ECE503: Finite Precision Signal Processing
Lecture 11

D. Richard Brown III

WPI

09-Apr-2012

Lecture 11 Topics

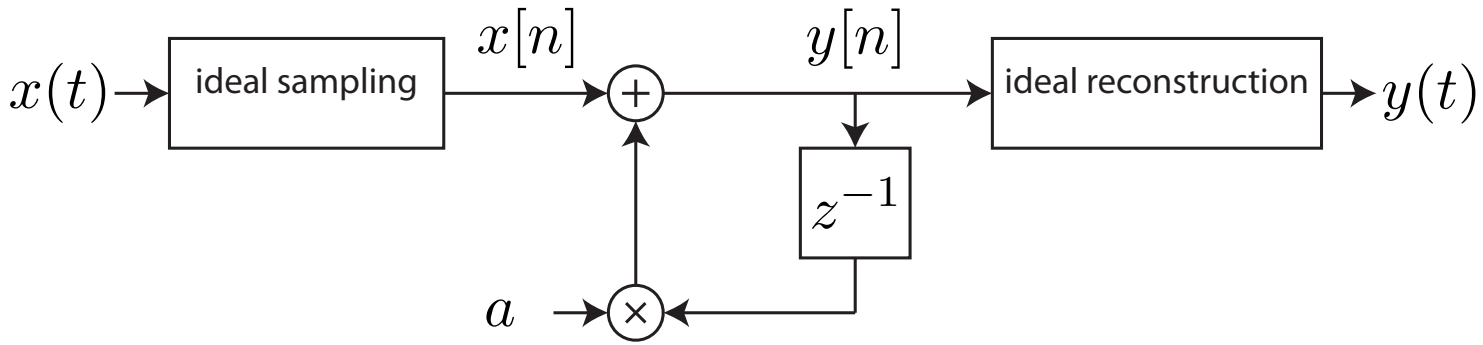
1. Motivation and context
2. Quantization basics
3. Effect of coefficient quantization on FIR filters
4. Effect of coefficient quantization on IIR filters: "pole sensitivity"
5. Input quantization and propagation of quantization noise to output
6. Product round-off error analysis

A Simple DSP System

Suppose we wish to implement the transfer function / difference equation

$$H(z) = \frac{1}{1 - az^{-1}} \qquad \Leftrightarrow \qquad y[n] = x[n] + ay[n-1]$$

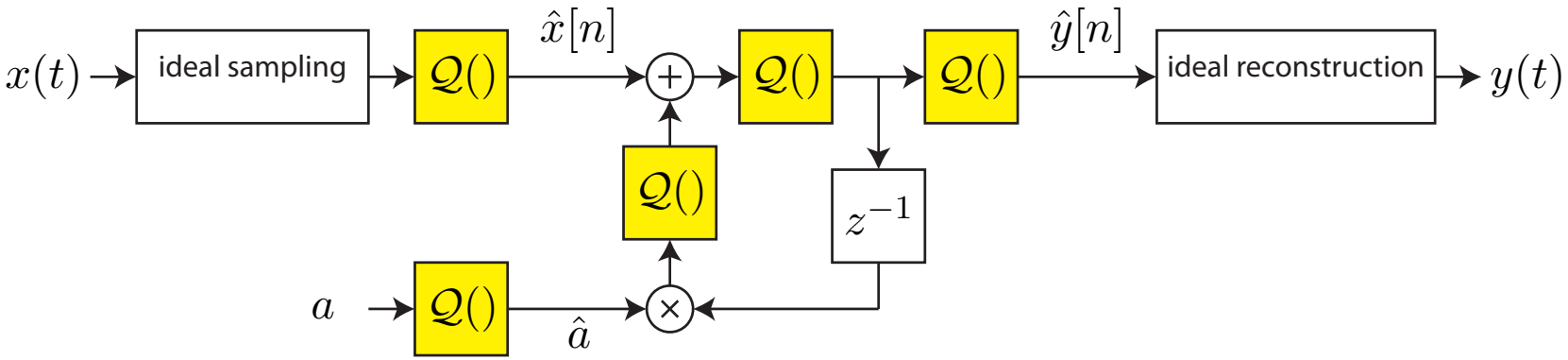
We could draw a block diagram (realization) of the system:



Unfortunately, this block diagram represents an idealized view of how the system is actually going to work. Some practical considerations:

- Input/output quantization.
- Filter coefficient quantization.
- Product roundoff.
- Potential overflow in sums.

A Simple DSP System: A More Realistic View



Our nice simple LTI system is now highly nonlinear.

In general, nonlinear systems like this are difficult to analyze. Hence, we must isolate the sources of quantization error and adopt some approximate approaches to make the analysis tractable:

- Analyze the effect of coefficient quantization, assuming all other quantization errors are negligible.
- Analyze the effect of input quantization, assuming a statistical model of quantization error and all other quantization errors are negligible.
- Analyze the effect of product roundoff, assuming a statistical model of quantization error and all other quantization errors are negligible.

Quantization Basics

Given a real number x, we denote the quantized value of x as

$$\hat{x} = \mathcal{Q}(x) = x + \epsilon$$

where ϵ is the "quantization error".

