.0385$ msec to satisfy the Nyquist criterion, as we observed above. Since $0.1 > 1/26$, the Nyquist criterion is not satisfied and there is aliasing in the original discrete-time waveform of Q5.1. In Q5.5 with $f = 3$ kHz, the maximum sampling period that satisfies the Nyquist criterion is $1/6 \approx 0.1667$ msec, Therefore, with $T = 0.1$ msec we obtain a discrete-time waveform that is not aliased. Finally, in Q5.5 with $f = 7$ kHz, the maximum sampling period to avoid aliasing is $1/14 \approx 0.0714$ msec. Thus, with $T = 0.1$ msec there is aliasing in the discrete-time waveform, but it is not as severe as what we saw above when $f = 13$ kHz. Interestingly, the samples $x[n]$ are exactly the same in all three cases!
Project 5.2  Aliasing Effect in the Time-Domain

A copy of Program P5_2 is given below:

```matlab
% Program P5_2
% Illustration of Aliasing Effect in the Time-Domain
% Program adapted from [Kra94] with permission from
% The Mathworks, Inc., Natick, MA.
cif;
T = 0.1; f = 13;
n = (0:T:1)';
x = cos(2*pi*f*n);
t = linspace(-0.5,1.5,500)';
y = sinc((1/T)*t(:,ones(size(n))) - (1/T)*n(:,ones(size(t)))')*x;
plot(n,x,'o',t,ya);grid;
xlabel('Time, msec');ylabel('Amplitude');
title('Reconstructed continuous-time signal y_{a}(t)');
axis([0  1  -1.2 1.2]);
```

Answers:

Q5.6  The plots of the discrete-time signal and its continuous-time equivalent obtained by running Program P5_2 are shown below:

Q5.7  The range of $t$ in the Program is $-0.5 \leq t \leq 1.5$

The value of the time increment is $-0.004$

The range of $t$ in the plot is $0.0 \leq t \leq 1.0$
The plot generated by running Program P5_2 again with the range of $t$ changed so as to display the full range of $y_a(t)$ is shown below:

![Reconstructed continuous-time signal $y_a(t)$]

Based on these results we make the following observations – To understand what is going on, the important line of the program is line 10, which reads:

$$y_a = \text{sinc} \left( \frac{1}{T} \cdot t(:,\text{ones(size}(n))) - \frac{1}{T} \cdot n(:,\text{ones(size}(t)))' \right) \cdot x_s;$$

Here, $t(:,\text{ones(size}(n)))$ is a matrix. It has one column for each element of the vector “n.” Each column contains a copy of the entire vector “t.” Similarly, $n(:,\text{ones(size}(t)))'$ is a matrix that has one row for each each entry in the vector “t.” Each one of these rows contains a copy of the vector “n.” It is important to realize that the vector “n” does not actually contain integers as you might expect. What it actually contains are the values $nT$, i.e., it contains integer multiples of the sampling interval.

So the quantity $t(:,\text{ones(size}(n))) - n(:,\text{ones(size}(t)))'$ is a matrix that has one row for each element of the vector “t” and one column for each element of the vector “n.” The argument of “sinc” in line 10 of the program is this matrix times $1/T$.

That is a matrix where the first row is exactly the values $(t-nT)/T$ shown on page 73 of the Lab Manual in the sum of Eq. (5.14) when $t$ is the first entry in the vector “t” and $n$ ranges not from $-\infty$ to $\infty$, but rather over all the values in the vector “n.”

Likewise, the second row of this matrix is exactly the values $(t-nT)/T$ from Eq. (5.14) when $t$ is the second entry of the vector “t” and the third row of this matrix is the values $(t-nT)$ in (5.14) when $t$ is the third entry of the vector “t.”
So \( \text{sinc}() \) in line 10 of the program returns a matrix. Each row of the matrix contains the values

\[
\frac{\sin(\pi(t - nT)/T)}{\pi(t - nT)/T}
\]

where \( t \) is an entry from the vector “t.” For the first row of this matrix, \( t \) is the first entry of the vector “t.” For the second row of this matrix, \( t \) is the second entry of the vector “t,” and so on.

