EECE5666 (DSP) : Homework-8 Solutions

Table of Contents

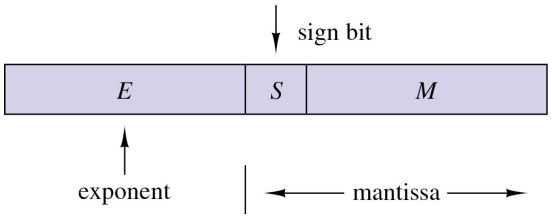
Default Plot Parameters

```
set(0,'defaultfigurepaperunits','inches','defaultfigureunits','inches');
set(0,'defaultaxesfontsize',10); set(0,'defaultaxeslinewidth',1.5);
set(0,'defaultaxestitlefontsize',1.2,'defaultaxeslabelfontsize',1.1);
```

Problem 8.1: Text Problem 9.24 (Page 657)

The following two parts (a) and (b) are not related.

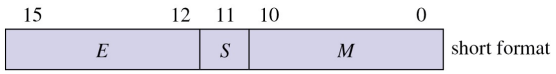
(a) The generic floating-point format for a DSP microprocessor is the following:



The value of the number X is given by

$$X = \begin{cases} 01.M \times 2^E, & \text{if } S = 0 \\ 10.M \times 2^E, & \text{is } S = 1 \\ 0, & \text{if } E \text{ is the most negative two's-complement value} \end{cases} \qquad (8.1.1)$$

[i] Determine the range of positive and negative numbers for the following format:



Solution: Note that in the 16-bit short word above, there are 11 bits in the mantissa M, 4 bits in the exponent E, and one sign bit S. Also note that from $(8.1.1)$, the most negative two's-complement value for 4-bit E is -8 which signifies the decimal value of 0. Hence the most negative two's-complement value for E we can use to compute non-negative short-format values is $E = -7 \equiv 1001$.

Largest positive number:

- We need largest M or $M = 01.\underbrace{111\ldots111}_{11} \equiv 1 + \sum_{n=1}^{11} 2^{-n} = 1 + 2^{-1}\sum_{n=0}^{10} 2^{-n} = 1 + 2^{-1}\frac{1-2^{-11}}{1-2^{-1}} = 1 + 1 - 2^{-11} = 2 - 2^{-11}$
- We need largest positive E or $E = 0111 \equiv 7$

Hence the largest positive number is $01.\underbrace{111\ldots111}_{11} \times 2^{0111} \equiv (2 - 2^{-11}) \times 2^7 = 255.9375$.

Smallest positive Number:

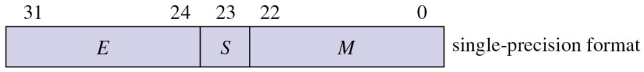
- We need smallest positive M or $M = 01.\underbrace{000\cdots000}_{11} \equiv 1$.
- We need smallest negative E or $E = -7$

Hence the smallest positive number is $01.\underbrace{000\ldots000}_{11} \times 2^{-7} \equiv 1 \times 2^{-7} = 0.0078125$. Therefore, the range of positive numbers is from

$$7.8125 \times 10^{-3} \text{ to } 2.559375 \times 10^{2}.$$

To achieve the maximum negative value, we need largest negative M and largest positve E, while for the minimum negative value we need smallest negative M and smallest negative E. Therefore, the maximum negative value is $10.\underbrace{000\ldots000}_{11} \times 2^{-7} \equiv -(2) \times 2^{-7} = -1.5625 \times 10^{-2}$, while the minimum is $10.\underbrace{111\ldots111}_{11} \times 2^{7} = -\left(2 + \sum_{n=1}^{11} 2^{-n}\right) \times 2^{7} = -3.839375 \times 10^{2}$. Therefore, the range for the negative numbers is from -3.839375×10^{2} to -1.5625×10^{-2}.

[ii] Determine the range of positive and negative numbers for the following format:



single-precision format

Solution: Note that in the 32-bit single-precision word above, there are 23 bits in the mantissa, 8 bits in the exponent E, and one sign bit S. Following the similar step in **[i]**, we have:

The maximum positive value is

$$\begin{aligned} 01.\underbrace{111\ldots111}_{23} \times 2^{127} &\equiv \left(1 + \sum_{n=1}^{23} 2^{-n}\right) \times 2^{127} = 1.999999940395355 \times 2^{127} \\ &= 3.402823567797336 \times 10^{38} \end{aligned},$$

while the minimum positive value is

$$01.\underbrace{000\ldots000}_{23} \times 2^{-127} = 1 \times 2^{-127} = 5.877471754111438 \times 10^{-39}.$$

In decimal, the range for the positive numbers is from $5.877471754111438 \times 10^{-39}$ to $3.402823567797336 \times 10^{38}$.

The maximum negative value is

$$10.\underbrace{000\ldots000}_{23}\times 2^{-127} \equiv -2\times 2^{-127} = -1.175494350822288\times 10^{-38}$$

while the minimum negative value is

$$10.\underbrace{111\ldots111}_{23}\times 2^{127} \equiv (-1)\times\left(2+1-\sum_{n=1}^{23}2^{-n}\right)\times 2^{127} = -5.104235300989981\times 10^{38}.$$

Therefore, the range is $-5.104235300989981\times 10^{38}$ to $-1.175494350822288\times 10^{-38}$.

(b) Determine the 10-bit sign-magnitude, one's-complement, and two's-complement representations of the following decimal numbers:

[i] $x_1 = 0.12345$, **[ii]** $x_2 = -0.56789$, **[iii]** $x_3 = 0.38452386$, **[iv]** $x_4 = -0.762349$, and **[v]** $x_5 = -0.90625$

Solution:

[i] All three representations are the same: 0.000111111

```
nbits = 10;
D = 0.12345;
B = dec2rep(D,nbits,'sign')
```

```
B = 1×10
     0     0     0     0     1     1     1     1     1     1
```

[ii] Sign-magnitude: 1.100100010, one's complement: 1.011011101, two's complement: 1.011011110

```
nbits = 10;
D = -0.56789;
B = dec2rep(D,nbits,'sign')
```

```
B = 1×10
     1     1     0     0     1     0     0     0     1     0
```

```
B = dec2rep(D,nbits,'one')
```

```
B = 1×10
     1     0     1     1     0     1     1     1     0     1
```

```
B = dec2rep(D,nbits,'two')
```

```
B = 1×10
     1     0     1     1     0     1     1     1     1     0
```

[iii] All three representations are the same: 0.011000100

```
nbits = 10;
D = 0.38452386;
B = dec2rep(D,nbits,'sign')
```

```
B = 1×10
     0     0     1     1     0     0     0     1     0     0
```

[iv] Sign-magnitude: 1.110000110, one's complement: 1.001111001, two's complement: 1.001111010

```
nbits = 10;
D = -0.762349;
B = dec2rep(D,nbits,'sign')
```

```
B = 1×10
     1     1     1     0     0     0     0     1     1     0
```

```
B = dec2rep(D,nbits,'one')
```

```
B = 1×10
```

```
     1     0     0     1     1     1     1     0     0     1
```

```
B = dec2rep(D,nbits,'two')
```

```
B = 1×10
     1     0     0     1     1     1     1     0     1     0
```

[v] Sign-magnitude: 1.111010000, one's complement: 1.000101111, two's complement: 1.000110000

```
nbits = 10;
D = -0.90625;
B = dec2rep(D,nbits,'sign')
```

```
B = 1×10
     1     1     1     1     0     1     0     0     0     0
```

```
B = dec2rep(D,nbits,'one')
```

```
B = 1×10
     1     0     0     0     1     0     1     1     1     1
```

```
B = dec2rep(D,nbits,'two')
```

```
B = 1×10
     1     0     0     0     1     1     0     0     0     0
```

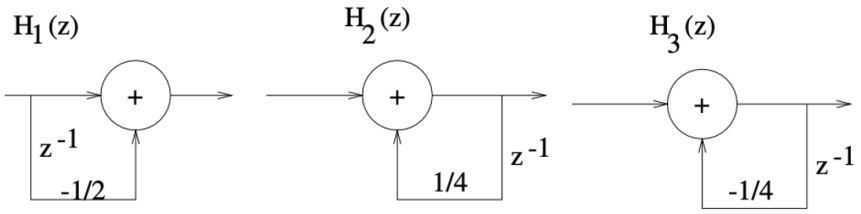
Problem 8.2: Text Problem 9.30 (Page 659)

Consider the system

$$H(z)=\frac{\left(1-\frac{1}{2}z^{-1}\right)}{\left(1-\frac{1}{4}z^{-1}\right)\left(1+\frac{1}{4}z^{-1}\right)}$$

(a) Draw all possible realizations of the system. **Hint**: Treat each first-order factor as a subsystem and consider six possible permutations of their cascade connections.

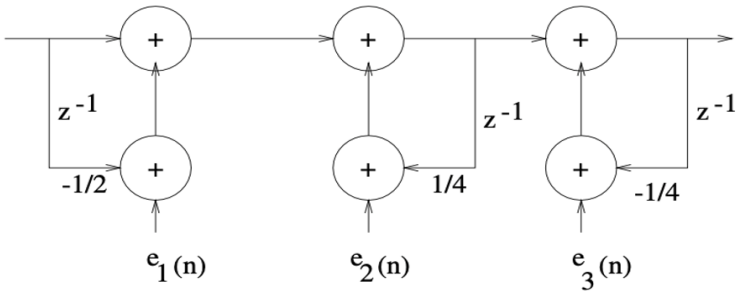
Drawings of realizations: The three first-order factors are: $H_1(z)=1-\frac{1}{2}z^{-1}$, $H_2(z)=\left(1-\frac{1}{4}z^{-1}\right)^{-1}$, and $H_3(z)=\left(1+\frac{1}{4}z^{-1}\right)^{-1}$



There are six possible cascade permutations:

$$H_1\to H_2\to H_3,\quad H_1\to H_3\to H_2,\quad H_2\to H_1\to H_3,\quad H_2\to H_3\to H_1,\quad H_3\to H_1\to H_2,\quad H_3\to H_2\to H_1.$$

One example of this realization, containing multiplication quantization error sources, is $H_1\to H_2\to H_3$:



The other realizations would be similar, but in different order of those first-order factors.

