ELEC6410 Project 3

Answers to Digital Signal Processing Project #3

Michael J. Carroll

September 16, 2010

Question 1

Examine a sampled waveform

Part A

Sample the waveform $x(t) = 3\cos 3000\pi t + \cos 800\pi t$ at a sampling frequency of 16 kHz for one second:

Since the signal is sampled at greater than the Nyquist rate, it may be treated as a continuous signal.

```
t = [0:1/16000:1];
xc = 3*cos(3000*pi*t) + cos(800*pi*t);
```

Part B

Plot the first 100 points of the signal with an appropriately labeled t axis.

Plot is used for the case of a continuous signal in Figure 1.

```
figure(1);
plot(t(1:100),xc(1:100));
xlabel('Time in Seconds'); ylabel('Amplitude');
title('First 100 Points of Waveform xc');
```

Part C

Plot the Fourier-transform magnitude of x(t).

```
figure(2);
plot([-8000:7999],fftshift(abs(fft(xc(1:2*8000)))));
xlabel('Normalized Frequency (\times 2\pi to give rad/s)');
title('Fourier-transform magnitude of waveform xc');
```

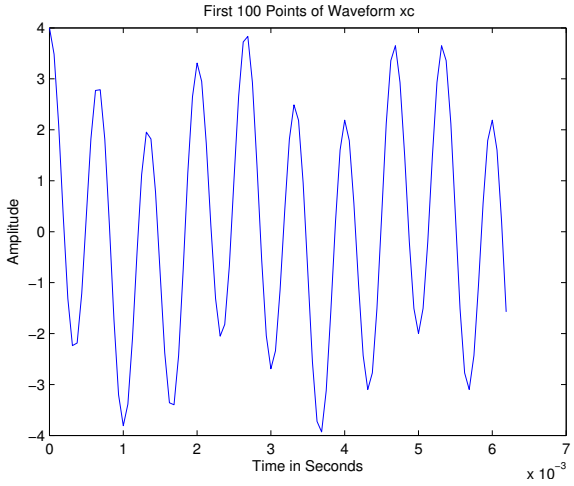

Figure 1: First 100 Points of Waveform x_c

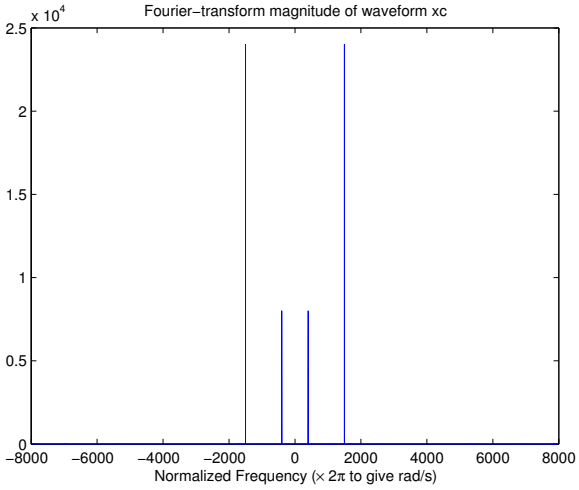

Figure 2: Fourier Transform Magnitude of Waveform x_c

Part D

Explain how this graph corresponds to the actual Fourier transform of x(t).

The plot of the Fourier transform in Figure 2. The important thing to note in the plot of the Fourier transform is that the amplitudes are a function of the number of samples as opposed to infinite Dirac Delta functions in the case of a true Fourier transform.

Question 2

Now we will consider the effect of sampling on the original signal.

Part A

Sample the waveform $x(t) = 3\cos 3000\pi t + \cos 800\pi t$ at a sampling frequency of 8kHz to obtain $x[n]$, where $t = nT$

```
tsampled = [0:1/8000:1];
xn1 = 3*cos(3000*pi*tsampled) + cos(800*pi*tsampled);
```

Part B

Plot the resulting sequence using stem for the first 100 points of the signal with an appropriately labeled n axis.

The plot of this signal is included in Figure 3.

```
figure(3);
stem(xn1(1:100));
title('First 100 points of x[n], Sampled at 8kHz');
xlabel('Samples (n)'); ylabel('Amplitude');
```

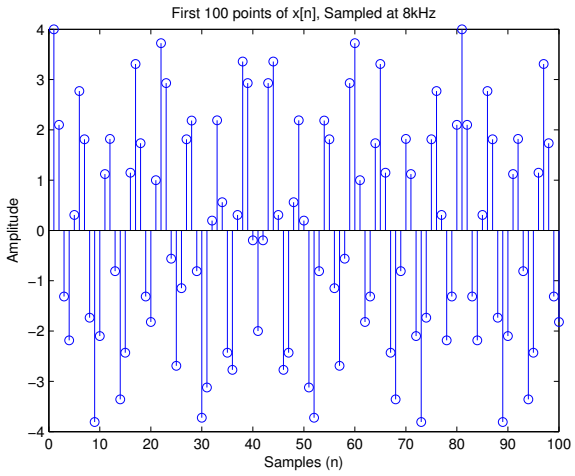

Figure 3: First 100 Points of Sequence $x_1[n]$

Part C

Plot the DTFT magnitude.

