EE 542

Applications In Digital Signal Processing

Computer Assignment #2

Zexi Liu

Mar 6th, 2007

Electrical and Computer Engineering

Temple University

1. Determine the transfer function for a lowpass elliptic filter, given the following filter specifications: passband edge 0.5π, passband ripple 0.01 dB, stopband edge 0.55π and minimum stopband attenuation of 60 dB.

1.1 Introduction

An elliptic filter (also known as a Cauer filter) is an electronic filter with equalized ripple (equiripple) behavior in both the passband and the stopband. The amount of ripple in each band is independently adjustable, and no other filter of equal order can have a faster transition in gain between the passband and the stopband, for the given values of ripple (whether the ripple is equalized or not). Alternatively, one may give up the ability to independently adjust the passband and stopband ripple, and instead design a filter which is maximally insensitive to component variations.

As the ripple in the stopband approaches zero, the filter becomes a type I Chebyshev filter. As the ripple in the in the passband approaches zero, the filter becomes a type II Chebyshev filter and finally, as both ripple values approach zero, the filter becomes a Butterworth filter.

The gain of a lowpass elliptic filter as a function of angular frequency ω is given by:

$$G_n(\omega) = \frac{1}{\sqrt{1+\varepsilon^2 R_n^2(\xi, \omega/\omega_0)}}$$

where R_n is the nth-order elliptic rational function (sometimes known as a Chebyshev rational function) and

ω_0 is the cutoff frequency

ε is the ripple factor

ξ is the selectivity factor

The value of the ripple factor specifies the passband ripple, while the combination of the ripple factor and the selectivity factor specify the stopband ripple.

1.2 Matlab Code

See Appendix A.

1.3 Results

$$G(z)=\frac{\begin{array}{c}0.02658+0.1062\,z^{-1}+0.2784\,z^{-2}+0.5095\,z^{-3}+0.7236\,z^{-4}+0.8087\,z^{-5}\\+0.7236\,z^{-6}+0.5095\,z^{-7}+0.2784\,z^{-8}+0.1062\,z^{-9}+0.02658\,z^{-10}\end{array}}{\begin{array}{c}1-0.8302\,z^{-1}+3.185\,z^{-2}-2.299\,z^{-3}+3.812\,z^{-4}-2.267\,z^{-5}\\+2.063\,z^{-6}-0.9227\,z^{-7}+0.4613\,z^{-8}-0.1245z^{-9}+0.02414\,z^{-10}\end{array}}$$

Denominator coefficients (64-bit Double-precision floating-point)	Numerator coefficients (64-bit Double-precision floating-point)
1.000000000000000	0.026575792803376
-0.830247848216304	0.106155488025726
3.184774926923402	0.278354546412372
-2.29855573413249	0.509529039709704
3.812081796859765	0.723569984683564
-2.267302942708255	0.808650785998917
2.062727336707079	0.723569984683565
-0.922693507309877	0.509529039709705
0.461296694866491	0.278354546412373
-0.124477492975693	0.106155488025727
0.024136844696851	0.026575792803377

2. Determine the frequency characteristics for the original filter without coefficient quantization. Modify Program 12_1 which uses the Matlab function a2dT developed by Mitra to quantize the coefficients to a word length of 6 bits.

2.1 Matlab Code

See Appendix B.

2.2 Results

Figure 1. Original vs. 6-bit Quantized

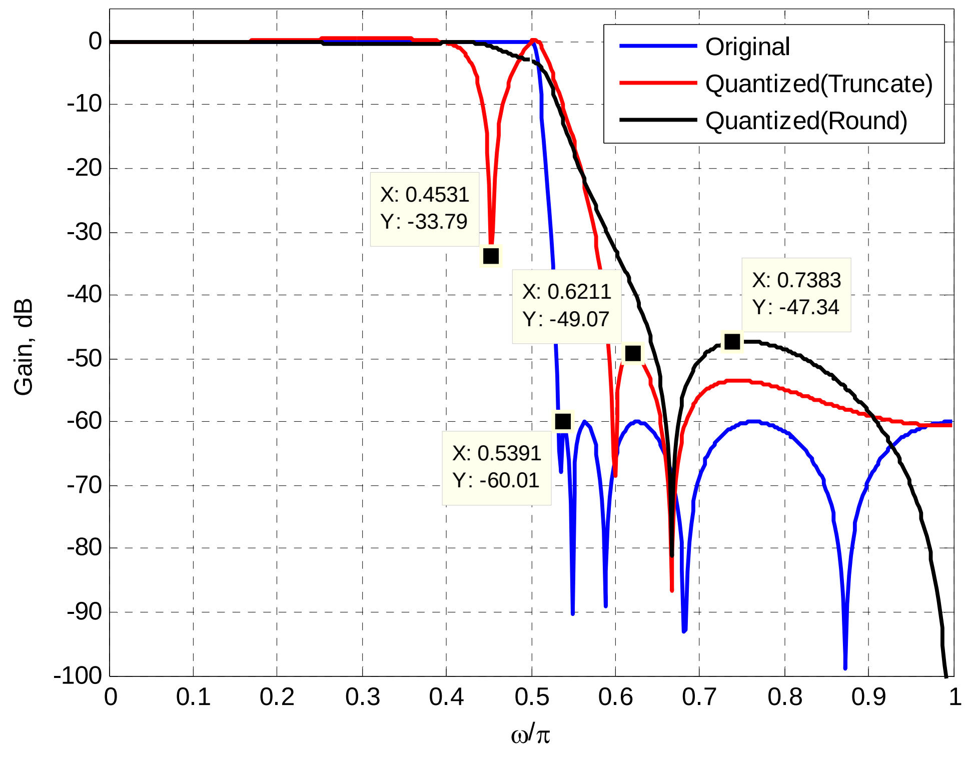


Figure 2. Original vs. 6-bit Quantized (Passband Details)