When you multiply this matrix of \( \text{sinc}() \) values times \( \text{xs} \) as is done in line 10 of the program, you get a vector where each entry gives a partial realization of the sum in (5.14) for one of the values \( t \) in the vector “t.” It is a partial realization because the summation implemented by line 10 of the program does not go from \( -\infty \) to \( \infty \). It only adds up the terms from zero to \( \text{size(n)}-1 \).

And that is why the reconstructed results shown in the graph above are only a poor approximation when \( t < 0 \) and when \( t > 1 \). For these ranges of \( t \), the partial realization of the sum (5.14) that is implemented by line 10 of the program includes few or no samples on the main lobe of the sinc function, resulting in a poor approximation.

**Q5.8** The plots of the discrete-time signal and its continuous-time equivalent obtained by running Program P5_2 with the original display range restored and with the frequency of the sinusoidal signal changed to 3 kHz are shown below: (NOTE: the book is again in error; the frequency of this sinusoid is 3kHz, not 3 Hz, because time in the program is in units of msec, not sec).
The plots of the discrete-time signal and its continuous-time equivalent obtained by running Program P5_2 with the original display range restored and with the frequency of the sinusoidal signal changed to 7 kHz are shown below: (Again the book is in error. The frequency is here is 7 kHz, not 7 Hz.)

Based on these results we make the following observations – With $f = 13$ kHz in Q5.6, there is fairly severe aliasing; the reconstructed analog waveform is not of the same frequency as the original analog waveform that was sampled. With $f = 3$ kHz in Q5.8, there is no aliasing. The reconstructed analog waveform exactly matches the original analog waveform that was sampled. With $f = 7$ kHz in Q5.8, there is aliasing but it is not as severe as what we saw in Q5.6.

These results can be explained as follows – In all three cases, the sampling interval is $T = 0.1$ msec. For an original analog sinusoid of frequency $f = 13$ kHz, the sampling interval would have to be shorter than $1/26 \approx 0.0385$ msec to avoid aliasing. Since that’s not the case (we have $T = 0.1$ msec), aliasing occurs. For an original analog sinusoid of frequency $f = 3$ kHz, the sampling interval would have to be shorter than $1/6 \approx 0.1667$ msec to avoid aliasing. Since that is the case here (we have $T = 0.1$ msec), aliasing does not occur. Finally, for an original analog sinusoid of frequency $f = 7$ kHz, the sampling interval would have to be shorter than $1/14 \approx 0.0714$ msec to avoid aliasing. Since that’s not the case here (we have $T=0.1$ msec), aliasing occurs.
5.2 EFFECT OF SAMPLING IN THE FREQUENCY-DOMAIN

Project 5.3 Aliasing Effect in the Frequency-Domain

A copy of Program P5_3 is given below:

```matlab
% Program P5_3
% Illustration of the Aliasing Effect
% in the Frequency-Domain
clf;
t = 0:0.005:10;
xa = 2*t.*exp(-t);
subplot(2,2,1)
plot(t,xa);grid
xlabel('Time, msec');ylabel('Amplitude');
title('Continuous-time signal x_{a}(t)');
subplot(2,2,2)
wa = 0:10/511:10;
ha = freqs(2,[1 2 1],wa);
plot(wa/(2*pi),abs(ha));grid;
xlabel('Frequency, kHz');ylabel('Amplitude');
title('|X_{a}(j\Omega)|');
axis([0 5/pi 0 2]);
subplot(2,2,3)
T = 1;
n = 0:T:10;
xs = 2*n.*exp(-n);
k = 0:length(n)-1;
stem(k,xs);grid
xlabel('Time index n');ylabel('Amplitude');
title('Discrete-time signal x[n]');
subplot(2,2,4)
wd = 0:pi/255:pi;
hd = freqz(xs,1,wd);
% The following line has to be corrected from Mitra's version
% by adding "2" to the denominator of wd
plot(wd/(T*2*pi), T*abs(hd));grid;
xlabel('Frequency, kHz');ylabel('Amplitude');
title('|X(e^{j\Omega})|');
axis([0 0.5/T 0 2])
```