(b) Suppose that we implement the filter with fixed-point sign-and-magnitude fractional arithmetic using $(b+1)$ bits (one bit is used for the sign). Each resulting product is rounded into b bits. Determine the variance of the round-off noise created by the multipliers at the output of each one of the realizations in part (a).

Solution: For the case of $H_1\to H_2\to H_3$, we define $h(n)=h_2(n)*h_3(n)$, and have

$$q(n)=e_1(n)*h(n)+e_2(n)*h(n)+e_3(n)*h_3(n),$$

leading to

$$\sigma_q^2=\left(\sum_0^\infty h(n)^2+\sum_0^\infty h(n)^2+\sum_0^\infty h_3(n)^2\right)\sigma_e^2=\left(2\sum_0^\infty h(n)^2+\sum_0^\infty h_3(n)^2\right)\sigma_e^2$$

According to the problem, we have $h_2(n) = \left(\frac{1}{4}\right)^n u(n),\ h_3(n) = \left(-\frac{1}{4}\right)^n u(n)$, and

$h(n) = \left[1, 0, \frac{1}{16}, 0, \frac{1}{16^2}, \ldots\right]$.

Therefore, the variance is

$$\sigma_q^2 = \left(2\sum_0^\infty h(n)^2 + \sum_0^\infty h_3(n)^2\right)\sigma_e^2 = \left(2\frac{1}{1-\frac{1}{16^2}} + 2\frac{1}{1-\frac{1}{4^2}}\right)\sigma_e^2 = 3.0745\sigma_e^2.$$

For the case of $H_3 \to H_2 \to H_1$, we have

$$\begin{aligned} q(n) &= e_3(n) * h(n) * h_1(n) + e_2(n) * h_2(n) * h_1(n) + e_1(n) * \delta(n) \\ &= e_3(n) * \left(h(n) + \frac{1}{2}h(n-1)\right) + e_2(n) * \left(h_2(n) + \frac{1}{2}h_2(n-1)\right) + e_1(n) * \delta(n)' \end{aligned}$$

leading to

$$\sigma_q^2 = \left[\sum_0^\infty \left(h(n) + \frac{1}{2}h(n-1)\right)^2 + \sum_0^\infty \left(h_2(n) + \frac{1}{2}h_2(n-1)\right)^2 + 1\right]\sigma_e^2 = 3.3216\sigma_e^2$$

SImilarly, we can have:

$$H_1 \to H_3 \to H_2\colon\ \sigma_q^2 = 3.0745\sigma_e^2$$

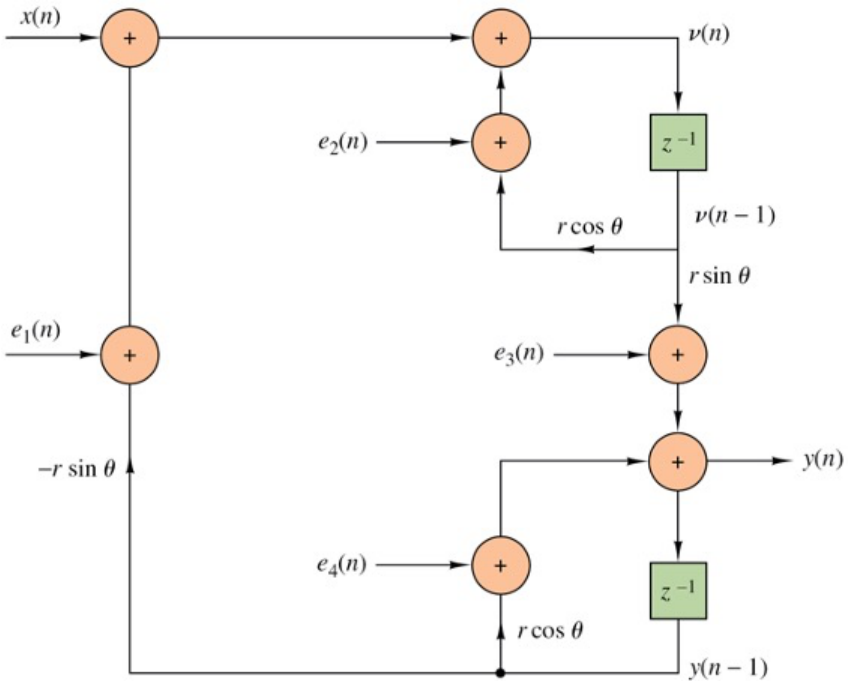
$$H_2 \to H_1 \to H_3\colon\ \sigma_q^2 = 3.3882\sigma_e^2$$

$$H_2 \to H_3 \to H_1\colon\ \sigma_q^2 = 3.2588\sigma_e^2$$

$$H_3 \to H_1 \to H_2\colon\ \sigma_q^2 = 3.2627\sigma_e^2$$

Problem 8.3: Text Problem 9.32 (Page 659)

Shown below is the coupled-form implementation of a two-pole filter with poles at $x = re^{\pm j\theta}$. There are four real multiplications per output point. Let $e_i(n),\ i = 1,\ 2,\ 3,\ 4$ represent the round-off noise in a fixed-point implementation of the filter. Assume that the noise sources are zero-mean mutually uncorrelated stationary white noise sequences. For each n the probability density function $p(e)$ is uniform in the range $-\Delta/2 \le e \le \Delta/2$, where $\Delta = 2^{-b}$.



(a) Write the two coupled difference equations for $y(n)$ and $v(n)$, including the noise sources and the input sequence $x(n)$.

Solution: Define $\rho_c = r\cos\theta,\ \rho_s = r\sin\theta$. Then we have:

$$v(n) = -\rho_s y(n-1) + e_1(n) + x(n) + \rho_c v(n-1) + e_2(n) \qquad (8.3.1)$$

$$y(n) = \rho_s v(n-1) + e_3(n) + \rho_c y(n-1) + e_4(n) \qquad (8.3.2)$$

(b) From these two difference equations, show that the filter system functions $H_1(z)$ and $H_2(z)$ between the input noise terms $e_1(n) + e_2(n)$ and $e_3(n) + e_4(n)$ and the output $y(n)$ are:

$$H_1(z) = \frac{r\sin(\theta)z^{-1}}{1 - 2r\cos(\theta)z^{-1} + r^2z^{-2}}$$

$$H_2(z) = \frac{1 - r\cos(\theta)z^{-1}}{1 - 2r\cos(\theta)z^{-1} + r^2z^{-2}}$$

We know that $H(z) = \dfrac{1}{1 - 2r\cos(\theta)z^{-1} + r^2z^{-2}} \quad \Rightarrow \quad h(n) = \dfrac{r^n}{\sin(\theta)}\sin[(n+1)\theta]u(n)$. Determine $h_1(n)$ and $h_2(n)$.

Solution: According to the difference equations in (a), we have:

$$V(z) = -\rho_s z^{-1}Y(z) + E_1(z) + X(z) + \rho_c z^{-1}V(z) + E_2(z)$$

$$Y(z) = \rho_s z^{-1}V(z) + E_3(z) + \rho_c z^{-1}Y(z) + E_4(z)$$

Combinating those two equations, we have

$$\begin{aligned} Y(z) &= \left(\frac{r\sin(\theta)z^{-1}}{1-2r\cos(\theta)z^{-1}+r^2z^{-2}}\right)\left[X(z)+E_1(z)+E_2(z)\right] \\ &\quad +\left(\frac{1-r\cos(\theta)z^{-1}}{1-2r\cos(\theta)z^{-1}+r^2z^{-2}}\right)\left[E_3(z)+E_4(z)\right] \\ &= H_1(z)X(z)+H_1\left[E_1(z)+E_2(z)\right]+H_2(z)\left[E_3(z)+E_4(z)\right] \qquad (8.3.3) \end{aligned}$$

With $H_1(z)$ and $H_2(z)$ verified as the problem stated, we have $h_1(n)$ and $h_2(n)$ following the Z-transform property:

$$\begin{aligned} h_1(n) &= r^n\sin(n\theta)u(n) \qquad (8.3.4) \\ h_2(n) &= r^n\cos(n\theta)u(n) \qquad (8.3.4) \end{aligned}$$

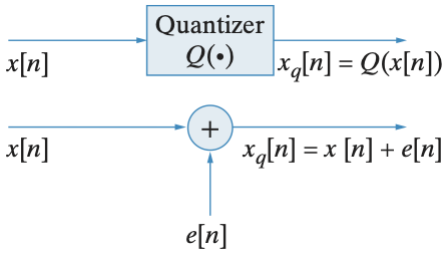
(c) Determine a closed-form expression for the variance of the total noise from $e_i(n)$, $i=1,\ 2,\ 3,\ 4$ at the output of the filter.