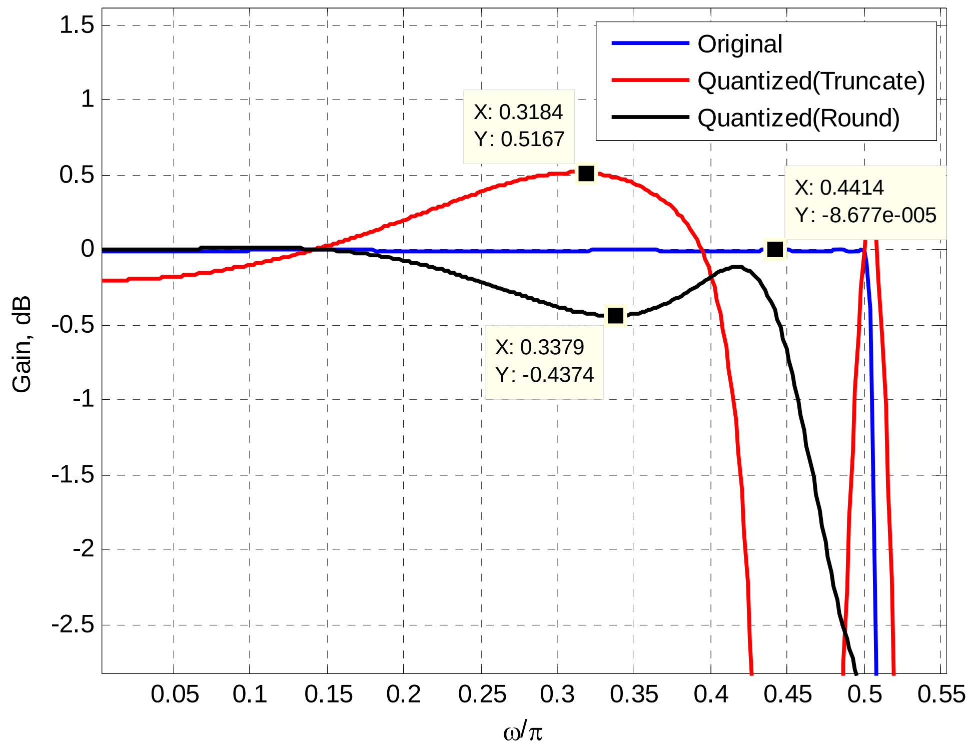


Figure 3. Original vs. 16-bit Quantized

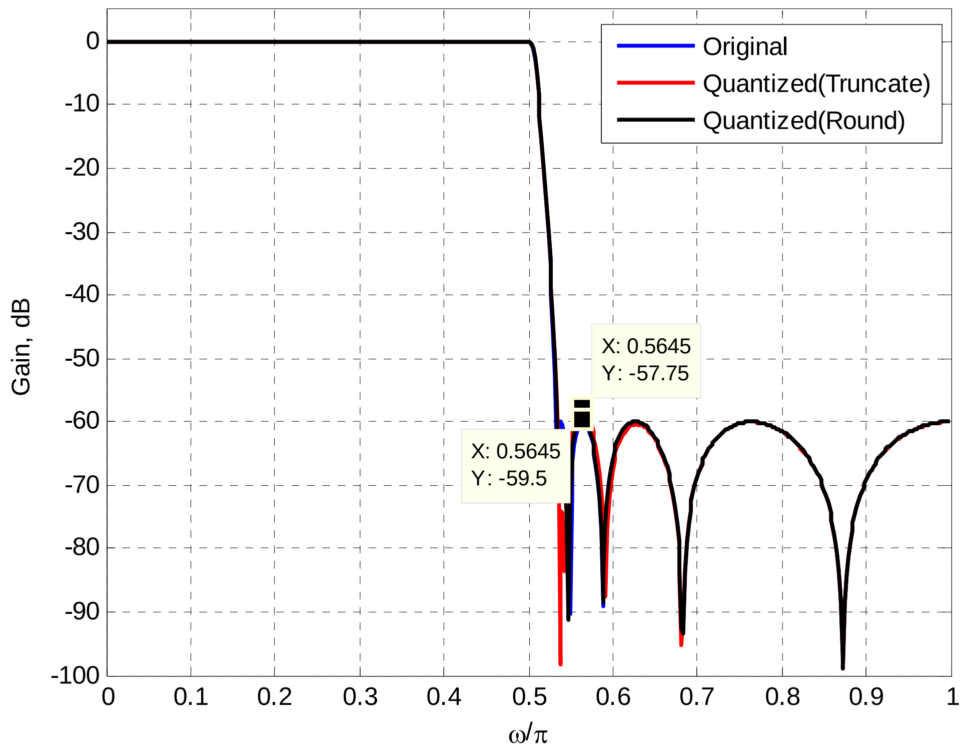


Figure 4. Original vs. 16-bit Quantized