There are two main types of quantization:

1. **Truncation**: just discard least significant bits
2. **Rounding**: choose closest value

As an example, suppose we want to quantize $\frac{1}{\sqrt{2}} \approx 0.7071$ to a fixed point number with two fractional bits. If we **truncate**, we have

$$\mathcal{Q}(1/\sqrt{2}) = \boxed{0_\Delta 10}110\cdots = 0.50$$

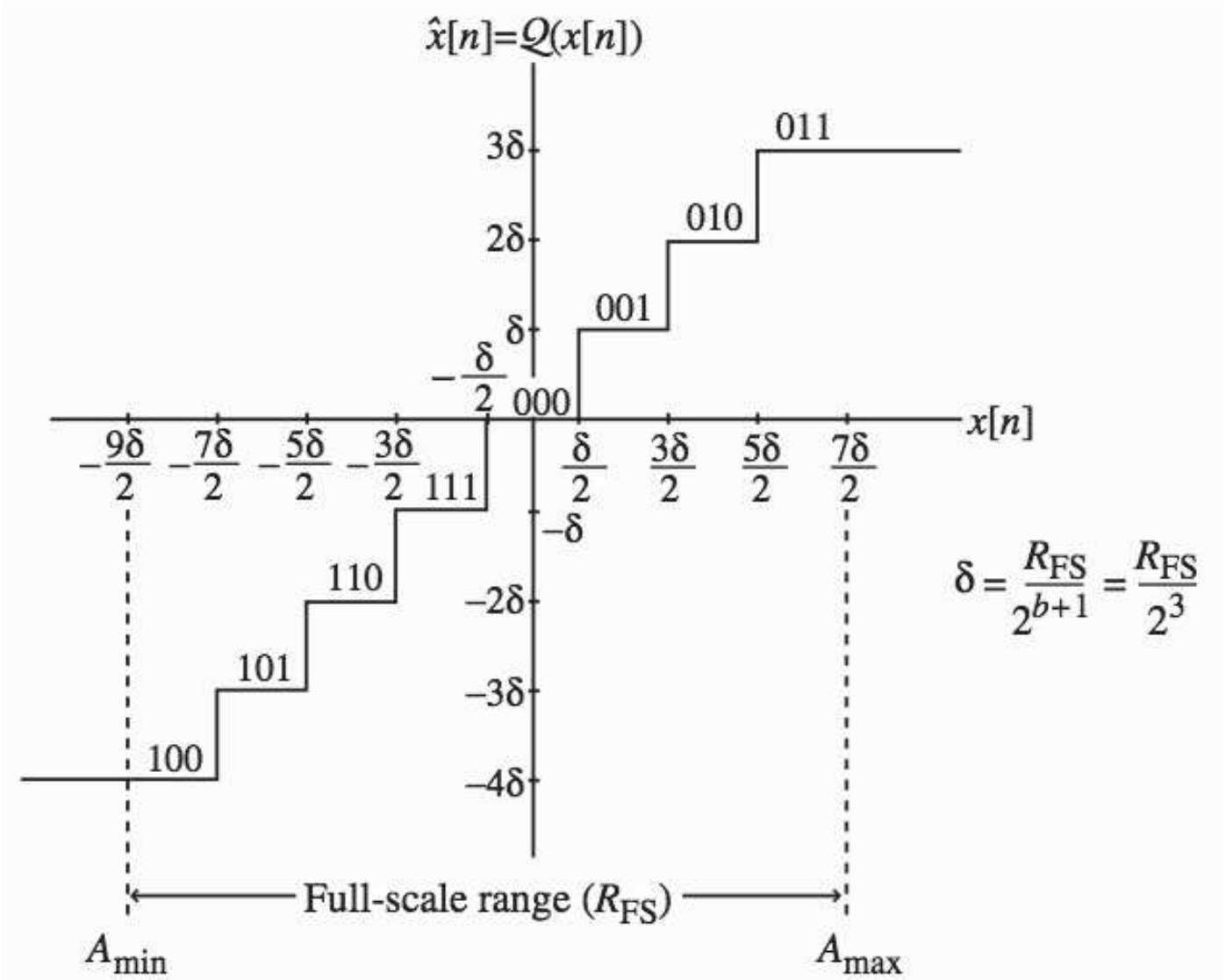
whereas if we **round**, we have

$$\mathcal{Q}(1/\sqrt{2}) = \boxed{0_\Delta 11} = 0.75$$

We usually prefer rounding because the quantization errors are zero-mean and bounded $-\frac{\delta}{2} \leq \epsilon < \frac{\delta}{2}$, where δ is the quantizer step size (assuming no overflow).

Bipolar 3-Bit Quantizer Example (Fig. 12.16)

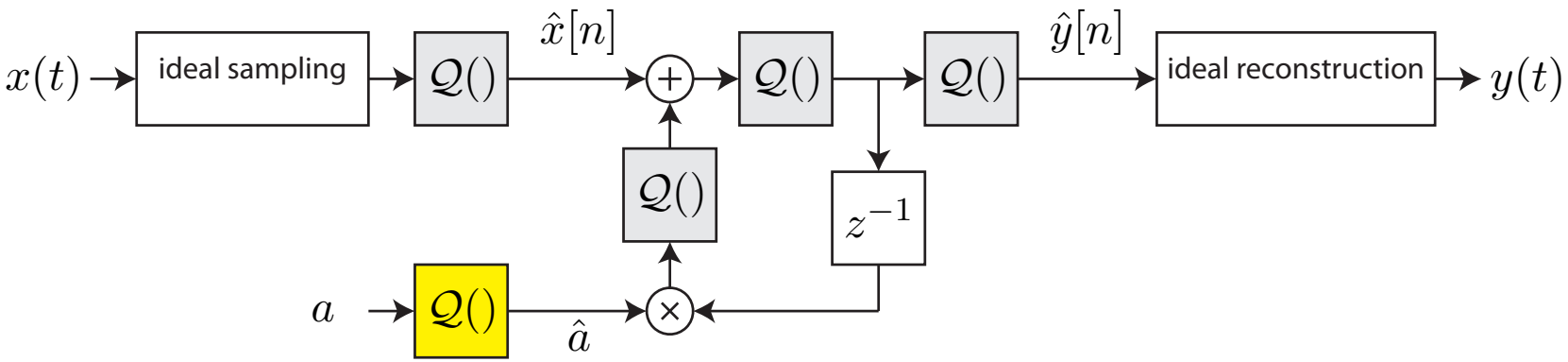
Rounding quantization, saturation overflow, and two's complement.