The plot of this signal is included in Figure 4. An additional plot with the three repeated copies of the spectrum is included in Figure 5.

```
figure(4);
plot([-4000:3999]/8000,fftshift(abs(fft(xn1(1:8000)))));
xlabel('Normalized frequency (\times 2\pi to give rad/sample)');

figure(5);
plot([-12000:11999]/8000,repmat(fftshift(abs(fft(xn1(1:8000)))),[1 3]));
xlabel('Normalized frequency (\times 2\pi to give rad/sample)');
```

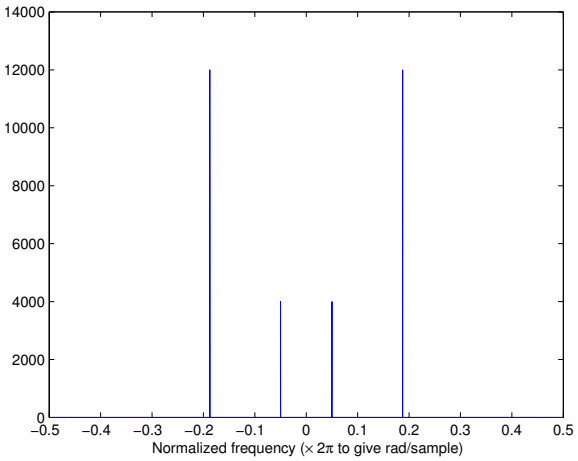

Figure 4: Fourier Transform Magnitude of Sequence $x_1[n]$

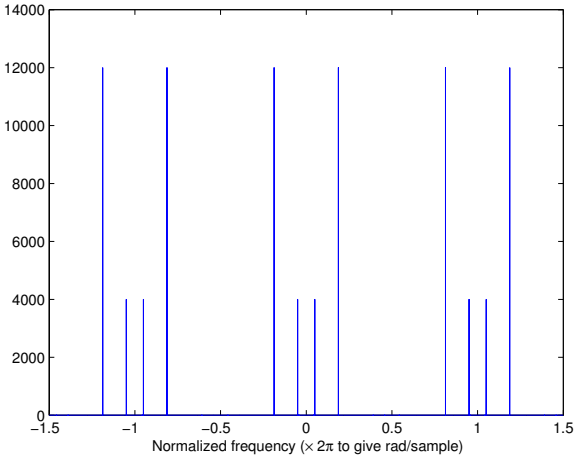

Figure 5: Fourier Transform Magnitude (extended) of Sequence $x_1[n]$

Part D

Does aliasing occur? Explain.

Aliasing does not occur in this sampled signal. The reason is that the sample rate is still above the Nyquist rate of 6000 kHz. Below that threshold is where aliasing will start to occur.

Question 3

Now consider a different sampling rate:

Part A

Sample the waveform in #1 at a sampling frequency of 2kHz for one second.

```
tsampled = [0:1/2000:1];
xn2 = 3*cos(3000*pi*tsampled) + cos(800*pi*tsampled);
```

Part B

Determine how to plot the DTFT magnitude in Question 2, Part C

The DTFT magnitude plot is included in 7. The additional plot with the three repeated copies of the spectrum is included in Figure 8

```
figure(6);
stem(xn2(1:100));
title('First 100 points of x[n], Sampled at 2kHz');
xlabel('Samples (n)'); ylabel('Amplitude');

figure(7);
plot([-1000:999]/2000,fftshift(abs(fft(xn2(1:2000)))));
xlabel('Normalized frequency (\times 2\pi to give rad/sample)');

figure(8);
plot([-3000:2999]/2000,repmat(fftshift(abs(fft(xn2(1:2000)))),[1 3]));
xlabel('Normalized frequency (\times 2\pi to give rad/sample)');
```