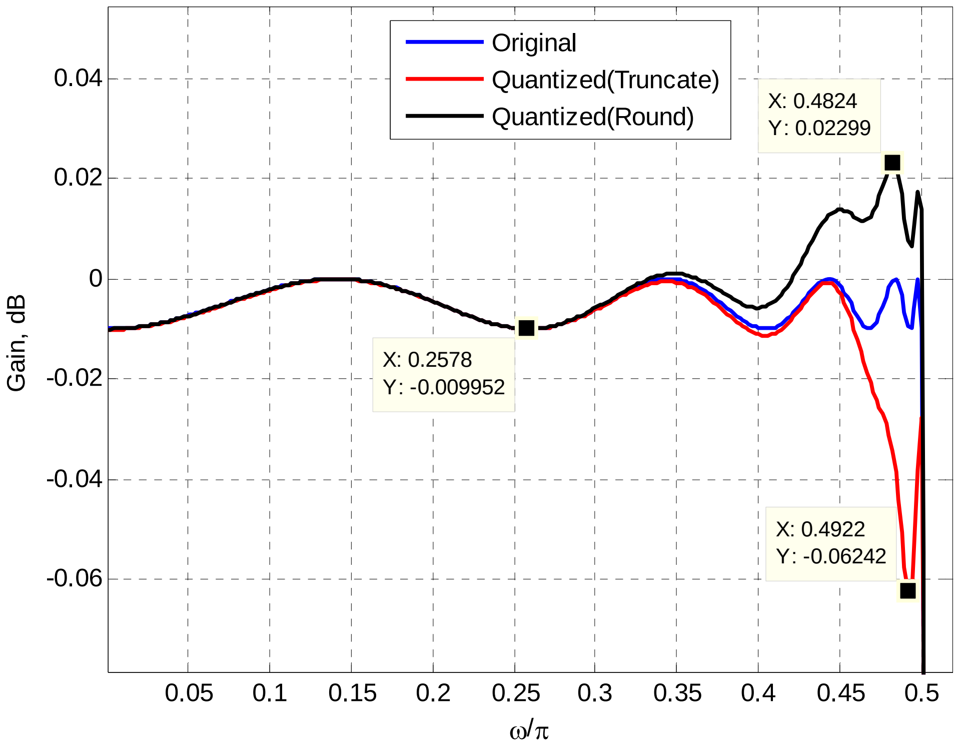


Figure 5. Original vs. 6-bit Quantized(Truncate)

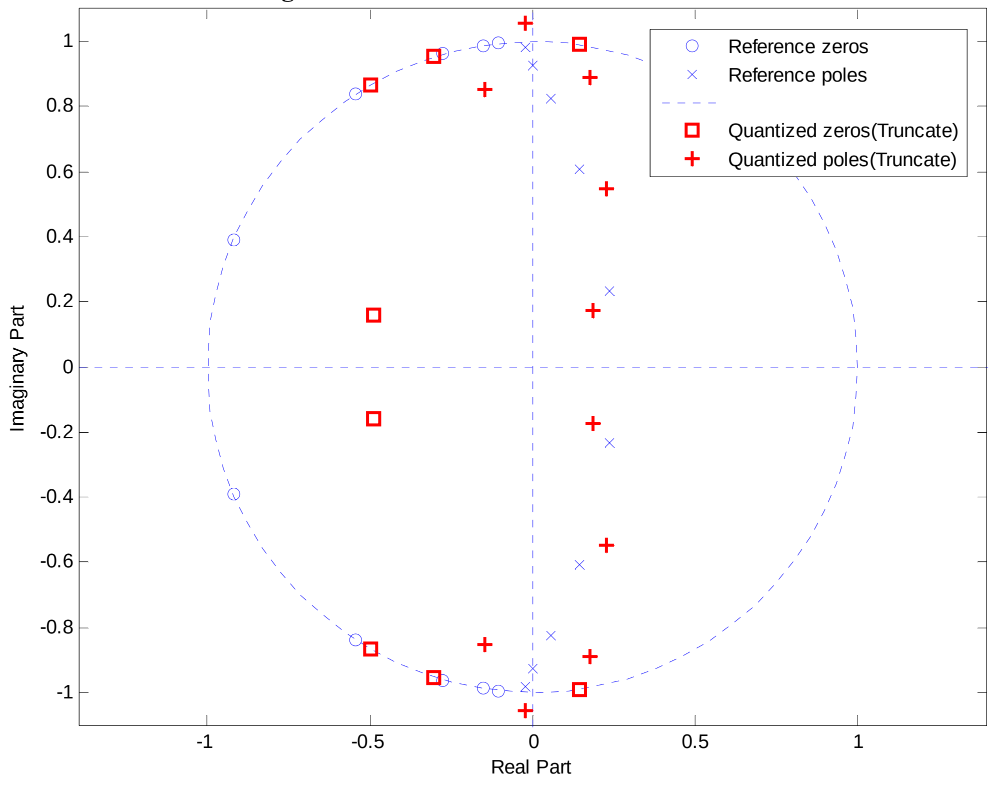
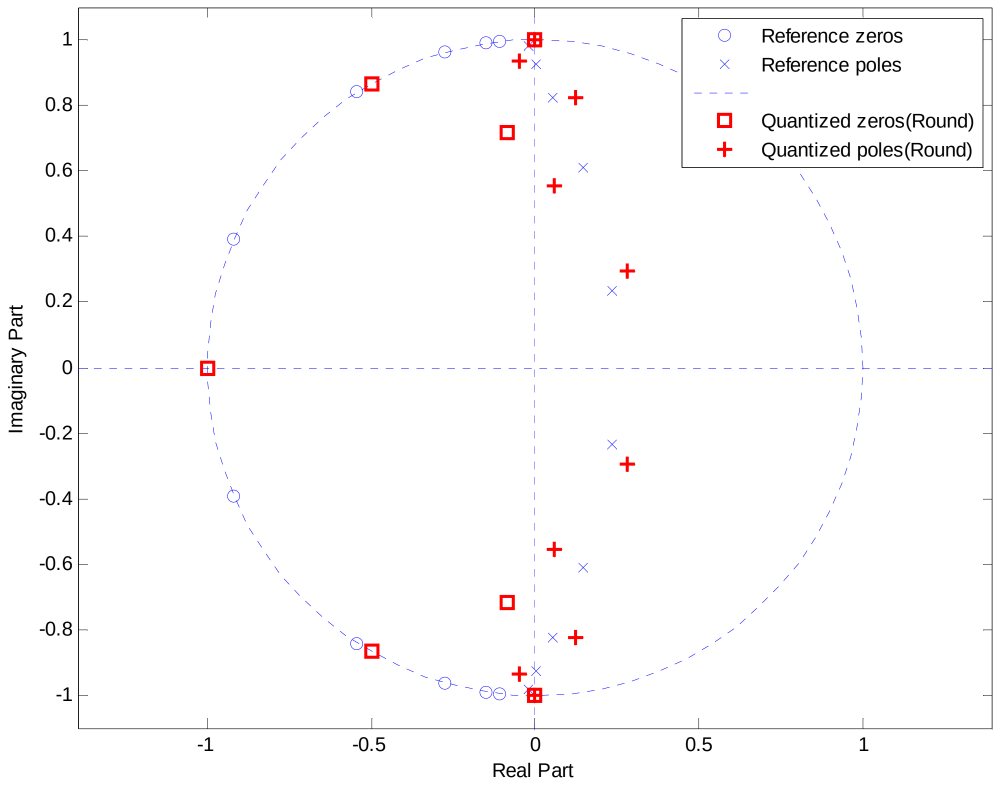