Matlab Code to Implement a Rounding Quantizer with Saturation Overflow

```
function [xhat,delta,qerr] = quant(x,nbits,RFS)
% function [xhat,delta,qerr] = quant(x,nbits,RFS)
%
% x is the unquantized value (floating point)
% nbits is the number of bits in the quantizer
% RFS is the full scale range (peak-to-peak)

% compute quantizer step size and min/max levels for saturation
delta = RFS/2^nbits;
xhatmax = (2^(nbits-1)-1)*delta;
xhatmin = (-2^(nbits-1))*delta;

% quantize
xhat = round(x/delta)*delta;

% check for overflow and saturate
if xhat>xhatmax
    xhat = xhatmax;
end
if xhat<xhatmin
    xhat = xhatmin;
end

% compute quantization error
qerr = xhat - x;
```

Part I: Effect of Coefficient Quantization

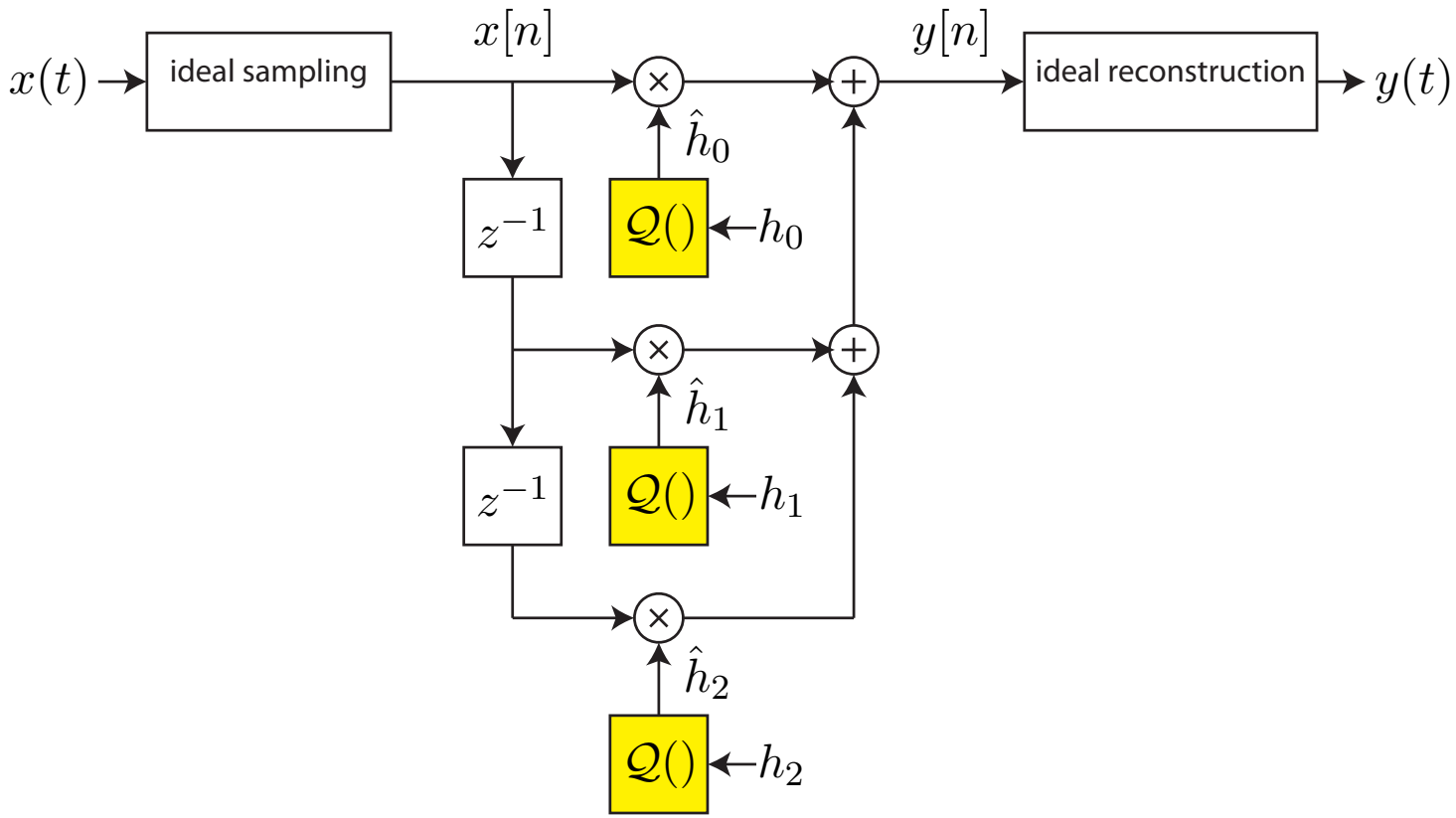


We will first focus on the effect of coefficient quantization and ignore the other sources of quantization error.