Answers:

**Q5.9** The continuous-time function \( x_a(t) \) in Program P5_3 is \( x_a(t) = 2te^{-t}u(t) \).

The CTFT of \( x_a(t) \) is being computed by – The Laplace transform was computed manually using a table to obtain

\[
X_a(s) = \frac{2}{(1+s)^2} = \frac{2}{s^2 + 2s + 1}.
\]

The coefficient vectors \([2]\) and \([1 \ 2 \ 1]\) where then used as arguments to `freqs` to obtain complex-valued samples of the frequency response \( X_a(\Omega) \) at the frequencies given in the vector \( wa \).
Q5.10 The plots generated by running Program P5_3 are shown below:

![Plots](image)

Based on these results we make the following observations — The analog signal is not bandlimited so there is aliasing. This is particularly visible in the raised tail of \(|X(e^{j\omega})|\) at 0.5 kHz relative to the tail of \(|X_\omega(\Omega)|\).

Q5.11 The plots generated by running Program P5_3 with sampling period increased to 1.5 are shown below:
Based on these results we make the following observations – with the reduction in sampling frequency, the aliasing is now much more severe. The tail of $|X(e^{j\omega})|$ is now raised even higher than it was before.

Q5.12 The modified Program P5_3 for the case of $x_a(t) = e^{-\pi t^2}$ is given below. Note: the CTFT was determined as follows. By applying the time scaling property to the Fourier transform pair $e^{-\frac{1}{4}t^2} \longleftrightarrow \sqrt{2\pi}e^{-\frac{1}{2}x^2}$ with a scale factor $a = \sqrt{2\pi}$, we obtain

$$X_a(\Omega) = e^{\frac{\Omega^2}{4\pi}}.$$

Since this function is real and non-negative, we have moreover that $|X_a(\Omega)| = X_a(\Omega)$. Thus, it is straightforward to include a code in the program to calculate the CTFT spectrum directly.

```matlab
% Program Q5_12
% Illustration of the Aliasing Effect
% in the Frequency-Domain
clf;
t = 0:0.005:10;
xa = exp(-pi*(t.^2));
subplot(2,2,1)
plot(t,xa);grid
xlabel('Time, msec');ylabel('Amplitude');
title('Continuous-time signal x_a(t)');
subplot(2,2,2)
wa = 0:10/511:10;
ha = exp(-(wa.^2)/(4*pi));
```
Based on these results we make the following observations – The continuous-time signal $x_a(t)$ is a rapidly decaying Gaussian in this case. Its Fourier transform is therefore a slowly decaying Gaussian in frequency. As a result, there is even more aliasing in the discrete spectrum this time than in our previous examples. This is not unexpected in
view of the fact that we have really only two samples of $x[n]$ falling on the main lobe of $x_a(t)$.

The plots generated by running the modified Program P5_3 with sampling period increased to 1.5 are shown below:

Based on these results we make the following observations – with the sampling rate significantly decreased in this case, the aliasing is now severe. The sampled waveform $x[n]$ has the appearance of a Kronecker delta (discrete-time unit impulse). As a result, the magnitude spectrum $|X(e^{j\omega})|$ is practically flat.

5.3 DESIGN OF ANALOG LOWPASS FILTERS

Project 5.4 Design of Analog Lowpass Filters

A copy of Program P5_4 is given below:

```matlab
% Program P5_4
% Design of Analog Lowpass Filter
clf;
Fp = 3500;Fs = 4500;
Wp = 2*pi*Fp; Ws = 2*pi*Fs;
[N, Wn] = buttord(Wp, Ws, 0.5, 30,'s');
[b,a] = butter(N, Wn, 's');
wa = 0:(3*Ws)/511:3*Ws;
h = freqs(b,a,wa);
```
Answers:

Q5.13 The passband ripple $R_p$ in dB is $-0.5$ dB

The minimum stopband attenuation $R_s$ in dB is $-30$ dB

The passband edge frequency in Hz is $-3.5$ kHz

The stopband edge frequency in Hz is $-4.5$ kHz

Q5.14 The gain response obtained by running Program P5_4 is shown below:

Based on this plot we conclude that the filter designed MEETS the given specifications.