Figure 6. Original vs. 6-bit Quantized(Round)



3. **Realize the transfer function obtained above as a parallel connection of two allpass power-complementary filters (see Example 8.19 [pages 470-471] for information about how to derive the allpass transfer functions used in the construction of the power complementary filters). Can you verify that your allpass filters are allpass? Verify that there is low passband sensitivity by quantizing the allpass filters coefficients to 6 bits.**

3.1 Matlab Code

See Appendix C.

3.2 Results

Figure 7. Original vs. 6-bit Quantized of two allpass

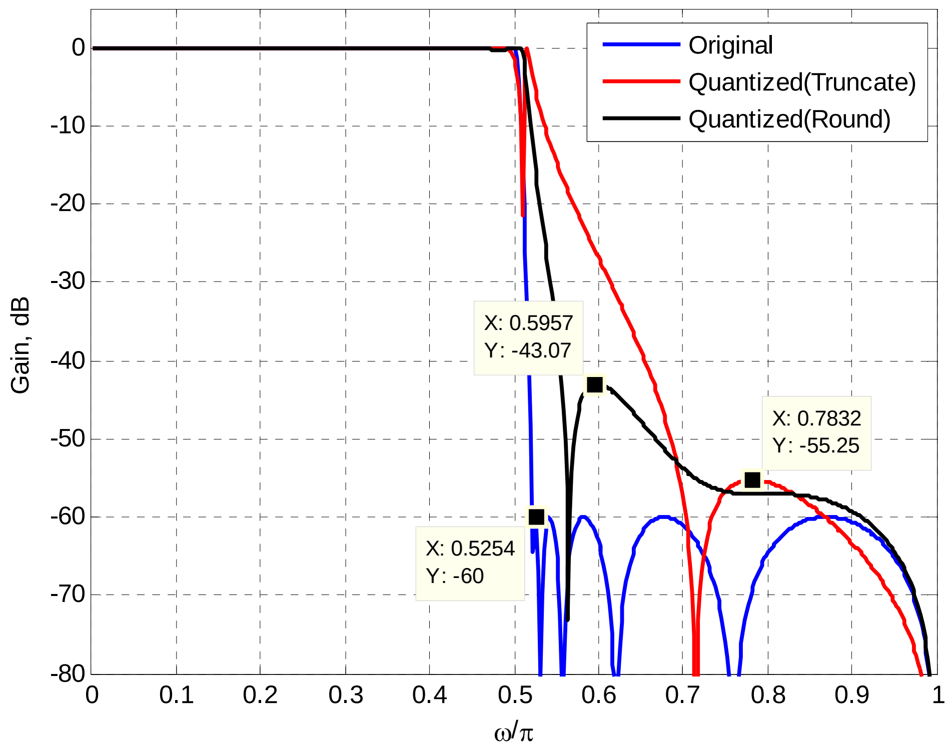


Figure 8.

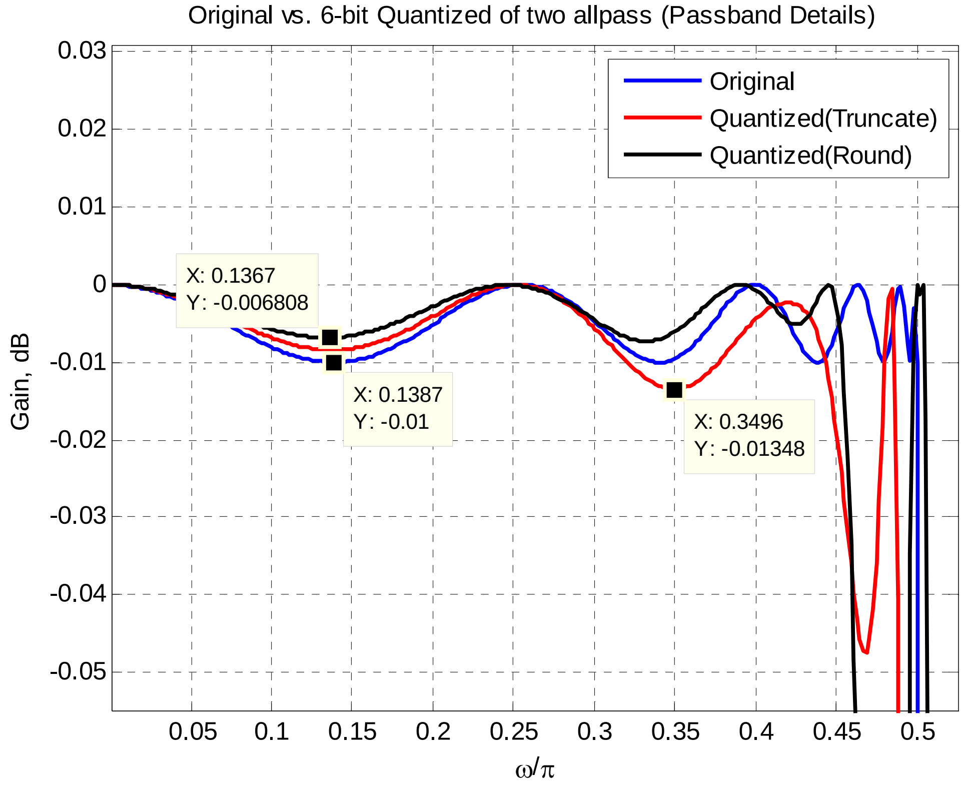


Figure 9.