1. How does coefficient quantization affect FIR filters?
2. How does coefficient quantization affect IIR filters?
   - Analysis of pole/zero sensitivity to coefficient quantization.

FIR Filter Coefficient Quantization (1 of 3)

For a direct form FIR filter, we have the realization structure



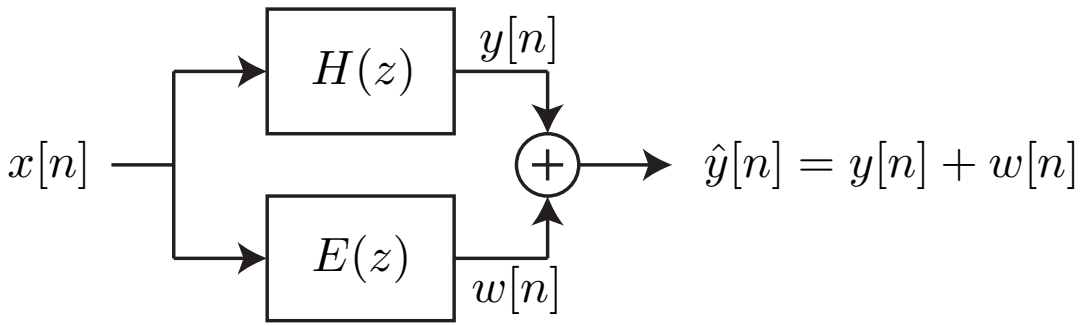
with $\hat{h}_n = h_n + e_n$.

FIR Filter Coefficient Quantization (2 of 3)

For a causal FIR filter with N coefficients, we have

$$\hat{H}(z) = \sum_{n=0}^{N-1} \hat{h}_n z^{-n} = \sum_{n=0}^{N-1} (h_n + e_n) z^{-n}$$
$$= \sum_{n=0}^{N-1} h_n z^{-n} + \sum_{n=0}^{N-1} e_n z^{-n} = H(z) + E(z)$$

Hence, the quantized FIR filter $\hat{H}(z)$ is equivalent to a parallel connection of $H(z)$ and $E(z)$:



Note $E(z)$ is FIR and causal.

FIR Filter Coefficient Quantization (3 of 3)

It is of interest to analyze the effect of the quantization error on the magnitude response of the system. We can develop a bound for the worst-case magnitude response error caused by FIR filter coefficient quantization as follows.

By definition, for FIR $E(z)$ with N coefficients

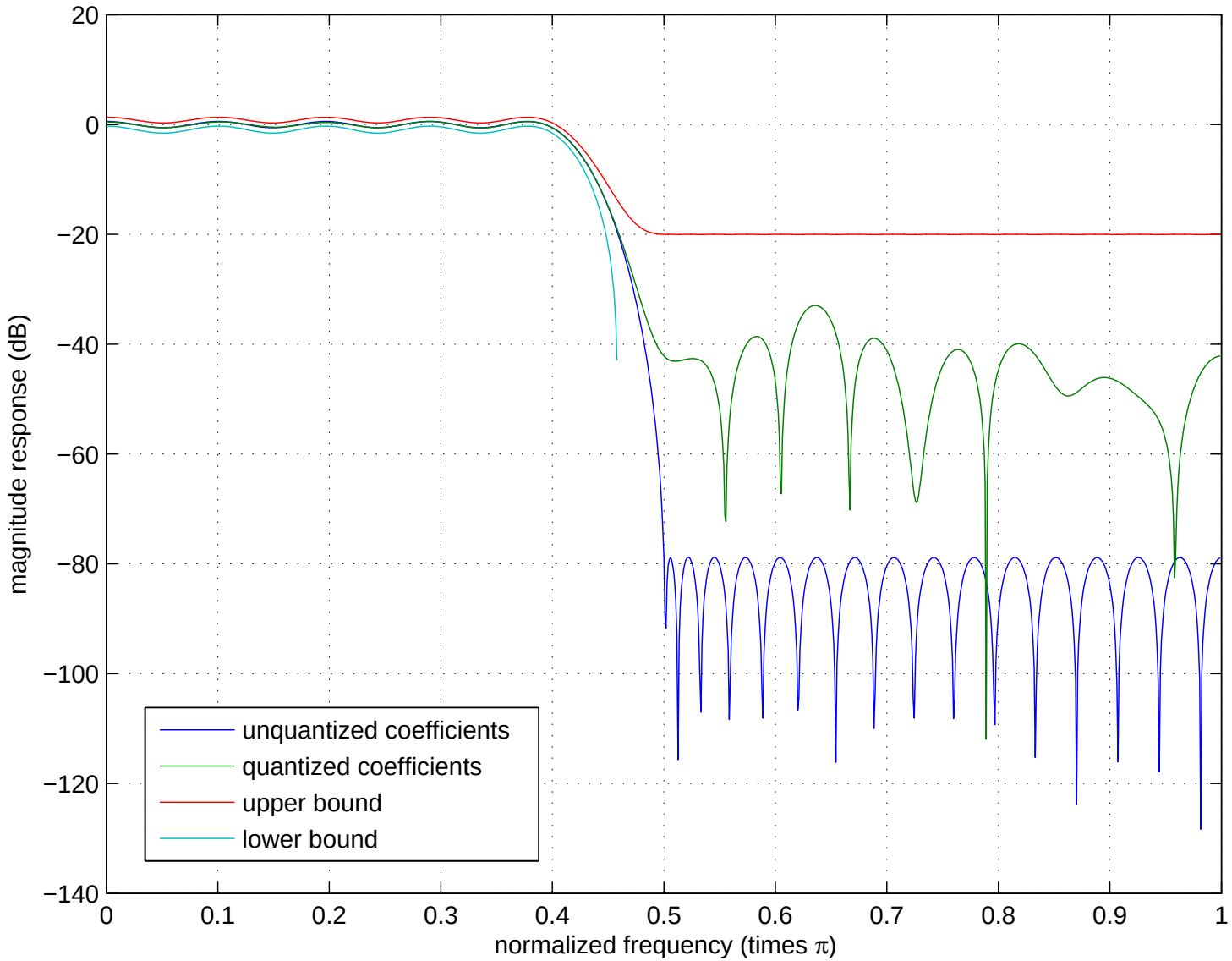
$$E(\omega) = \sum_{n=0}^{N-1} e_n e^{-j\omega n}$$