Solution: Assume $\sigma_e^2=\sigma_{e_i}^2=\dfrac{\Delta^2}{12}=\dfrac{2^{-2b}}{12}$, $\quad i=1,2,3,4$. Then from (8.3.3) through (8.3.5), we have

$$\begin{aligned} \sigma_q^2 &= \sigma_e^2\left[2\sum_0^\infty h_1(n)^2+2\sum_0^\infty h_2(n)^2\right] \\ &= \sigma_e^2\left[2\sum_0^\infty r^{2n}\sin^2(n\theta)+2\sum_0^\infty r^{2n}\cos^2(n\theta)\right] \\ &= \sigma_e^2\left[2\sum_0^\infty r^{2n}\underbrace{\left\{\sin^2(n\theta)+\cos^2(n\theta)\right\}}_{=1}\right]=\sigma_e^2\left[2\sum_0^\infty r^{2n}\right] \\ &= 2\sigma_e^2\left[\sum_0^\infty\left(r^2\right)^n\right]=\frac{2\sigma_e^2}{1-r^2}=\frac{2^{-2b}}{6(1-r^2)}=\frac{2^{-(2b+1)}}{3(1-r^2)} \end{aligned}$$

Problem 8.4: Statistical Analysis of Quantization Noise

Consider the following statistical quantization error (noise) model.



Under the condition 2 given on Page-642 of the textbook (P&M DSP Edition-4), the consecutive quantization noise $e[n]$ samples are uncorrelated. In this problem, we want to investigate under which conditions the consecutive noise samples, $e[n]$ and $e[n+1]$, are also statistically independent, that is, their joint probability density function (pdf) is a separable function of their marginal pdfs

$$f_{e(n),e(n+1)}(e_1, e_2) = f_{e(n)}(e_1) f_{e(n+1)}(e_2). \qquad (8.4.1)$$

```
clc; close all; clear;
newgr = [0.466,0.674,0.188]; % New green color
```

(a) Assume that $e(n)$ samples are statistically *independent and identically distributed* (IID) with uniform distribution over $\left[-\frac{\Delta}{2}, \frac{\Delta}{2}\right]$, that is,

$$f_{e(n)}(e_1) = f_{e(n+1)}(e_2) \triangleq f(e) = \mathcal{U}\left[-\frac{\Delta}{2}, \frac{\Delta}{2}\right], \qquad (8.4.2)$$

where Δ is the quantization interval. Let $e_0(n) = \frac{1}{2\Delta} e(n)$ be the normalized quantization noise sequence and let

$$g(n) = e_0(n) + e_0(n+1) \qquad (8.4.3)$$

be the sum of the consecutive normalized quantization noise samples. Show that the pdf, $f_G(g)$, of $g(n)$ is given by the triangular distribution

$$f_G(g) = 2\Lambda(2g) \qquad (8.4.4)$$

where

$$\Lambda(g) \triangleq \begin{cases} 1 - |g|, & |g| \le 1 \\ 0, & |g| > 1 \end{cases} \qquad (8.4.5)$$

is a triangular pulse between $-1 \le g \le 1$.

Solution: Since $e[n]$ is IID with pdf $f_E(e) \sim \mathcal{U}\left[-\frac{\Delta}{2}, \frac{\Delta}{2}\right]$, then the normalized sequence $e_0[n]$ is also IID with pdf, $f_{E_0}(e_0)$, which is obtained using the fundamental theorem of densities

$$f_{E_0}(e_0) = \left.\frac{f_E(e)}{de_0/de}\right|_{e=2\Delta e_0} = \frac{f_E(2\Delta e_0)}{1/(2\Delta)} = 2\Delta f_E(2\Delta e_0) \qquad (8.4.6)$$

But $f_E(e) = \mathcal{U}\left[-\frac{\Delta}{2},\frac{\Delta}{2}\right] = \begin{cases} 1/\Delta, & -\Delta/2 \le e \le \Delta/2 \\ 0, & \text{otherwise} \end{cases}$.

Substituting in (8.4.6), we obtain

$$f_{E_0}(e_0) = 2\Delta\left(\begin{cases} 1/\Delta, & -\Delta/2 \le 2\Delta e_0 \le \Delta/2 \\ 0, & \text{otherwise} \end{cases}\right)$$
$$= \begin{cases} 2, & -1/4 \le e_0 \le 1/4 \\ 0, & \text{otherwise} \end{cases} = \mathcal{U}\left[-\frac{1}{4},\frac{1}{4}\right]. \qquad (8.4.7)$$

Now, since $g[n] = e_0[n] + e_0[n+1]$, which is the sum of two IID random variables, the pdf of $g[n]$ is the convolution of the two identical uniform distributions. Thus,

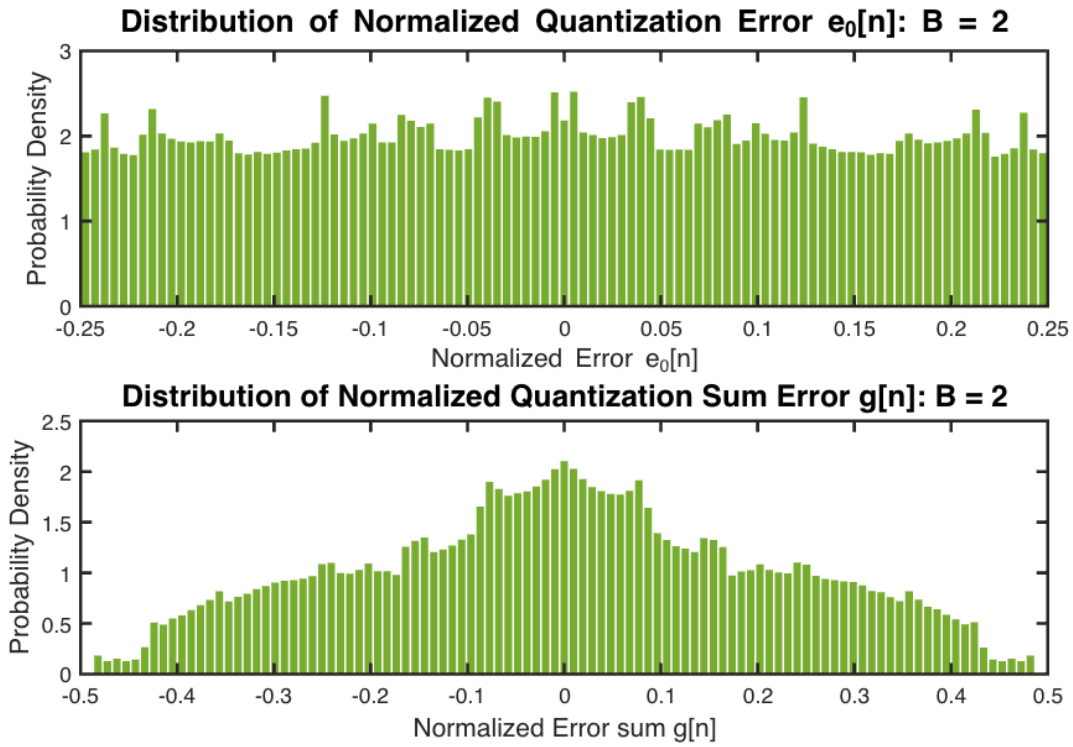
$$f_G(g) = f_{E_0}(g) * f_{E_0}(g) = \mathcal{U}\left[-\frac{1}{4},\frac{1}{4}\right] * \mathcal{U}\left[-\frac{1}{4},\frac{1}{4}\right]$$
$$= \begin{cases} 4\left(g+\frac{1}{4}+\frac{1}{4}\right), & -\frac{1}{2} \le g \le 0 \\ 4\left(\frac{1}{4}+\frac{1}{4}-g\right), & 0 \le g \le \frac{1}{2} \\ 0, & \text{else} \end{cases} = \begin{cases} 2+4g, & -\frac{1}{2} \le g \le 0 \\ 2-4g, & 0 \le g \le \frac{1}{2} \\ 0, & \text{else} \end{cases}$$
$$= \begin{cases} 2-4|g|, & |g| \le 1/2 \\ 0, & \text{else} \end{cases} = 2\begin{cases} 1-|2g|, & |2g| \le 1 \\ 0, & \text{else} \end{cases} = 2\Lambda(2g)$$

which proves (8.4.4).

(b) Let $x(n) = 0.5\left(\cos(n/7) + \sin(n/17)\right)$. Generate 500,000 samples of . Using the **dec2beqR** function quantize $x(n)$ to $B+1 = 3$ bits where B is the number of fractional bits. Obtain the sequences $e_0(n)$ and $g(n)$ using (8.2.3). Compute the histograms of both sequences using the **epdf** function and **bar** graph them in one figure with two rows and one column. Can you conclude that $e_0(n)$ is uniformly distributed and that its consecutive samples are statistically independent? Explain..