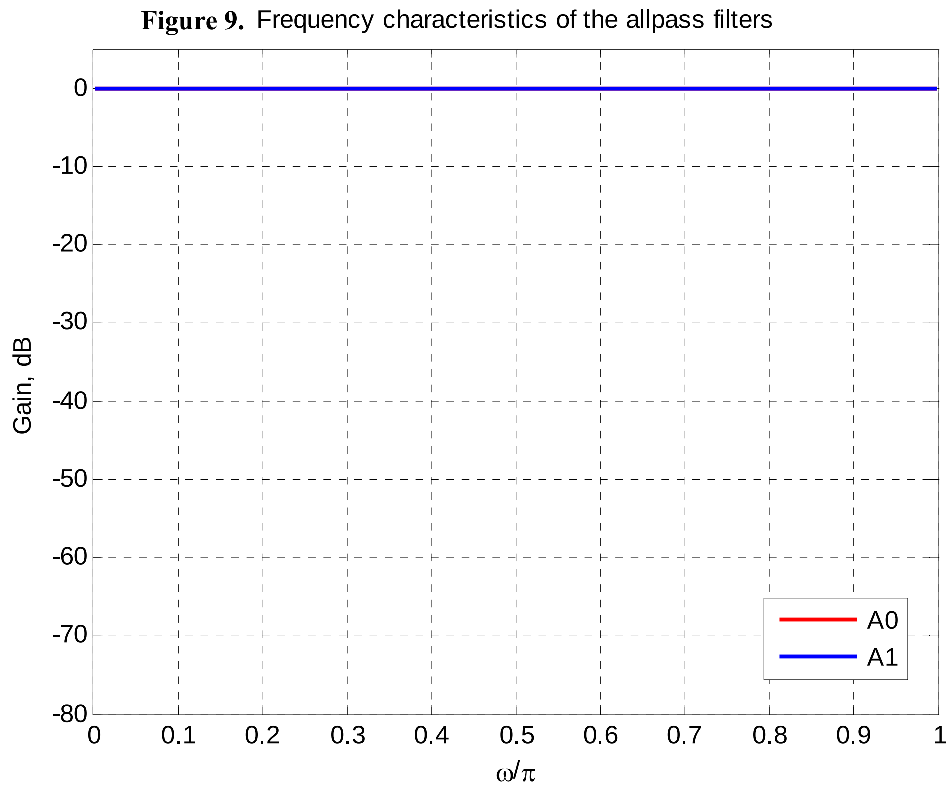
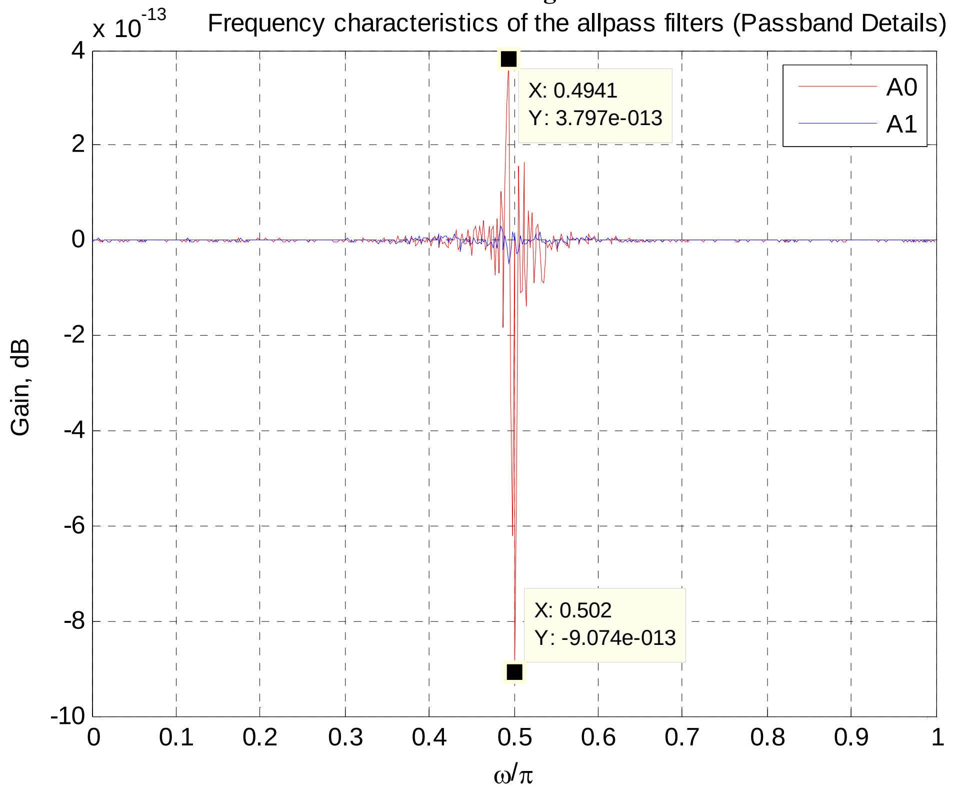


Figure 10.



4. EXTRA: For the original filter, what is the effect if you use cascaded first and second order sections instead of a single transfer function?

4.1 Matlab Code

See Appendix D.

4.2 Results

Figure 11.