With rounding quantization, each e_n is bounded between $-\delta/2$ and $\delta/2$. Hence,

$$|E(\omega)| = \left| \sum_{n=0}^{N-1} e_n e^{-j\omega n} \right| \leq \sum_{n=0}^{N-1} \left| e_n e^{-j\omega n} \right| \leq \sum_{n=0}^{N-1} \frac{\delta}{2} = \frac{N\delta}{2}$$

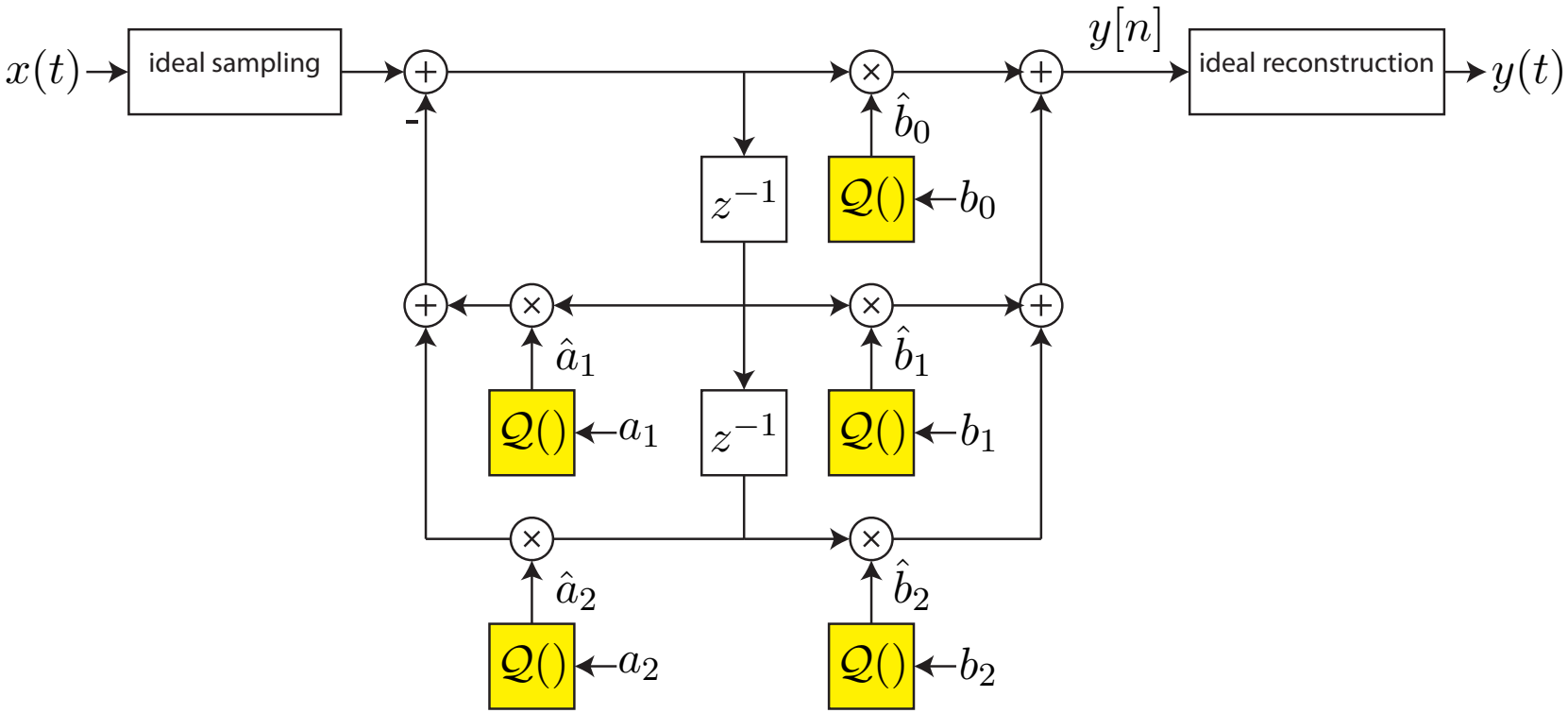
Note that the bound is worse for longer filters but better when the quantizer step size is small.