The filter order $N$ is $-18$

The 3-dB cutoff frequency in Hz of the filter is $-W_n = 3.7144$ kHz
The required modifications to Program P5_4 to design a Type 1 Chebyshev lowpass filter meeting the same specifications are given below:

% Program Q5_15
% Design of Analog Lowpass Filter
clf;
Fp = 3500;Fs = 4500;
Wp = 2*pi*Fp; Ws = 2*pi*Fs;
[N, Wn] = cheb1ord(Wp, Ws, 0.5, 30,'s');
[b,a] = cheby1(N, 0.5, Wn, 's');
wa = 0:(3*Ws)/511:3*Ws;
h = freqs(b,a,wa);
plot(wa/(2*pi), 20*log10(abs(h)));grid
xlabel('Frequency, Hz');ylabel('Gain, dB');
title('Gain response');
axis([0 3*Fs -60 5]);
N

The gain response obtained by running the modified Program P5_4 is shown below:

![Gain response graph]

Based on this plot we conclude that the filter designed MEETS the given specifications.

The filter order N is - 8
The passband edge frequency in Hz of the filter is $W_n = 3.5 \text{ kHz}$

Q5.16 The required modifications to Program P5_4 to design a Type 2 Chebyshev lowpass filter meeting the same specifications are given below:

% Program Q5_16
% Design of Analog Lowpass Filter
clf;
Fp = 3500;Fs = 4500;
Wp = 2*pi*Fp; Ws = 2*pi*Fs;
[N, Wn] = cheb2ord(Wp, Ws, 0.5, 30,'s');
[b,a] = cheby2(N, 30, Wn, 's');
wa = 0:(3*Ws)/511:3*Ws;
h = freqs(b,a,wa);
plot(wa/(2*pi), 20*log10(abs(h)));grid
xlabel('Frequency, Hz');ylabel('Gain, dB');
title('Gain response');
axis([0 3*Fs -60 5]);
N
Wn

The gain response obtained by running the modified Program P5_4 is shown below:

Based on this plot we conclude that the filter designed MEETS the given specifications.

The filter order N is -8
The stopband edge frequency in Hz of the filter is $–4.2653$ kHz.

Q5.17 The required modifications to Program P5_4 to design an elliptic lowpass filter meeting the same specifications are given below:

% Program Q5_17
% Design of Analog Lowpass Filter
cif;
Fp = 3500; Fs = 4500;
Wp = 2*pi*Fp; Ws = 2*pi*Fs;
[N, Wn] = ellipord(Wp, Ws, 0.5, 30,'s');
[b,a] = ellip(N, 0.5, 30, Wn, 's');
wa = 0:(3*Ws)/511:3*Ws;
h = freqs(b,a,wa);
plot(wa/(2*pi), 20*log10(abs(h)));grid
xlabel('Frequency, Hz');ylabel('Gain, dB');
title('Gain response');
axis([0 3*Fs -60 5]);
N
Wn/(2*pi)

The gain response obtained by running the modified Program P5_4 is shown below:

![Gain response graph]

Based on this plot we conclude that the filter designed **MEETS** the given specifications.

The filter order $N$ is $5$.
The passband edge frequency in Hz of the filter is \( W_n = 3.5 \text{ kHz} \)

### 5.4 A/D AND D/A CONVERSIONS

#### Project 5.5 Binary Equivalent of a Decimal Number

**Answers:**

Q5.18 The function of the operator `==` is – “==” is a relational operator that returns a value of one if its two operands are equal and a value of zero if its two operands are unequal.