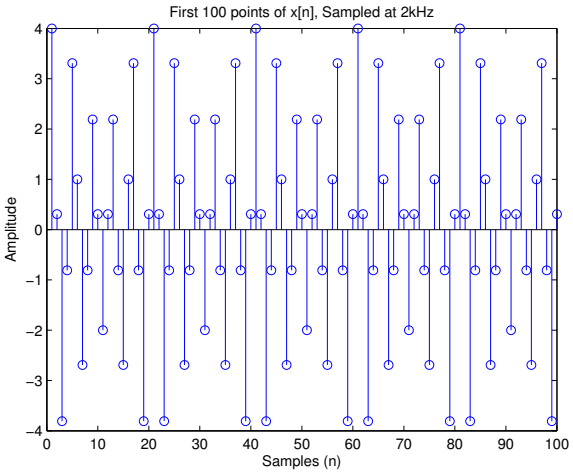

Figure 6: First 100 Points of Sequence $x_2[n]$

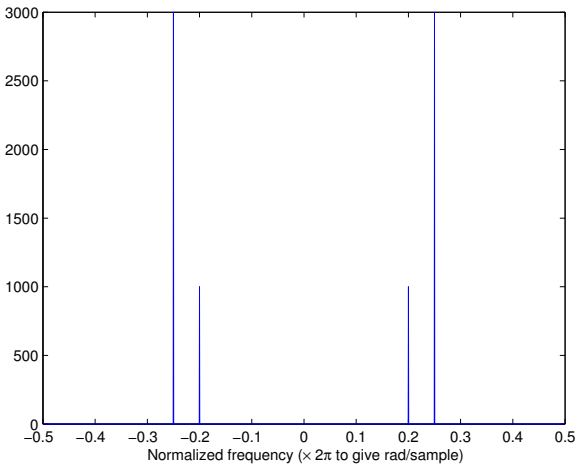

Figure 7: Fourier Transform Magnitude of Sequence $x_2[n]$

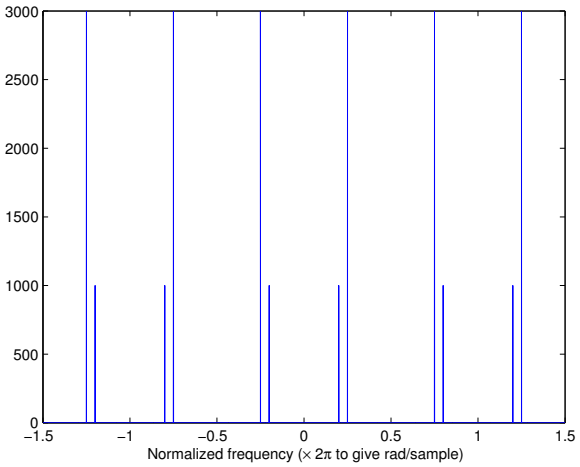

Figure 8: Fourier Transform Magnitude (extended) of Sequence $x_2[n]$

Part C

Explain the differences between this plot and the one in #2 based on the properties of sampling.

The plot for the lower sampling frequency in Figure 7 has different values of magnitude than the plot for the higher sampling frequency in Figure 4. This means that some of the information about the original signal has been lost over the course of sampling it at the lower frequency. Also, the frequencies have changed their positions due to undersampling.

Part D

Does aliasing occur? Explain.

Aliasing does occur in this sampled signal. The reason is that the sampling rate has dropped below the Nyquist rate of 6kHz for the original signal. It is immediately apparent from the Fourier plot that the frequency spectrum is no longer close to the original.

Question 4

We will now attempt to reconstruct the signal in #1 from the sequence in #2 using zero-order hold.

Part A

Form a zero-order hold signal with 2 equal output samples per input sample. Plot the first 100 points considered as a continuous-time signal.

Using some MATLAB magic (reshape and repmat), I can do this without any loops

```
xn1_zoh=reshape(repmat(xn1',1,2)', length(xn1(:,1)), 2*length(xn1(1,: )));
figure(9);
plot(t(1:100),xn1_zoh(1:100));
xlabel('Time in Seconds'); ylabel('Amplitude');
title('First 100 Points of Waveform XN1 with Zero-Order Hold');
```

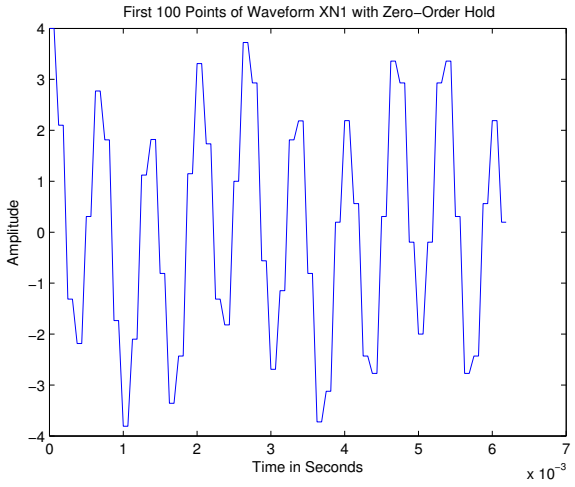

Figure 9: First 100 Points of Waveform $x_1[n]$ with Zero Order Hold

Part B

Plot the FT magnitude using the syntax from #1, and explain what you observe.