FIR Coefficient Quantization Example ($\delta = 2^{-8}$, $N = 51$)



IIR Filter Coefficient Quantization

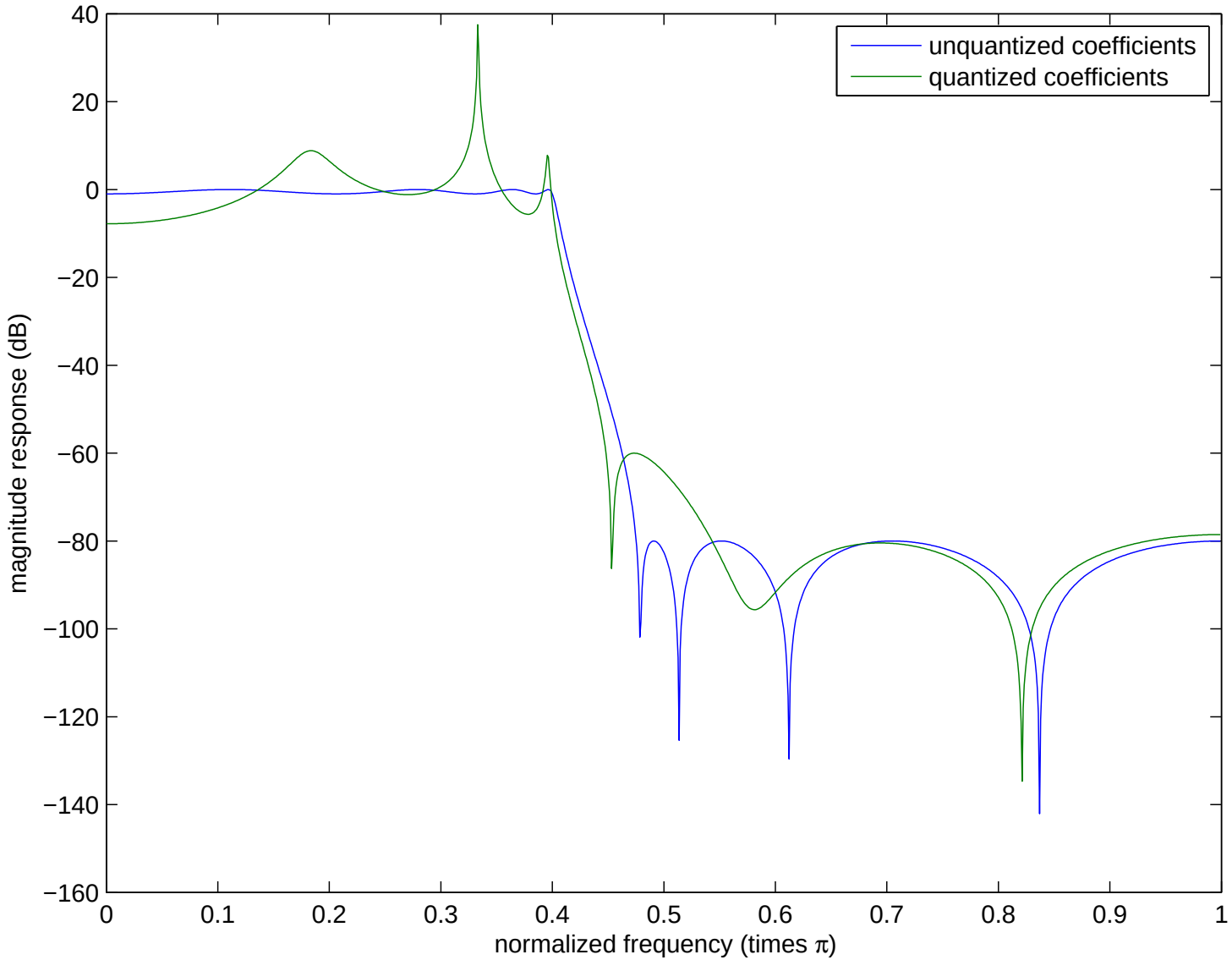
For a direct-form II IIR filter, we have