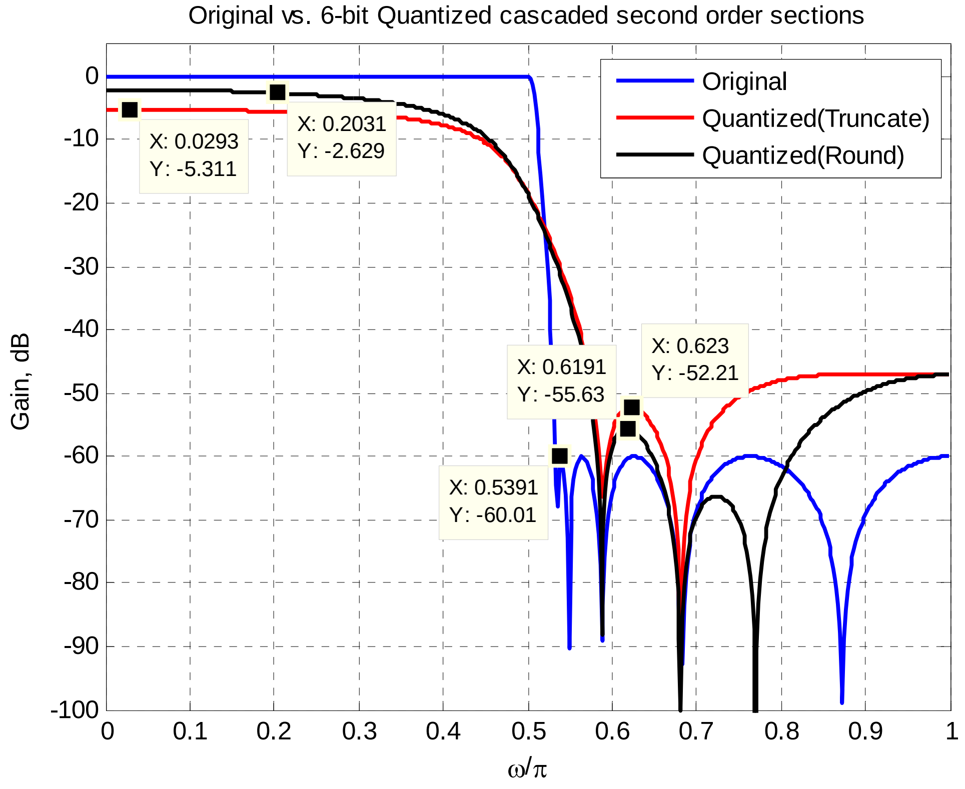
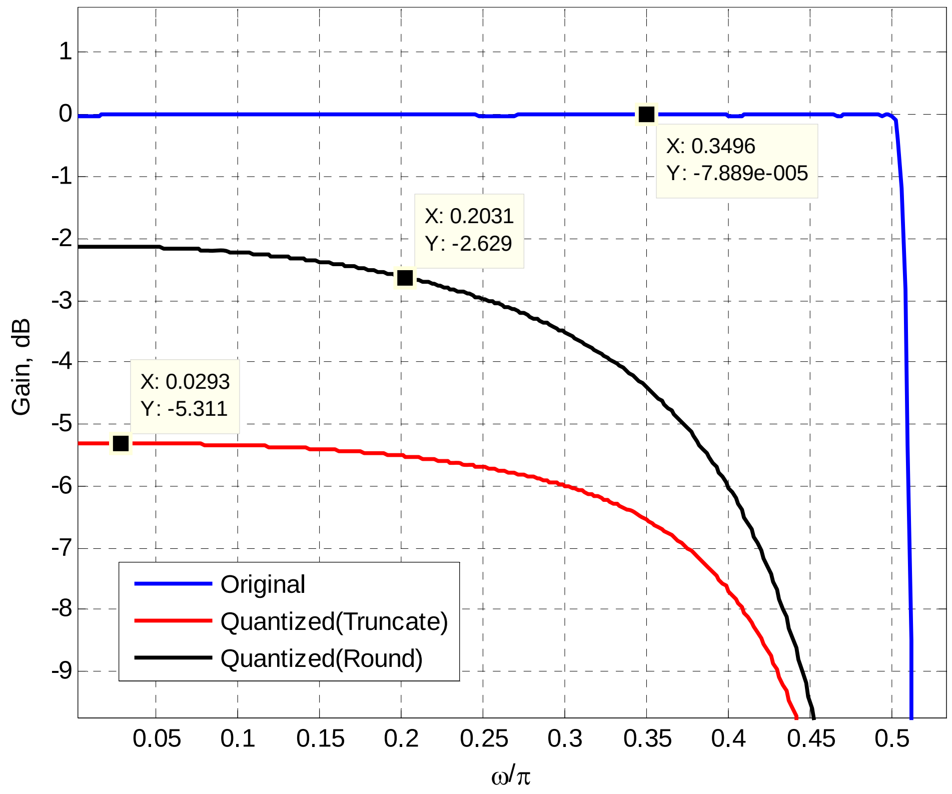


Figure 12. Original vs. 6-bit Quantized cascaded second order sections (Passband Details)



5. Results and Conclusions

Since Matlab uses double-precision decimal numbers (32-bit) and arithmetic, the filter coefficients and signals generated using Matlab can be considered to be of infinite precision for all practical purpose.

Figure 1 and 2 shows the gain response of the ideal filter with infinite precision coefficients (shown with blue line) and the gain response obtained when the transfer function coefficients are truncated to 6-bit length (shown with a red line) and also the gain response obtained when the transfer function coefficients are rounded to 6-bit length (shown with a black line). It can be seen from this figure that the effect of the coefficient quantization is more severe around the bandedges with a higher passband ripple and a smaller transition band. The minimum stopband attenuation has also become smaller. Moreover, the transmission zeros closest to the stopband edge have moved closer to the passband edge.