The Fourier transform plot included in Figure 10 now has some higher frequency components added from the zero-order hold. These additional frequency components come from the "sharp" squared-off edges, which contain high-frequencies.

```
figure(10);
plot([-8000:7999],fftshift(abs(fft(xn1_zoh(1:2*8000)))));
xlabel('Normalized Frequency (\times 2\pi to give rad/s)');
title('Fourier-transform magnitude of waveform XN1 with ZOH');
```

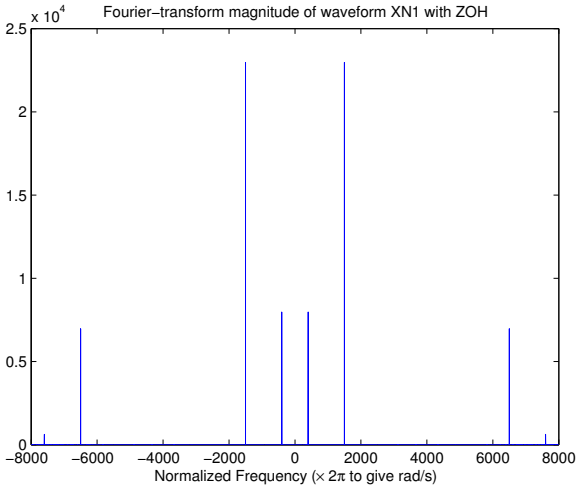

Figure 10: Fourier Transform Magnitude of $x_1[n]$ with Zero Order Hold

Part C

Lowpass filter the ZOH signal as follows:

```
h = fir1(9,1/2);
xn1_reconstructed = filter(h,1,xn1_zoh);
```

Part D

Plot the first 100 points considered as a continuous-time signal.

The reconstructed continuous-time signal is included in Figure 11

```
figure(11);
plot(t(1:100),xn1_reconstructed(1:100));
xlabel('Time in Seconds'); ylabel('Amplitude');
title('First 100 Points of Reconstructed Signal');
```

Part E

Plot the FT magnitude of the result using the syntax from #1. Explain what you observe. Is the signal in #1 perfectly reconstructed? Justify your answer.

The Fourier transform magnitude plots for the two signals are identical, so this leads me to believe that the signal has been perfectly reconstructed. This does not necessarily imply that the two signals are sample-for-sample identical, as can be seen in the differences between the continuous-time plots in Figure 11 and Figure 2, but that the frequency content of both signals matches in both frequency and magnitude.

```
figure(12);
plot([-8000:7999],fftshift(abs(fft(xn1_reconstructed(1:2*8000)))));
xlabel('Normalized Frequency (\times 2\pi to give rad/s)');
title('Fourier-transform magnitude of reconstructed waveform');
```