Q5.19 The binary equivalents in sign-magnitude form of the decimal fractions are:

(a) \( 0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1; \ 0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1 \)

(b) \( 1 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1; \ 1 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1 \)

(c) \( 0 \ 1 \ 0 \ 1 \ 0 \ 0 \ 1; \ 0 \ 1 \ 0 \ 1 \ 0 \ 0 \ 1 \ 0 \ 0 \)

(d) \( 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 0; \ 1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 1 \)

#### Project 5.6 Decimal Equivalent of a Binary Number

**Answer:**

Q5.20 The decimal equivalents of the binary fractions along with the errors in conversion are as follows *(Note: use program P5_6fix.m for this. The original program P5_6.m contains errors)*:

(a) \( 0.796875 \) (error=0.004775); \( 0.80078125 \) (error=0.00086875)

(b) \( -0.796875 \) (error=-0.004775); \( -0.80078125 \) (error=-0.00086875)

(c) \( 0.640625 \) (error=0.002705); \( 0.64062500 \) (error=0.002705)

(d) \( -0.90625 \) (error=-0.00625); \( -0.91015625 \) (error=-0.00234375)

#### Project 5.7 Binary Number Representation Schemes

**Answers:**

Q5.21 The function of the operator `~` is - *(Note: the “~=” operator does not appear in P5_7.m. As stated in the lab manual, what is meant in this question is the “~” operator)*. The “~” operator performs a bitwise logical complement. Colloquially, it “flips the bits.” It turns each “zero bit” into a “one bit” and each “one bit” into a “zero bit.”

Q5.22 The ones'-complement representations of the binary numbers developed in Question Q5.19 are as follows:
Q5.23 The function of the operator $|$ is – The “$|$” operator performs a bitwise logical OR operation on its operands.

Q5.24 The two's-complement representations of the binary numbers developed in Question Q5.19 are as follows: Note: It’s important to realize that P5.8 works differently from the others. For non-negative fractions, the representation in two’s complement, one’s complement, and signed magnitude are all exactly the same. The only difference is in the representation of negative fractions. P5.8 only works for negative fractions. For input, it takes the one’s complement representation of the negative fraction. For output, it produces the two’s complement representation of that same negative fraction. In particular, this means that the answers to Q5.24 (a) and (c) are the same as the answers to Q5.22 (a) and (c) and Q5.19 (a) and (c).

(a) $0 1 1 0 0 1 1; \quad 0 1 1 0 0 1 1 0 1$
(b) $1 0 0 1 1 0 0; \quad 1 0 0 1 1 0 0 1 0$
(c) $0 1 0 1 0 0 1; \quad 0 1 0 1 0 0 1 0 0$
(d) $1 0 0 0 1 0 1; \quad 1 0 0 0 1 0 1 1 0$

Project 5.8 D/A Converter Droop Compensation

Answer:

Q5.25 The MATLAB program to determine and plot the magnitude responses of the uncompensated and the droop-compensated D/A converters is given below:

Notes on the Program Development: The sampling interval is $T$, which will be set to $T = 1$ by default in the program. The uncompensated DAC is modeled as a series connection of two continuous-time systems: the ideal reconstruction filter $H_r(j\Omega)$ given in (5.10) on page 72 of the lab manual and the zero-order hold model $H_z(j\Omega)$ given in (5.27) on page 78 of the lab manual. All frequency response magnitudes will be plotted against a vector of 512 frequency samples equally spaced between DC and $\Omega_s / 2 = \pi / T$ rad/sec. To plot the gain of the uncompensated DAC, we simply evaluate $|H_r(j\Omega)H_z(j\Omega)|$ and convert to dB. To evaluate $H_z(j\Omega)$ at the point $\Omega = 0$, it is necessary to apply L’Hopital’s rule as follows:
\[
\lim_{\Omega \to 0} H_z(j\Omega) = e^0 \times \lim_{\Omega \to 0} \frac{\delta}{\pi T} \sin(\Omega T/2) = 1 \times \lim_{\Omega \to 0} \frac{T}{\Omega} \cos(\Omega T/2) = 1.
\]

For the other frequency samples, \( H_z(j\Omega) \) can be evaluated directly using formula (5.27) of the lab manual.