MATLAB script:

```
N = 500000; n = 1:N; xn = 0.5*(cos(n/7)+sin(n/17)); clear n;
B = 2; L = B+1;
[xq,~,~] = dec2beqR(xn,L); % Quantize the sequence
en = xq-xn; % Quantization noise;
Delta = 2^(-B); e0n = en/(2*Delta); % normalized quantization noise
gn = e0n(1:end-1)+e0n(2:end);
clear xn xq en; % Clear no longer needed large size vaiable to conserve memory
[e0,pe0]=epdf(e0n,101); % Histogram of e0n
[g,pg]=epdf(gn,101); % Histogram of gn
figure('Position',[0,0,8,5]*72);
subplot(2,1,1); % Normalized histogram of e0n
bar(e0,pe0,'FaceColor',newgr,'LineStyle','none'); axis([-0.25,0.25,0,3]);
xlabel('Normalized Error e_0[n]'); ylabel('Probability Density');
title('Distribution of Normalized Quantization Error e_0[n]: B = 2');
subplot(2,1,2); % Normalized histogram of gn
bar(g,pg,'FaceColor',newgr,'LineStyle','none'); axis([-0.5,0.5,0,2.5]);
xlabel('Normalized Error sum g[n]'); ylabel('Probability Density');
title('Distribution of Normalized Quantization Sum Error g[n]: B = 2');
```

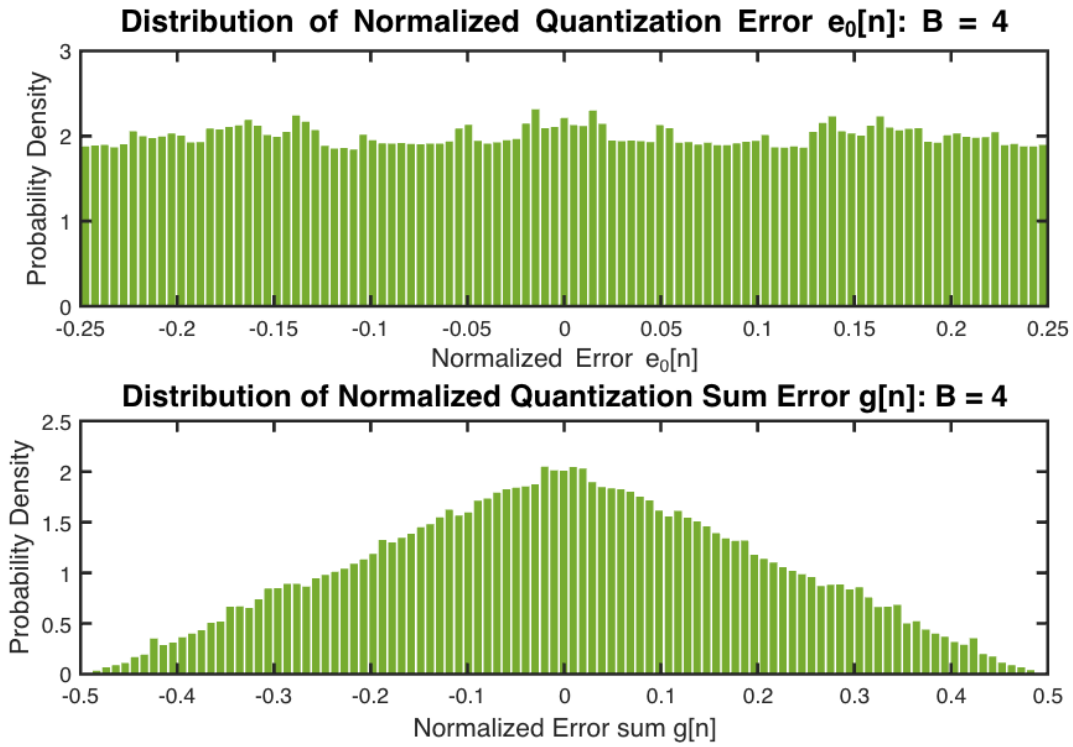


Explanation: The pdf of the normalized quantization error is not quite a uniform distribution while that of the sum of the consecutive samples is not triangular. This indicates that the error samples are not uniform and that the consecutive samples are not independent.

(c) Repeat part (b) using $B+1=5$ bits.

MATLAB script::

```
N = 500000; n = 1:N; xn = 0.5*(cos(n/7)+sin(n/17)); clear n;
B = 4; L = B+1;
[xq,~,~] = dec2beqR(xn,L); % Quantize the sequence
en = xq-xn; % Quantization noise;
Delta = 2^(-B); e0n = en/(2*Delta); % normalized quantization noise
gn = e0n(1:end-1)+e0n(2:end);
clear xn xq en; % Clear no longer needed large size vaiable to conserve memory
[e0,pe0]=epdf(e0n,101); % Histogram of e0n
[g,pg]=epdf(gn,101); % Histogram of gn
figure('Position',[0,0,8,5]*72);
subplot(2,1,1); % Normalized histogram of e0n
bar(e0,pe0,'FaceColor',newgr,'LineStyle','none'); axis([-0.25,0.25,0,3]);
xlabel('Normalized Error e_0[n]'); ylabel('Probability Density');
title('Distribution of Normalized Quantization Error e_0[n]: B = 4');
subplot(2,1,2); % Normalized histogram of gn
bar(g,pg,'FaceColor',newgr,'LineStyle','none'); axis([-0.5,0.5,0,2.5]);
xlabel('Normalized Error sum g[n]'); ylabel('Probability Density');
title('Distribution of Normalized Quantization Sum Error g[n]: B = 4');
```

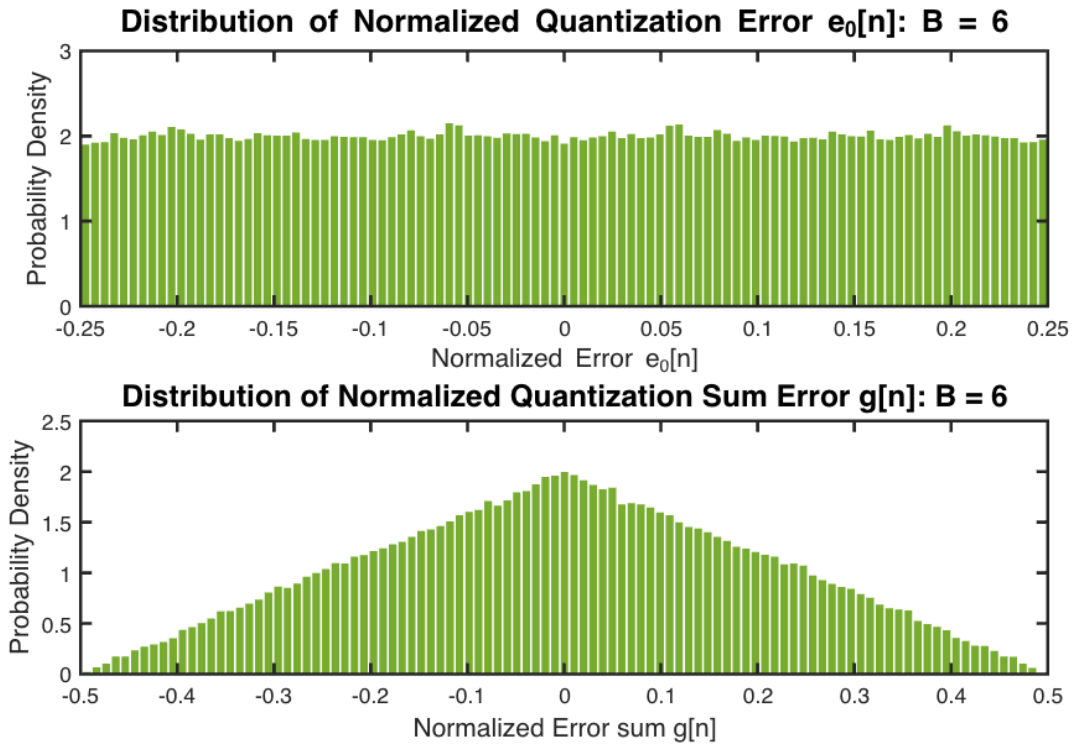


Explanation: The pdf of the normalized quantization error is very close (but not exact) to the uniform distribution while that of the sum of the consecutive samples is very close to a triangular distribution. This indicates that the consecutive samples are almost independent.

(d) Repeat part (b) using $B+1=7$ bits.

MATLAB script::

```
N = 500000; n = 1:N; xn = 0.5*(cos(n/7)+sin(n/17)); clear n;
B = 6; L = B+1;
[xq,~,~] = dec2beqR(xn,L); % Quantize the sequence
en = xq-xn; % Quantization noise;
Delta = 2^(-B); e0n = en/(2*Delta); % normalized quantization noise
gn = e0n(1:end-1)+e0n(2:end);
clear xn xq en; % Clear no longer needed large size vaiable to conserve memory
[e0,pe0]=epdf(e0n,101); % Histogram of e0n
[g,pg]=epdf(gn,101); % Histogram of gn
figure('Position',[0,0,8,5]*72);
subplot(2,1,1); % Normalized histogram of e0n
bar(e0,pe0,'FaceColor',newgr,'LineStyle','none'); axis([-0.25,0.25,0,3]);
xlabel('Normalized Error e_0[n]'); ylabel('Probability Density');
title('Distribution of Normalized Quantization Error e_0[n]: B = 6');
subplot(2,1,2); % Normalized histogram of gn
bar(g,pg,'FaceColor',newgr,'LineStyle','none'); axis([-0.5,0.5,0,2.5]);
xlabel('Normalized Error sum g[n]'); ylabel('Probability Density');
title('Distribution of Normalized Quantization Sum Error g[n]: B = 6');
```



Explanation: The pdf of the normalized quantization error is uniformly distributed while that of the sum of the consecutive samples is almost triangular. This indicates that the consecutive samples are independent.

(e) What is the minimum value of B for which you can assume that consecutive samples are independent?

Answer: Based on the above results, we can assume that when $B \geq 6$ fractional bits, the quantization noise is uniformly distributed and are independent of each other.

Problem 8.5: Coefficient Quantization of an Allpass Filter

Consider the filter described by the system function

$$H(z) = \frac{1-\sqrt{3}\,z^{-1}}{1-z^{-1}/\sqrt{3}}, \quad |z| > 1/\sqrt{3}. \qquad (8.5.1)$$

```
clc; close all; clear;
```

(a) Show that this filter is an all-pass filter. Plot its magnitude response over $-\pi \le \omega \le \pi$ and verify.