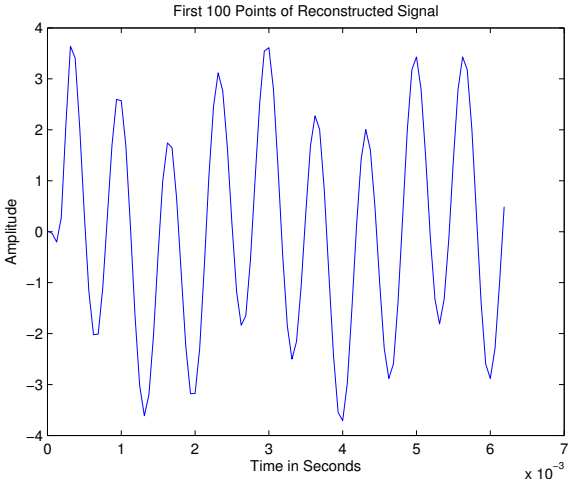

Figure 11: First 100 Points of Reconstructed Signal

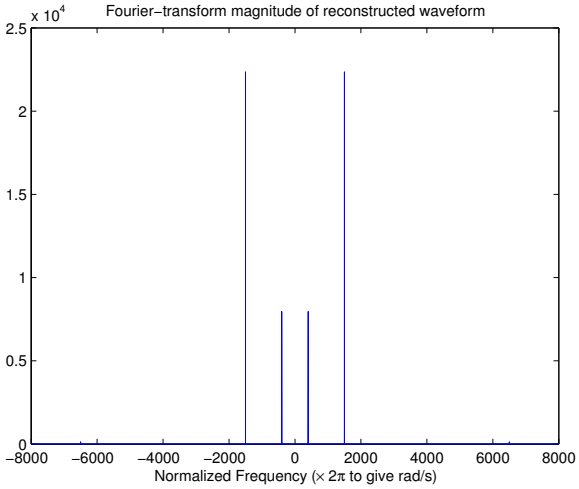

Figure 12: Fourier Transform Magnitude of Reconstructed Waveform with Lowpass

Question 5

Play the sound from #1, the ZOH sound, and the reconstructed sound using: soundsc(x,16000) Note any differences in the sounds.

The first sound (original signal) was a mixture of two tones with nothing remarkable about it. The second sound (ZOH) had some higher pitched ringing or noise in it. The higher pitched noise corresponds to what the Fourier transform magnitude plot in Figure 10. The third sound was similar to the first one.

```
% Commented so that I don't have to hear them every time I run the file.
% soundsc(xc,16000)
% soundsc(xn1_zoh,16000)
% soundsc(xn1_reconstructed,16000)
```

Question 6

Reconstruct the signal in #1 from the sequence in #2 using linear interpolation. You must use the filter function to get maximum credit. Compare the FT magnitude to the ZOH interpolation and the original. Explain the differences.

The first step in linear interpolation is the upsampling of the signal. In this case, I upsampled by a value of 2 (made every other sample a zero after expanding the signal).

The linear interpolation of the signal uses a filter with the following difference equation:

$$y[n] = x[n] + \frac{1}{2}x[n-1] + x[n+1]$$

$$y[n] = \frac{1}{2}x[n+1] + x[n] + \frac{1}{2}x[n-1]$$

This is a non-causal filter (which uses a sample from a future reading). To get around this, I can use the padding options of the MATLAB filter command.

To do this, I create the filter vector as I normally would, and MATLAB interprets this as:

$$y[n] = \frac{1}{2}x[n] + x[n-1] + \frac{1}{2}x[n-2]$$

Taking advantage of the initial conditions of the filter command, I made xn(2) and xn(1) initial conditions of the filter. I have included a small proof of concept of this operation for the series [1,5] with an interpolation factor of 2.

The Fourier transform magnitude of the linearly-interpolated waveform is included in Figure 13. Comparing this to Figure 10, some of the same higher-frequency components are there, but at a much lower magnitude. Doing the linear interpolation is similar to creating a low-pass filter as it "smooths off" some of the harsher transitions that the zero-order hold creates.

```
b = [.5 1 .5];

% Small proof of concept.
x = [1 0 -5 0 2 0 6 0 7 0];
x_linear = [x(1),filter(b,1,[x(3:end)],[.5*x(1),x(2)])]

xn1_upsampled = zeros(length(xn1)*2,1);
xn1_upsampled(1:2:length(xn1)*2) = xn1;

xn1_linear = [xn1_upsampled(1),...
    filter(b,1,xn1_upsampled(3:end),[.5*xn1(1),xn(2)])'];

figure(13);
plot([-8000:7999],fftshift(abs(fft(xn1_linear(1:2*8000)))));
xlabel('Normalized Frequency (\times 2\pi to give rad/s)');
title('Fourier-transform magnitude of linear interpolated waveform');
```

```
x_linear =

  Columns 1 through 5

    1.0000   -2.0000   -5.0000   -1.5000    2.0000

  Columns 6 through 9

    4.0000    6.0000    6.5000    7.0000
```

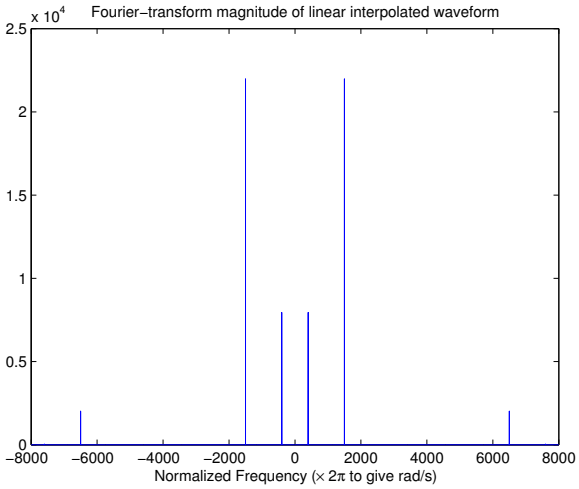

Figure 13: Fourier Transform Magnitude of Linear Interpolation