For the two systems with digital droop compensation, we consider that the series connection of the digital compensation filter with the ideal reconstruction filter results in an equivalent analog compensation filter that is in series with \( H_z(j\Omega) \) and \( H_r(j\Omega) \). For the FIR compensation filter, we have from (5.29) on page 86 of the lab manual that

\[
H_{FIR}(z) = \frac{1}{16} \left( -1 + 18z^{-1} - z^{-2} \right),
\]

from which we obtain

\[
H_{FIR}(e^{j\omega}) = \frac{1}{16} \left( -1 + 18e^{-j\omega} - e^{-j2\omega} \right).
\]

The frequency response of the equivalent analog compensation filter that is realized by the series combination of \( H_{FIR}(z) \) with the ideal DAC \( H_r(j\Omega) \) is given by

\[
H_{FIR,a}(j\Omega) = \begin{cases} H_{FIR}(e^{j\Omega}) , & |\Omega| \leq \pi / T \\ 0 & |\Omega| > \pi / T \end{cases}
= \begin{cases} \frac{1}{16} \left( -1 + 18e^{-j\Omega} - e^{-j2\Omega} \right) , & |\Omega| \leq \pi / T \\ 0 & |\Omega| > \pi / T \end{cases}.
\]

The overall magnitude response of the FIR compensated DAC is then given by

\[ |H_r(j\Omega)H_{FIR,a}(j\Omega)H_z(j\Omega)|, \]
which will be evaluated by the program and converted to units of dB for plotting.

For the IIR compensation filter, we have from (5.30) on page 86 of the lab manual that

\[
H_{IRR}(e^{j\omega}) = \frac{9}{8 + e^{-j\omega}},
\]
so

\[
H_{IRR,a}(j\Omega) = \begin{cases} H_{IRR}(e^{j\Omega}) , & |\Omega| \leq \pi / T \\ 0 & |\Omega| > \pi / T \end{cases}
= \begin{cases} \frac{9}{8 + e^{-j\Omega}} , & |\Omega| \leq \pi / T \\ 0 & |\Omega| > \pi / T \end{cases}.
\]

The overall magnitude response of the IIR compensated DAC is then given by

\[ |H_r(j\Omega)H_{IRR,a}(j\Omega)H_z(j\Omega)|, \]
which will be evaluated by the program and converted to units of dB for plotting.
The program is given below.

% Program Q5_25.m
% For a non-ideal sample-and-hold D/A converter,
% compute the magnitude response of
% 1. the uncompensated D/A
% 2. D/A with FIR droop compensation
% 3. D/A with IIR droop compensation
% all on a single graph.
T = 1.0; % sampling interval
wa = 0:(pi/T)/511:(pi/T); % analog freq vector
Lwa = length(wa);
Hra = T * ones(1,length(wa)); % ideal reconstruction filter
% Frequency response of uncompensated zero-order hold
Hza = zeros(1,Lwa);
Hza(1) = 1; % follows from L'Hopital's Rule
wa2 = wa(2:Lwa);
Hza(2:Lwa) = exp(-i*wa2*T*0.5) .* sin(0.5*T*wa2) ./ (0.5*T*wa2);
MagHza = abs(Hza);
GainHza = 20*log10(MagHza);
% Frequency response of FIR compensated D/A
Hfira = (1/16)*(-1 + 18*exp(-i*wa*T) - exp(-i*2*T*wa));
HfiraComp = Hfira .* Hza;
GainHfiraComp = 20*log10(MagHfiraComp);
% Frequency response of IIR compensated D/A
Hiira = 9*ones(1,Lwa) ./ (8 + exp(-i*wa*T));
HiiraComp = Hiira .* Hza;
GainHiiraComp = 20*log10(MagHiiraComp);

% plot
plot(wa,GainHza,wa,GainHfiraComp,'--',wa,GainHiiraComp,'-.');
grid;
xlabel('Frequency');
ylabel('Gain, dB');
axis([0 pi/T -4 1]);
legend('DAC','FIR','IIR');
The plot of the magnitude responses is shown below:

From this plot we make the following observations: Both the FIR and IIR compensated DACs offer a significant improvement in terms of reducing the drop of the uncompensated DAC. The spectral magnitude of the IIR compensated DAC is slightly flatter in the pass band and rolls off slightly slower than the FIR compensated DAC. It is interesting to note that this plot is in perfect agreement with the one given in Fig. 4.56 on page 224 of the course text.

Date: 18 November 2018
Signature: SOLUTION (Havlicek)