Solution: The frequency response from $(8.5.1)$ is given by

$$H(e^{j\omega}) = \frac{1-\sqrt{3}\,e^{-j\omega}}{1-\frac{1}{\sqrt{3}}e^{-j\omega}} = \frac{e^{j\omega}-\sqrt{3}}{e^{j\omega}-\frac{1}{\sqrt{3}}}$$

Hence

$$|H(e^{j\omega})| = \sqrt{\frac{(\cos\omega-\sqrt{3})^2+(\sin\omega)^2}{(\cos\omega-1/\sqrt{3})^2+(\sin\omega)^2}} = \sqrt{3}$$

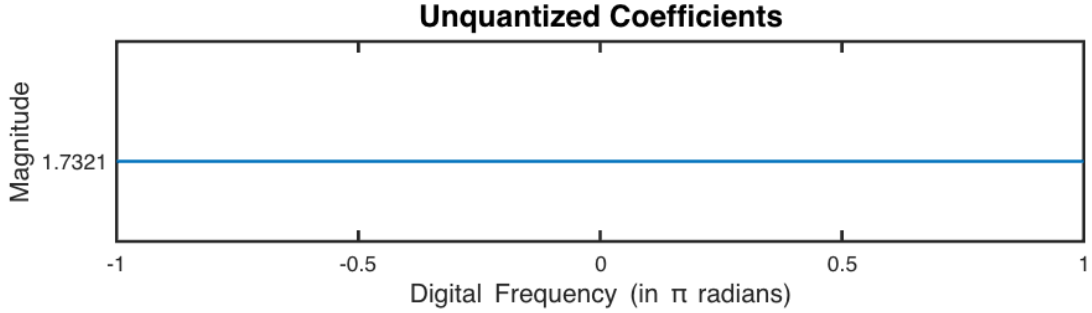
which is a constant for all ω. Hence the filter is an all-pass filter. Also note that the pole location $1/\sqrt{3}$ is the reciprocal of the zero location $\sqrt{3}$.

MATLAB verification:

```
b = [1,-sqrt(3)], a = [1,-1/sqrt(3)],
```

```
b = 1×2
    1.0000   -1.7321
a = 1×2
    1.0000   -0.5774
```

```
f = linspace(-1,1,1001); H = freqz(b,a,pi*f); Hmag = abs(H);
figure('position',[0,0,8,2]*72); plot(f,Hmag,'linewidth',1.5);
axis([-1,1,1.71,1.765]); ylabel('Magnitude');
title('Unquantized Coefficients');
xlabel('Digital Frequency (in \pi radians)');
set(gca,'ytick',sqrt(3),'xtick',(-1:0.5:1));
```



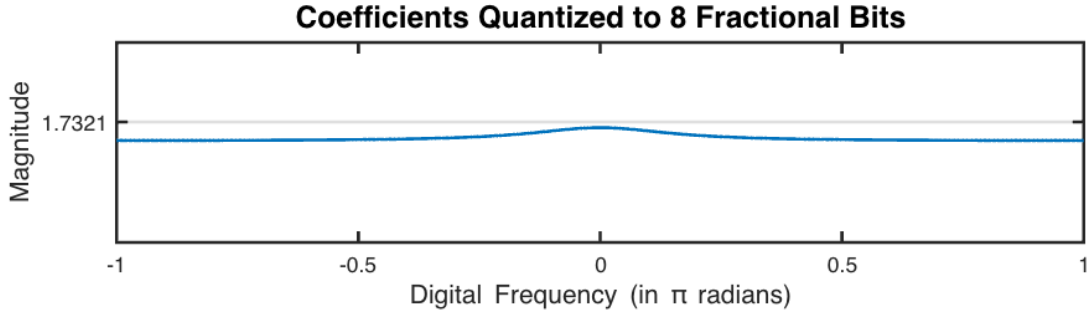
(b) Round the filter coefficients to $B = 8$ fraction bits using the **dec2beqR** function and then plot the magnitude response of the resulting filter. Is the filter still all-pass?

Solution: Note from the above filter coefficients that we will need one integer bit. Since we want 8 fractional bits, the total number of bits is $L = 1 + 1 + 8 = 10$.

```
L = 10; [bahat,~,~] = dec2beqR([b;a],L)
```

```
bahat = 2×2
    1.0000   -1.7305
    1.0000   -0.5781
```

```
bhat = bahat(1,:); ahat = bahat(2,:);
H = freqz(bhat,ahat,pi*f); Hmag = abs(H);
figure('position',[0,0,8,2]*72); plot(f,Hmag,'linewidth',1.5);
axis([-1,1,1.72,1.74]); ylabel('Magnitude');
title('Coefficients Quantized to 8 Fractional Bits');
xlabel('Digital Frequency (in \pi radians)');
set(gca,'ytick',sqrt(3),'xtick',(-1:0.5:1),'ygrid','on');
```



Observe that the magnitude response is not constant and hence the resulting filter is not an allpass system. Also pole location 0.5781 is not reciprocal of the zero location 1.7305.

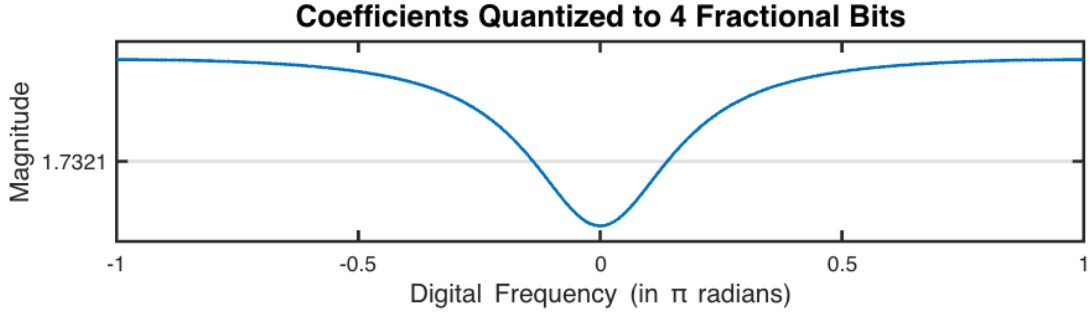
(c) Round the filter coefficients to $B = 4$ fractional bits using the **`dec2beqR`** function and then plot the magnitude response of the resulting filter. Is the filter still all-pass?

Solution: Note from the above filter coefficients that we will need one integer bit. Since we want 4 fractional bits, the total number of bits is $L = 1 + 1 + 4 = 6$.

```
L = 6; [bahat,~,~] = dec2beqR([b;a],L)
```

```
bahat = 2×2
    1.0000   -1.7500
    1.0000   -0.5625
```

```
bhat = bahat(1,:); ahat = bahat(2,:);
H = freqz(bhat,ahat,pi*f); Hmag = abs(H);
figure('position',[0,0,8,2]*72); plot(f,Hmag,'linewidth',1.5);
axis([-1,1,1.71,1.765]); ylabel('Magnitude');
title('Coefficients Quantized to 4 Fractional Bits');
xlabel('Digital Frequency (in \pi radians)');
set(gca,'ytick',sqrt(3),'xtick',(-1:0.5:1),'ygrid','on');
```



Observe that the magnitude response is certainly not constant and hence the resulting filter is not an allpass system. Also pole location 0.5625 is not reciprocal of the zero location 1.75.

Problem 8.6: Coefficient Quantization of a Bandpass IIR Filter

Design a digital bandpass filter using the elliptic prototype to satisfy the requirements:

$$\omega_{s_1} = 0.25\pi,\ \omega_{p_1} = 0.3\pi,\ \omega_{p_2} = 0.5\pi,\ \omega_{s_2} = 0.6\pi,\ A_p = 0.5 \text{ dB, and } A_s = 60 \text{ dB}.$$

```
clc; close all; clear;
```

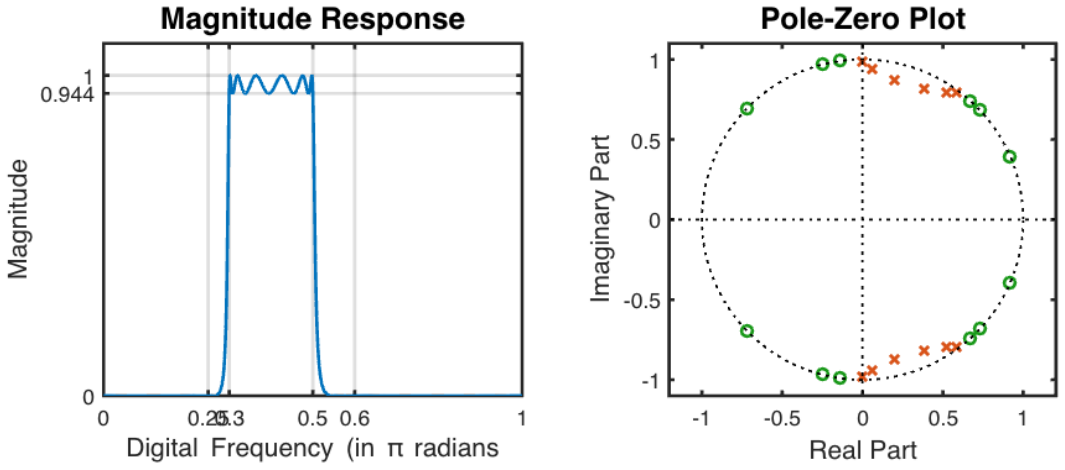
(a) Plot the magnitude response over $0 \le \omega \le \pi$ and pole-zero diagram of the filter.