Figure 3 and 4 shows the gain response of the ideal filter with infinite precision coefficients (shown with blue line) and the gain response obtained when the transfer function coefficients are truncated to 16-bit length (shown with a red line) and also the gain response obtained when the transfer function coefficients are rounded to 6-bit length (shown with a black line). Since the precision was upgraded to 16-bit, both the truncated and rounded transfer functions are almost the same as the original one. It can be seen from Figure 4, there is just a small difference around the passband edge.

Figure 5 and 6 shows the locations of the poles and zeros of the original elliptic lowpass filter transfer function with unquantized coefficients and of the transfer function of the elliptic filter implemented with quantized coefficients. As can be seen from this plot, coefficient quantization can cause substantial displacement of the poles and zero from their desired nominal locations. In this case, the zero closest to the poles has moved the farthest from its original location and has moved closer to the new location of its nearest pole, which is now much closer th the unit circle.

Figure 7 and 8 shows the gain response of the parallel allpass realization with infinite precision coefficients (shown with blue line) and with quantized coefficients (red and black line). The low passband sensitivity properties of the parallel allpass structure are evident from this figure. Figure 9 and 10 shows the frequency characteristics of the allpass filters. This is to verify that the allpass filters are indeed allpass. And the power-complementary transfer function H(z) does not need to be computed in this case. Since low passband sensitivity of H(z) does not imply low stopband sensitivity of G(z).

Figure 10 and 11 is the answer to the extra problem. It can be seen from these figures that the effect of the coefficient quantization here is not as severe as in the previous case. A flat loss has been added the passband response with an increase in the passband ripple.

Here each complex zero-pair is realized by a second-order section, and hence, the zeros remain on the unit circle. In fact, all of the zeros remain pretty much at their original locations. The overall effect on the stopband response is minimal.

In general, a higher-order IIR transfer function should never be realized as a single direct form structure, but instead should be realized as a cascade of second-order and first-order sections to minimize the effect of coefficient quantization.

Appendix A

```
% ---------------------------------------------------------------
% Determine the transfer function for a lowpass elliptic
% filter, given the following filter specifications:
%        passband edge:                      0.5 * pi
%        passband ripple:                    0.01 dB
%        stopband edge:                      0.55 * pi
%        minimum stopband attenuation:       60 dB.
% ---------------------------------------------------------------
% ---------------------------------------------------------------
% Clear workspace & command window
% ---------------------------------------------------------------
clc;            % clear command window
clear;          % clear workplace
% ---------------------------------------------------------------
% calculate the order of the desired filter
% ---------------------------------------------------------------
Wp = 0.5;
Ws = 0.55;
Rp = 0.01;
Rs = 60;
[N, Wp] = ellipord(Wp, Ws, Rp, Rs);
% ---------------------------------------------------------------
% obtain the transfer function
% ---------------------------------------------------------------
[b, a] = ellip(N, Rp, Rs, Wp);
Transfer_Function = tf(b, a, -1, 'variable', 'z^-1');
```

Appendix B

```
% --------------------------------------------------------------
% Determine the frequency characteristics for the original
% filter without coefficient quantization. Modify Program
% 12_1 which uses the Matlab function a2dT developed
% by Mitra to quantize the coefficients to a word length
% of 6 bits.
% --------------------------------------------------------------
% --------------------------------------------------------------
% Clear workspace & command window
% --------------------------------------------------------------
clc;            % clear command window
clear;          % clear workplace
% --------------------------------------------------------------
% calculate the order of the desired filter
% --------------------------------------------------------------
Wp = 0.5;
Ws = 0.55;
Rp = 0.01;
Rs = 60;
[N, Wp] = ellipord(Wp, Ws, Rp, Rs);
% --------------------------------------------------------------
% obtain the transfer function
% --------------------------------------------------------------
[b, a] = ellip(10, Rp, Rs, Wp);
Transfer_Function = tf(b, a, -1, 'variable', 'z^-1');
% --------------------------------------------------------------
% truncate/round the filter coefficients to 6 bits
% --------------------------------------------------------------
bits = 6;
bqT = a2dT(b, bits);
aqT = a2dT(a, bits);
bqR = a2dR(b, bits);
aqR = a2dR(a, bits);
% --------------------------------------------------------------
% frequency characteristics
% --------------------------------------------------------------
[h, w] = freqz(b, a, 512);
[hqT, w] = freqz(bqT, aqT, 512);
[hqR, w] = freqz(bqR, aqR, 512);
```

```
g = 20*log10(abs(h));
gqR = 20*log10(abs(hqR));
gqT = 20*log10(abs(hqT));
% ------------------------------------------------------------------
% plot results
% ------------------------------------------------------------------
figure(1);
plot(w/pi, g, 'b', ...
     w/pi, gqT, 'r', ...
     w/pi, gqR, 'k', 'LineWidth', 2);
grid on;
axis([0 1 -100 5]);
xlabel('\omega/\pi');
ylabel('Gain, dB');
title('Original vs. 6-bit Quantized');
legend('Original', ...
     'Quantized(Truncate)', ...
     'Quantized(Round)');
% ------------------------------------------------------------------
% The call to the ZPLANE function below sets the axes
% properties to those corresponding to a pole-zero plot.
% ------------------------------------------------------------------
figure(2);
zplane(b, a);
hold on;
plotzp(bqT, aqT);
legend('Reference zeros', 'Reference poles', '', ...
     'Quantized zeros(Truncate)', 'Quantized poles(Truncate)');
title('Original vs. 6-bit Quantized(Truncate)');

figure(3);
zplane(b, a);
hold on;
plotzp(bqR, aqR);
legend('Reference zeros', 'Reference poles', '', ...
     'Quantized zeros(Round)', 'Quantized poles(Round)');
title('Original vs. 6-bit Quantized(Round)');
```