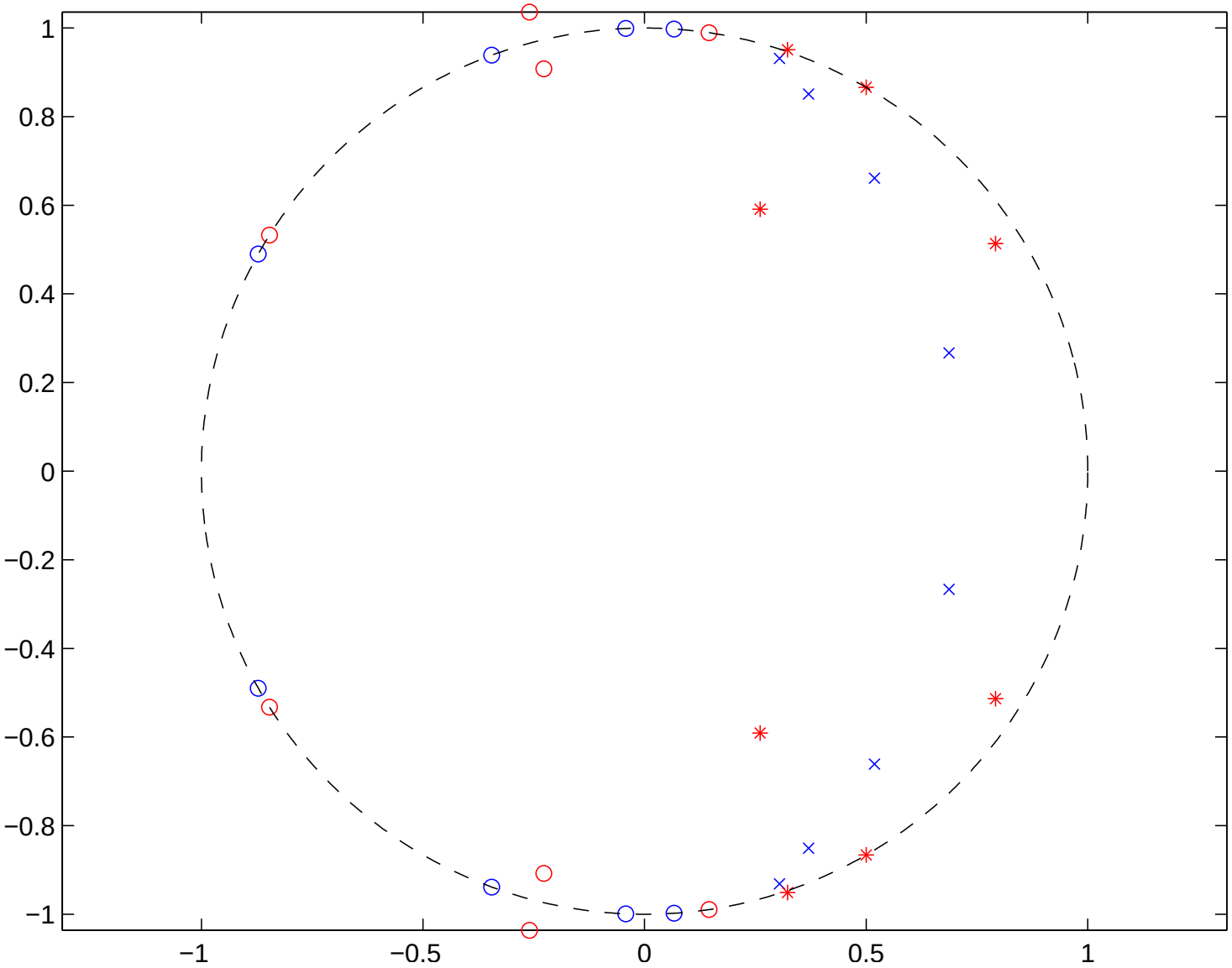
with $\hat{b}_n = b_n + \Delta b_n$ and $\hat{a}_n = a_n + \Delta a_n$.

Unfortunately, the FIR analysis techniques used previously are not applicable here because of the feedback in the system. Let's look at some examples before developing analytical techniques.

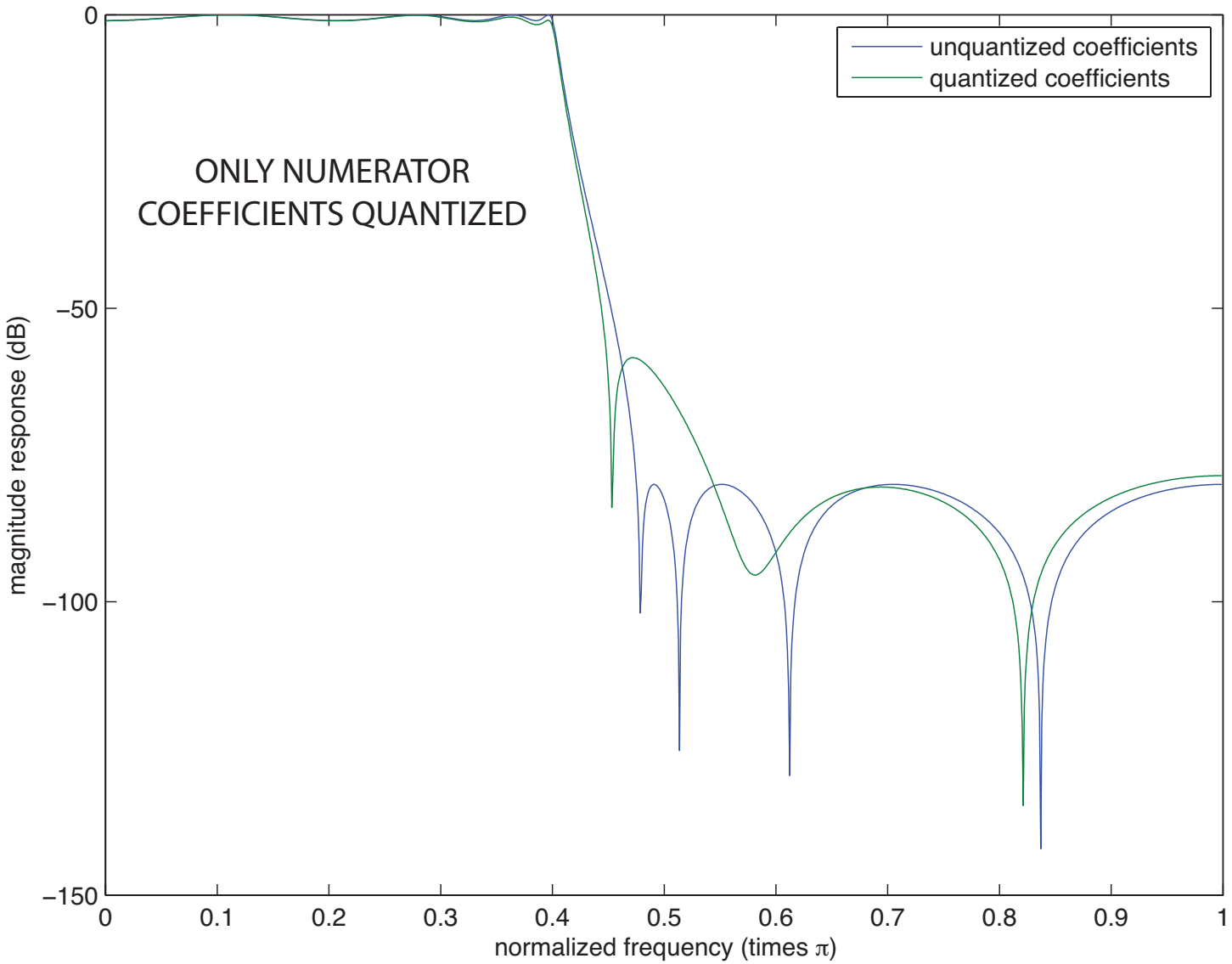
IIR Coefficient Quantization Example (8th order DF-II)