Solution: The required filter designed and its responses are plotted using the following script.

```
% Given Specifications
fs1 = 0.25; fp1 = 0.3; fp2 = 0.5; fs2 = 0.6; Ap = 0.5; As = 60;
fp = [fp1,fp2]; fs = [fs1,fs2];
[N,fc] = ellipord(fp,fs,Ap,As); N
```

```
N = 6
```

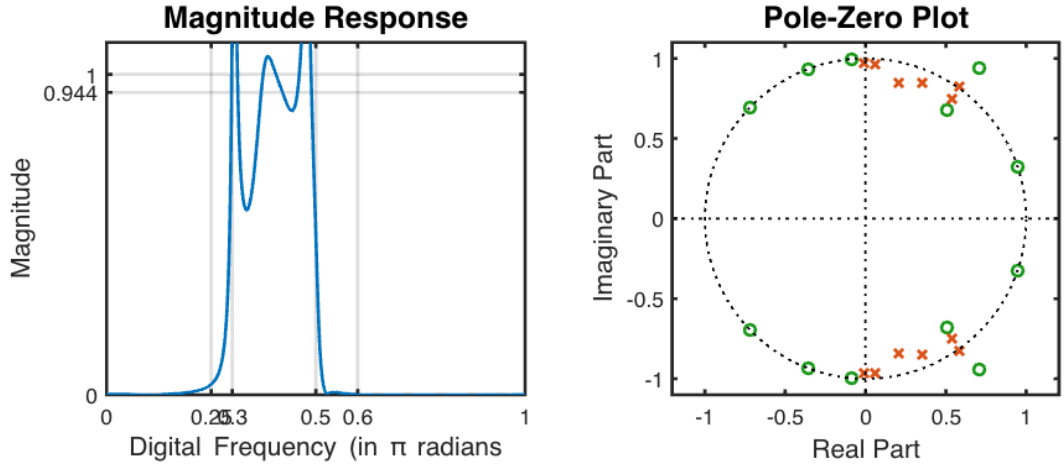
```
[b,a] = ellip(N,Ap,As,fc); % Direct form coefficients
f = linspace(0,1,1001); H = freqz(b,a,f*pi); Hmag = abs(H);
fcutoff = [0,fs1,fp1,fp2,fs2,1];
[epsi,~] = spec_convert(Ap,As,'rel','ana');
PBripple = round(sqrt(1/(1+epsi^2)),3);
% Filter Response Plots
figure('position',[0,0,8,3]*72);
subplot(1,2,1); % Magnitude response
plot(f,Hmag,'linewidth',1.5); axis([0,1,0,1.1]);
xlabel('Digital Frequency (in \pi radians'); ylabel('Magnitude');
title('Magnitude Response'); set(gca,'xtick',fcutoff);
set(gca,'ytick',[0,PBripple,1]); grid;
subplot(1,2,2); % Pole-zero plot
[Hz,Hp,Hl] = zplane(b,a); title('Pole-Zero Plot');
set(Hz,'markersize',5,'linewidth',1.5,'color',[0.1,0.6,0.1]);
set(Hp,'markersize',6,'linewidth',1.5,'color',[0.85,0.325,0.098]);
set(Hl,'linewidth',1,'color','k');
```



(b) Quantize the direct form coefficients to $L = 16$ bits using the **`dec2beqR`** function and plot the resulting magnitude response and pole-zero diagram.

Solution: We will quantize both **b** and **a** array together using matrix representation **[b;a]**, extract quantized arrays, and then obtain new plots. The following script performs all steps.

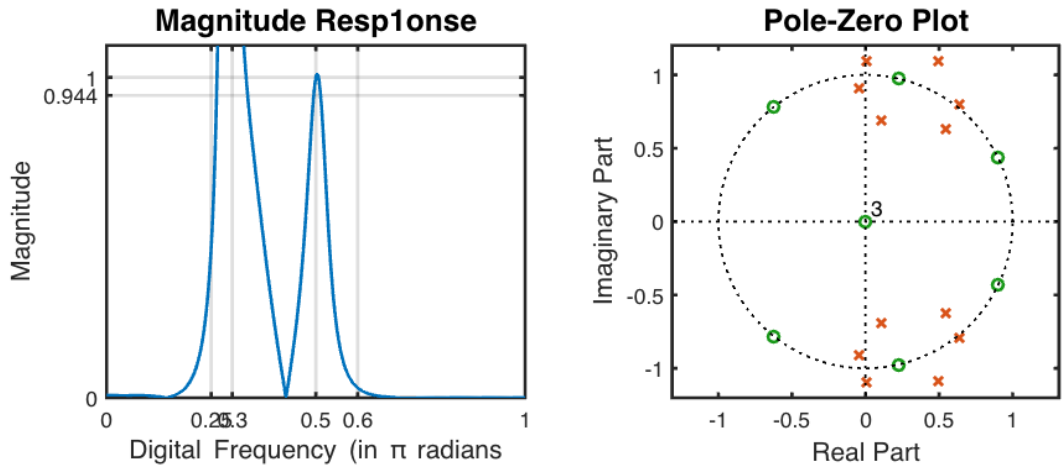
```
L = 16; [bahat,~,~] = dec2beqR([b;a],L);
bhat = bahat(1,:); ahat = bahat(2,:);
Hhat = freqz(bhat,ahat,f*pi); Hhatmag = abs(Hhat);
% Filter Response Plots
figure('position',[0,0,8,3]*72);
subplot(1,2,1); % Magnitude response
plot(f,Hhatmag,'linewidth',1.5); axis([0,1,0,1.1]);
xlabel('Digital Frequency (in \pi radians'); ylabel('Magnitude');
title('Magnitude Response'); set(gca,'xtick',fcutoff);
set(gca,'ytick',[0,PBripple,1]); grid;
subplot(1,2,2); % Pole-zero plot
[Hz,Hp,Hl] = zplane(bhat,ahat); title('Pole-Zero Plot');
set(Hz,'markersize',5,'linewidth',1.5,'color',[0.1,0.6,0.1]);
set(Hp,'markersize',6,'linewidth',1.5,'color',[0.85,0.325,0.098]);
set(Hl,'linewidth',1,'color','k');
```



(c) Quantize the direct form coefficients to $L = 12$ bits using the **dec2beqR** function and plot the resulting magnitude response and pole-zero diagram.

Solution: We repeat the above procedure using $L = 12$ bits.

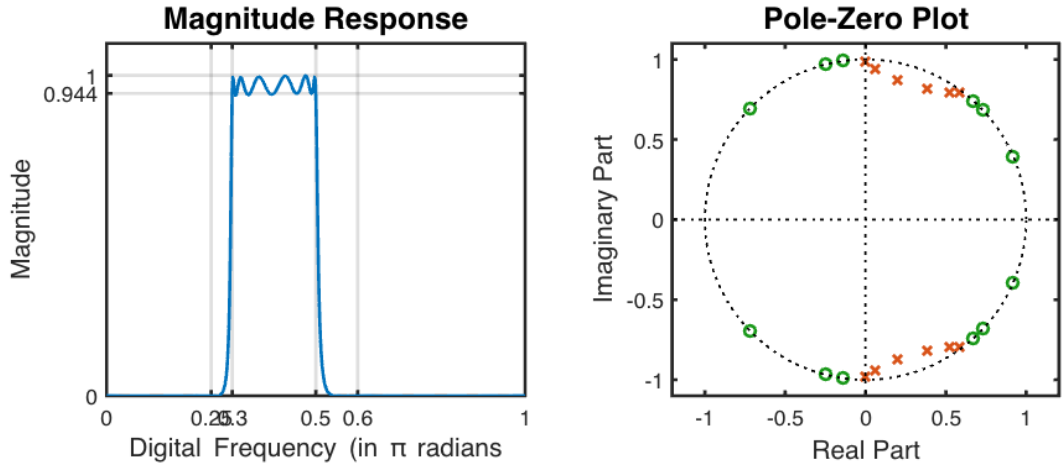
```
L = 12; [bahat,~,~] = dec2beqR([b;a],L);
bhat = bahat(1,:); ahat = bahat(2,:);
Hhat = freqz(bhat,ahat,f*pi); Hhatmag = abs(Hhat);
% Filter Response Plots
figure('position',[0,0,8,3]*72);
subplot(1,2,1); % Magnitude response
plot(f,Hhatmag,'linewidth',1.5); axis([0,1,0,1.1]);
xlabel('Digital Frequency (in \pi radians'); ylabel('Magnitude');
title('Magnitude Resp1onse'); set(gca,'xtick',fcutoff);
set(gca,'ytick',[0,PBripple,1]); grid;
subplot(1,2,2); % Pole-zero plot
[Hz,Hp,Hl] = zplane(bhat,ahat); title('Pole-Zero Plot');
set(Hz,'markersize',5,'linewidth',1.5,'color',[0.1,0.6,0.1]);
set(Hp,'markersize',6,'linewidth',1.5,'color',[0.85,0.325,0.098]);
set(Hl,'linewidth',1,'color','k');
```



(d) Quantize the cascade form coefficients to $L = 12$ bits using the **`dec2beqR`** function and plot the resulting magnitude response and pole-zero diagram.