Appendix C

```
% -----------------------------------------------------------------
% Realize the transfer function obtained above as a parallel
% connection of two allpass power-complementary filters (see Example
% 8.19 [pages 470-471] for information about how to derive the
% allpass transfer functions used in the construction of the
% powercomplementary filters). Can you verify that your allpass
% filters are allpass? Verify that there is low passband sensitivity
% by quantizing the allpass filters coefficients to 6 bits.
% -----------------------------------------------------------------
% -----------------------------------------------------------------
% Clear workspace & command window
% -----------------------------------------------------------------
clc;            % clear command window
clear;          % clear workplace
% -----------------------------------------------------------------
% calculate the order of the desired filter
% -----------------------------------------------------------------
Wp = 0.5;
Ws = 0.55;
Rp = 0.01;
Rs = 60;
[N, Wp] = ellipord(Wp, Ws, Rp, Rs);
% -----------------------------------------------------------------
% obtain the transfer function
% -----------------------------------------------------------------
[b, a] = ellip(N + 1, Rp, Rs, Wp);
% -----------------------------------------------------------------
% obtain the allpass filter
% -----------------------------------------------------------------
[d0, d1] = tf2ca(b, a);
d0qT = a2dT(d0, 6);
d1qT = a2dT(d1, 6);
d0qR = a2dR(d0, 6);
d1qR = a2dR(d1, 6);
% -----------------------------------------------------------------
% reconstruct the original filter
% -----------------------------------------------------------------
num1 = 0.5 * (conv(fliplr(d1), d0) + conv(d1, fliplr(d0)));
den = conv(d0, d1);
```

```
num1qR = 0.5 * (conv(fliplr(d1qR), d0qR) + conv(d1qR, fliplr(d0qR)));
denqR = conv(d0qR, d1qR);
num1qT = 0.5 * (conv(fliplr(d1qT), d0qT) + conv(d1qT, fliplr(d0qT)));
denqT = conv(d0qT, d1qT);
[G1, w] = freqz(num1, den, 512);
[G1qR, w] = freqz(num1qR, denqR, 512);
[G1qT, w] = freqz(num1qT, denqT, 512);
% --------------------------------------------------------------
% plot results
% --------------------------------------------------------------
figure(1);
plot(w/pi, 20*log10(abs(G1)), '-b', ...
     w/pi, 20*log10(abs(G1qT)), '-r', ...
     w/pi, 20*log10(abs(G1qR)), '-k', ...
     'LineWidth', 2);
axis([0 1 -80 5]);
xlabel('\omega/\pi');
ylabel('Gain, dB'); grid on;
title('Original vs. 6-bit Quantized of two allpass');
legend('Original', ...
     'Quantized(Truncate)', ...
     'Quantized(Round)');
% --------------------------------------------------------------
% frequency characteristics of the allpass filters
% --------------------------------------------------------------
n0 = fliplr(d0);
n1 = fliplr(d1);
[A0, w] = freqz(n0, d0, 512);
[A1, w] = freqz(n1, d1, 512);
% --------------------------------------------------------------
% plot results
% --------------------------------------------------------------
figure(2);
plot(w/pi, 20*log10(abs(A0)), '-r', ...
     w/pi, 20*log10(abs(A1)), '-b', ...
     'LineWidth', 1);
% axis([0 1 -80 5]);
xlabel('\omega/\pi');
ylabel('Gain, dB'); grid on;
title('Frequency characteristics of the allpass filters');
legend('A0', 'A1');
```

Appendix D

```
% ------------------------------------------------------------------
% EXTRA - For the original filter, what is the effect if
% you use cascaded first and second order sections
% instead of a single transfer function?
% ------------------------------------------------------------------
% ------------------------------------------------------------------
% Clear workspace & command window
% ------------------------------------------------------------------
clc;            % clear command window
clear;          % clear workplace
% ------------------------------------------------------------------
% calculate the order of the desired filter
% ------------------------------------------------------------------
Wp = 0.5;
Ws = 0.55;
Rp = 0.01;
Rs = 60;
[N, Wp] = ellipord(Wp, Ws, Rp, Rs);
% ------------------------------------------------------------------
% obtain the transfer function
% ------------------------------------------------------------------
[z, p, k] = ellip(N, Rp, Rs, Wp);
% ------------------------------------------------------------------
% Coefficient Quantization Effects on the
% Frequency Response of a Cascade Form IIR Filter
% ------------------------------------------------------------------
[b, a] = zp2tf(z, p, k);
[h, w] = freqz(b, a, 512);
g = 20*log10(abs(h));

sos = zp2sos(z, p, k);
sosqT = a2dT(sos, 6);
R1T = sosqT(1, :);
R2T = sosqT(2, :);
R3T = sosqT(3, :);
b1T = conv(R1T(1:3), R2T(1:3));
bqT = conv(R3T(1:3), b1T);
a1T = conv(R1T(4:6), R2T(4:6));
aqT = conv(R3T(4:6), a1T);
```

```
[hqT, w] = freqz(bqT, aqT, 512);
gqT = 20*log10(abs(hqT));

sosqR = a2dR(sos, 6);
R1R = sosqR(1, :);
R2R = sosqR(2, :);
R3R = sosqR(3, :);
b1R = conv(R1R(1:3), R2R(1:3));
bqR = conv(R3R(1:3), b1R);
a1R = conv(R1R(4:6), R2R(4:6));
aqR = conv(R3R(4:6), a1R);
[hqR, w] = freqz(bqR, aqR, 512);
gqR = 20*log10(abs(hqR));

plot(w/pi, g, 'b', ...
     w/pi, gqT, 'r', ...
     w/pi, gqR, 'k', ...
     'LineWidth', 2);
grid on;
axis([0 1 -100 5]);
xlabel('\omega/\pi');
ylabel('Gain, dB');
title('Original vs. 6-bit Quantized');
legend('Original', ...
     'Quantized(Truncate)', ...
     'Quantized(Round)');
```