Solution: We first convert direct form coefficients into cascade form coefficents, quantize them to $L = 12$ bits, convert quantized cascade coefficients into direct form (because we do not have a function equivalent to **`freqz`** using cascade coefficients), and then repeat the procedure of part (c). The following script performs all these steps.

```
[sos,G] = tf2sos(b,a); % cascade form coefficients
L = 12; [soshat,E3,B3] = dec2beqR([sos;[G,zeros(1,5)]],L);
Ghat = soshat(end,1); soshat = soshat(1:end-1,:);
[bhat,ahat] = sos2tf(soshat,Ghat);
Hhat = freqz(bhat,ahat,f*pi); Hhatmag = abs(Hhat/max(Hhat));
% Filter Response Plots
figure('position',[0,0,8,3]*72);
subplot(1,2,1); % Magnitude response
plot(f,Hhatmag,'linewidth',1.5); axis([0,1,0,1.1]);
xlabel('Digital Frequency (in \pi radians'); ylabel('Magnitude');
title('Magnitude Response'); set(gca,'xtick',fcutoff);
set(gca,'ytick',[0,PBripple,1]); grid;
subplot(1,2,2); % Pole-zero plot
[Hz,Hp,Hl] = zplane(bhat,ahat); title('Pole-Zero Plot');
set(Hz,'markersize',5,'linewidth',1.5,'color',[0.1,0.6,0.1]);
set(Hp,'markersize',6,'linewidth',1.5,'color',[0.85,0.325,0.098]);
set(Hl,'linewidth',1,'color','k');
```



(e) Comment on your plots.

Comments: The magnitude plots from parts (b) and (c) indicate that direct form implementations using even 16 bits give filters that fail miserably in satisfying specifications. The pole-zero plots indicate that we certainly get an unstable filter in (c) and a possibly unstable filter in (b). The cascade form implementation, on the other hand, give us a desirable and stable filter even for 12 bits.

Problem 8.7: Coefficient Quantization of a Bandpass FIR Filter

Consider the design of a 32-length linear-phase FIR bandpass filter that satisfies requirements of 60-dB stopband attenuation, lower stopband edge-frequency $\omega_{s_1} = 0.2\pi$, and upper stopband edge-frequency $\omega_{s_2} = 0.8\pi$.

```
clc; close all; clear;
```

(a) Using the Parks-McClellan algorithm, design the above mentioned FIR bandpass filter. Since no other requirements are given, you are free to choose any additional needed parameters. The resulting filter impulse response $h[n]$ can be considered to have infinite precision.

Solution: Note that the passband edge frequencies are not given. However, order of the filter is known. We also know that there is a relationship between transition bandwidth and order of the filter. Hence it is possible to determine passband edge frequencies iteratively. Since no other information is available, we are free to make few choices. We will assume that two transition bandwidths (upper and lower) are equal and that the ripples in each band are also equal. Then, starting with an assumed transition bandwidth $\Delta f = \Delta\omega/\pi$, we will compute $f_{p_1} = f_{s_1} + \delta f$ and $f_{p_2} = f_{s_2} - \delta f$, execute the **`firpm`** function, and note the resulting δ. If $\delta > \delta_s$ we will increase Δf, otherwise decrease Δf.We will iterate until $\delta \leq \delta_s$ to obtain the desired filter impulse response. Since we assumed that $\delta_s = \delta_p$ we do not need a weighting function. Also note that since $\delta_s = \delta_p$, we have

$$A_s = -20\log_{10}\left(\frac{\delta_s}{1+\delta_s}\right) \quad \Rightarrow \quad \delta_s = \frac{10^{-A_s/20}}{1-10^{-A_s/20}}.$$

```
fs1 = 0.2; fs2 = 0.8; As = 60; % Given specifications
L = 32; M = L-1; % Given Order of the filter
deltas = 10^(-As/20)/(1-10^(-As/20)); % Stopband ripple
deltaf = 0.2; % Assumed value
fp1 = fs1+deltaf; fp2 = fs2-deltaf;
fo = [0,fs1,fp1,fp2,fs2,1]; ao = [0,0,1,1,0,0];
[~,delta] = firpm(M,fo,ao);
fprintf('Required ripple: %g, Obtained ripple: %g',deltas,delta);
```

```
Required ripple: 0.001001, Obtained ripple: 0.0017409
```

```
% Hence we need to increase deltaf. After few iterations deltaf = 0.2115
deltaf = 0.2115; % Value after iteration
fp1 = fs1+deltaf; fp2 = fs2-deltaf;
fo = [0,fs1,fp1,fp2,fs2,1]; a0 = [0,0,1,1,0,0];
[h,delta] = firpm(M,fo,ao);
fprintf('Required ripple: %g, Obtained ripple: %g',deltas,delta);
```

```
Required ripple: 0.001001, Obtained ripple: 0.000991197
```

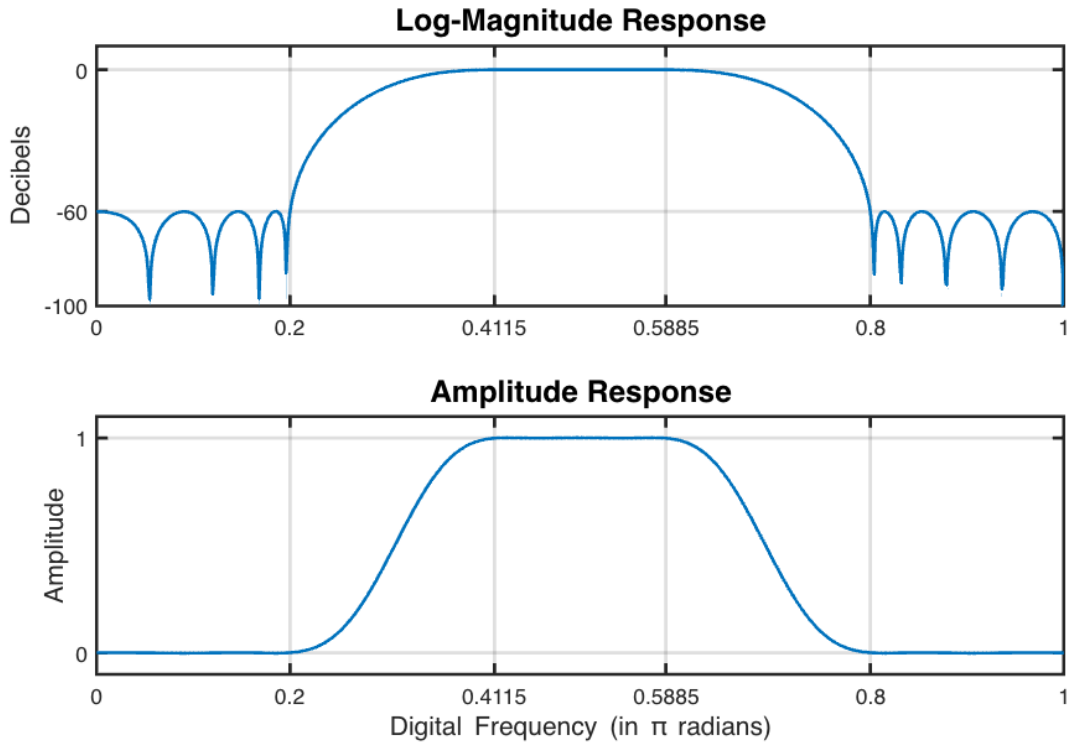
Now we have the required filter impulse response.

(b) Using infinite precision, plot the log-magnitude and amplitude responses of the designed filter. Use 2 rows and 1 column of subplots.

Solution: The following script plots the responses

```
f = linspace(0,1,1001); H = zerophase(h,1,f*pi); Hmag = abs(H);
Hdb = 20*log10(Hmag/max(Hmag));
figure('position',[0,0,8,5]*72);
subplot(2,1,1); % Log-magnitude plot
```

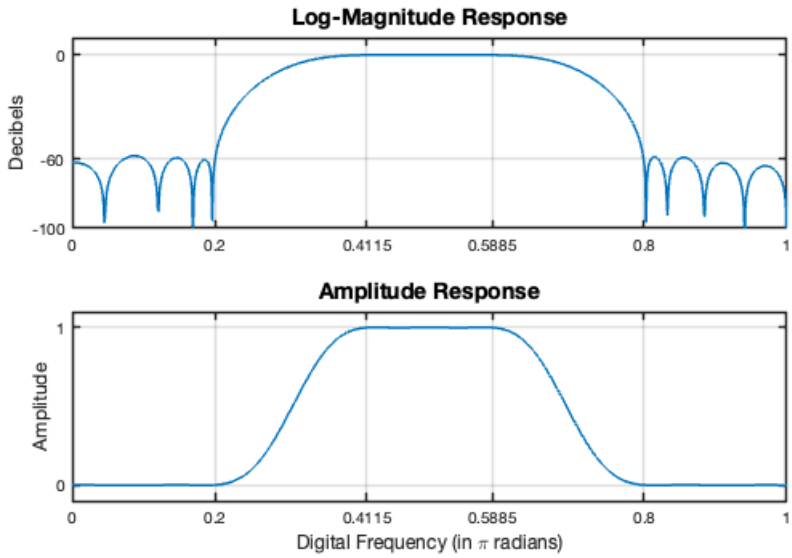
```
plot(f,Hdb,'linewidth',1.5); axis([0,1,-100,10]);
ylabel('Decibels'); title('Log-Magnitude Response');
set(gca,'xtick',fo,'ytick',[-100,-As,0]); grid;
subplot(2,1,2); % Amplitude plot
plot(f,H,'linewidth',1.5); axis([0,1,-0.1,1.1]);
ylabel('Amplitude'); title('Amplitude Response');
xlabel('Digital Frequency (in \pi radians)');
set(gca,'xtick',fo,'ytick',[0,1]); grid;
```



(c) Quantize the direct-form coefficients to 4 decimals (by rounding). Now plot the log-magnitude and amplitude responses of the resulting filter. Use 2 rows and 1 column of subplots.

Solution: The following script plots the responses

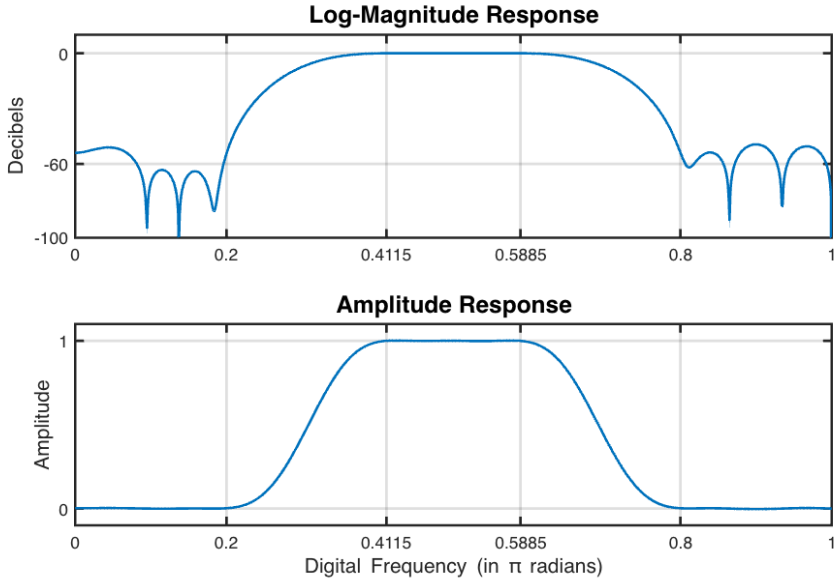
```
D = 4; h1 = round(h*10^D)/10^D; %Quantized to 4 decimals
H = zerophase(h1,1,f*pi); Hmag = abs(H);
Hdb = 20*log10(Hmag/max(Hmag));
figure('position',[0,0,8,5]*72);
subplot(2,1,1); % Log-magnitude plot
plot(f,Hdb,'linewidth',1.5); axis([0,1,-100,10]);
ylabel('Decibels'); title('Log-Magnitude Response');
set(gca,'xtick',fo,'ytick',[-100,-As,0]); grid;
subplot(2,1,2); % Amplitude plot
plot(f,H,'linewidth',1.5); axis([0,1,-0.1,1.1]);
ylabel('Amplitude'); title('Amplitude Response');
xlabel('Digital Frequency (in \pi radians)');
set(gca,'xtick',fo,'ytick',[0,1]); grid;
```



(d) Quantize the direct-form coefficients to 3 decimals (by rounding). Now plot the log-magnitude and amplitude responses of the resulting filter. Use 2 rows and 1 column of subplots.

Solution: The following script plots the responses.

```
D = 3; h2 = round(h*10^D)/10^D; %Quantized to 3 decimals
H = zerophase(h2,1,f*pi); Hmag = abs(H);
Hdb = 20*log10(Hmag/max(Hmag));
figure('position',[0,0,8,5]*72);
subplot(2,1,1); % Log-magnitude plot
plot(f,Hdb,'linewidth',1.5); axis([0,1,-100,10]);
ylabel('Decibels'); title('Log-Magnitude Response');
set(gca,'xtick',fo,'ytick',[-100,-As,0]); grid;
subplot(2,1,2); % Amplitude plot
plot(f,H,'linewidth',1.5); axis([0,1,-0.1,1.1]);
ylabel('Amplitude'); title('Amplitude Response');
xlabel('Digital Frequency (in \pi radians)');
set(gca,'xtick',fo,'ytick',[0,1]); grid;
```



(e) Comment on the plots in parts (b), (c), and (d).

Comment: Design specifications in part (c) clearly are not satisfied (although the amplitude doesn't reveal it) while those in part (b) are barely satisfied. This is because some zeros of the filter are closely clustered.

(f) Based on the results of this problem, determine how many significant *bits* (and not decimals) are needed in practice to represent FIR direct form realizations.

Solution: We need more than 4 decimal accuracy. Since each digit is equal to $\log_2(10) = 3.3219$ bits we need $4(\log_2(10)) = 13.2877$ or 14 bits of binary accuracy.

Problem 8.8: Pairing and Ordering of SOS

Consider a sixth-order filter with a system function

$$H(z)=\frac{(1+z^{-2})(1+z^{-1})^2(1-2\cos(\pi/6)z^{-1}+z^{-2})}{(1-1.6\cos(\pi/4)z^{-1}+0.64z^{-2})(1+1.6\cos(\pi/4)z^{-1}+0.64z^{-2})(1-1.8\cos(\pi/6)z^{-1}+0.81z^{-2})}$$

It is to be implemented as a cascade of second-order sections. Considering only the efects of roind-off noise, determine what the best pole-zero pairing is, and the best ordering of the second-order sections is.

Answer: Consider the pole-zero locations of the system:

```
clear; clc;
sos = [1,0,1,1,-1.6*cos(pi/4),0.64;
    1,2,1,1,1.6*cos(pi/4),0.64;
    1,-2*cos(pi/6),1,1,-1.8*cos(pi/6),0.81];
[b,a] = sos2tf(sos);
Zmag = abs(roots(b))'; Zpha = angle(roots(b))'/pi;
```

Zero Locations:

```
format rational; disp([Zmag;Zpha]);
```

```
        1           1           1           1           1           1
        1/6        -1/6         1/2        -1/2         1           1
```

```
Pmag = abs(roots(a))'; Ppha = angle(roots(a))'/pi;
```

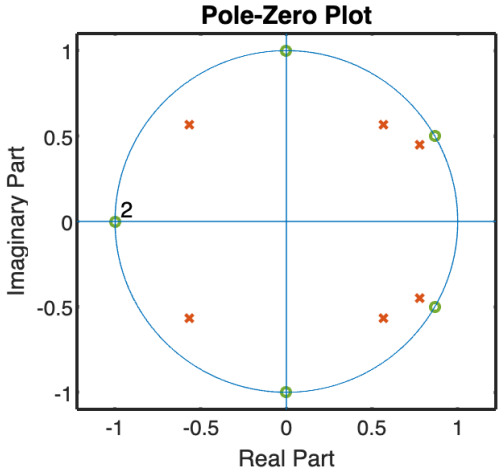
Pole Locations:

```
disp([Pmag;Ppha]); format short;
```

```
        4/5         4/5        9/10        9/10         4/5         4/5
        3/4        -3/4         1/6        -1/6         1/4        -1/4
```

Note that all six zeros of this system lie on the unit circle, with two at $z=-1$, a complex pair at $z=\pm\jmath$, and a complex pair at $z=\mathrm{e}^{\pm\jmath\pi/6}$. The poles, on the other hand, are at $z=0.8\mathrm{e}^{\pm\jmath 3\pi/4}$, $z=0.8\mathrm{e}^{\pm\jmath\pi/4}$, and $z=0.9\mathrm{e}^{\pm\jmath\pi/6}$. The pole-zero plot of the system function $H(z)$ is:

```
figure('Units','inches','Position',[0,0,3.5,3]);
[Hz,Hp,Hl] = zplane(b,a);
set(Hz,'linewidth',1.5,'Color',[0.466,0.674,0.188],'markersize',4);
set(Hp,'linewidth',1.5,'Color',[0.850,0.325,0.098],'markersize',5);
set(Hl,'linewidth',0.5,'linestyle','-');
```



The general strategy for pairing poles with zeros is to first find the two poles that are closest to the unit circle, in this casethose at $z = 0.9e^{\pm j\pi/6}$, and pair these with the two zeros that are closest to these poles, which are those on the unit circle at $z = e^{\pm j\pi/6}$, Thus the first pole-zero pairing yieplds the secod-0rder section

$$H_1(z) = \frac{1 - 2\cos(\pi/6)z^{-1} + z^{-2}}{1 - 1.8\cos(\pi/6)z^{-1} + 0.81z^{-2}}.$$

Of the remaining poles, we next find the pair that is closest to the unit circle, which are either those at $z = 0.8e^{\pm j\pi/4}$ or those at $z = 0.8e^{\pm j3\pi/4}$. Let us arbitrarily select the first of these. Paired with these poles would then be the zeros at $z = \pm j$, which gives the second-order section

$$H_2(z) = \frac{1 + z^{-2}}{1 - 1.6\cos(\pi/4)z^{-1} + 0.64z^{-2}}.$$

Finally, for the last section we have

$$H_3(z) = \frac{(1 + z^{-1})^2}{1 + 1.6\cos(\pi/4)z^{-1} + 0.64z^{-2}}.$$

The cascade is the done in the reverse order, with the first second-order section being $H_3(z)$, followed by $H_2(z)$, and then $H_1(z)$, which will reduce the variance and the PDF of the output noise.

$$H(z) = \left(\frac{(1 + z^{-1})^2}{1 + 1.6\cos(\pi/4)z^{-1} + 0.64z^{-2}}\right) \times \left(\frac{1 + z^{-2}}{1 - 1.6\cos(\pi/4)z^{-1} + 0.64z^{-2}}\right) \times \left(\frac{1 - 2\cos(\pi/6)z^{-1} + z^{-2}}{1 - 1.8\cos(\pi/6)z^{-1} + 0.81z^{-2}}\right).$$

MATLAB Function

```
function B = dec2rep(Dorg,nbits,option)
D = abs(Dorg);
B = zeros(1,nbits);
B(2:end)= fix(rem( rem(D,1)*pow2(1:nbits-1),2));
if Dorg < 0
    B(1) = 1;
    switch option
        case 'sign'
            B = B;
        case 'one'
            B(2:end) = 1-B(2:end);
        case 'two'
            B(2:end) = 1-B(2:end);
            B(end) = B(end) + 1;
            for i = length(B):-1:2
                if B(i) == 2
                    B(i) = 0;
                    B(i-1) = B(i-1)+1;
                end
            end
    end
else
    B(1) = 0;
end
end
```