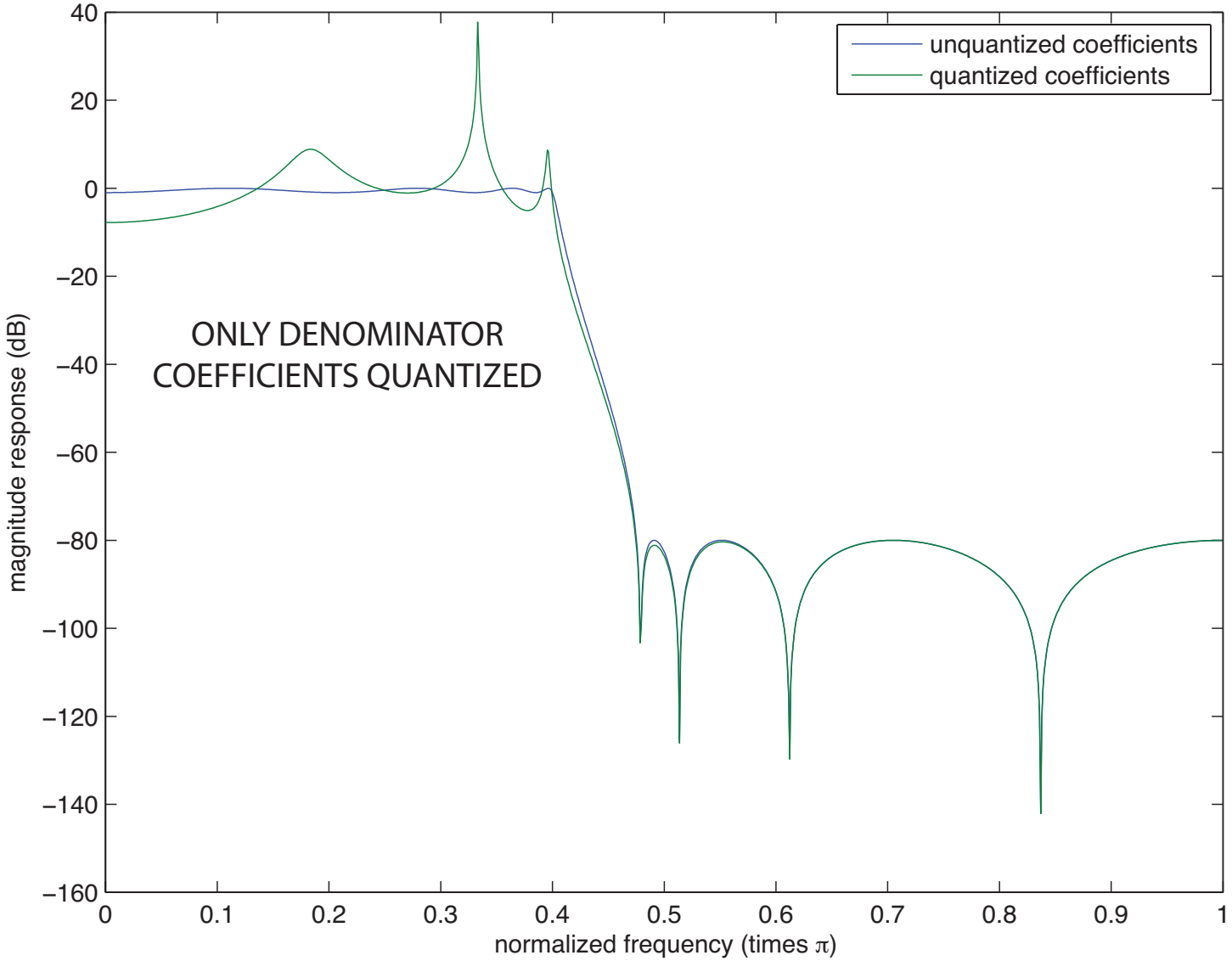
IIR Coefficient Quantization Example (8th order DF-II)



IIR Coefficient Quantization Example (8th order DF-II)



IIR Coefficient Quantization Example (8th order DF-II)



IIR Filter Coefficient Quantization Remarks/Observations

1. Previous examples all used 8-bit coefficient quantization.
2. Numerator coefficient quantization tends to have less effect on the magnitude response than denominator coefficient quantization.
3. Denominator coefficient quantization can cause an IIR filter to become unstable (e.g. homework problem).
4. This filter was realized as a single 8th order DF-II section, rather than four cascaded 2nd order sections.
5. In general, a cascaded 2nd order sections (SOS) realization is going to be better with finite precision coefficients because the dynamic range of the coefficients is reduced.

Denominator of $H(z)$ when realized as a single 8th order section:

$$Q(z) = 1 - 3.76z^{-1} + 8.20z^{-2} - 11.85z^{-3} + 12.33z^{-4} - 9.30z^{-5} + 4.98z^{-6} - 1.74z^{-7} + 0.32z^{-8}$$

Ratio of 38.9 between largest coefficient and smallest non-zero coefficient.

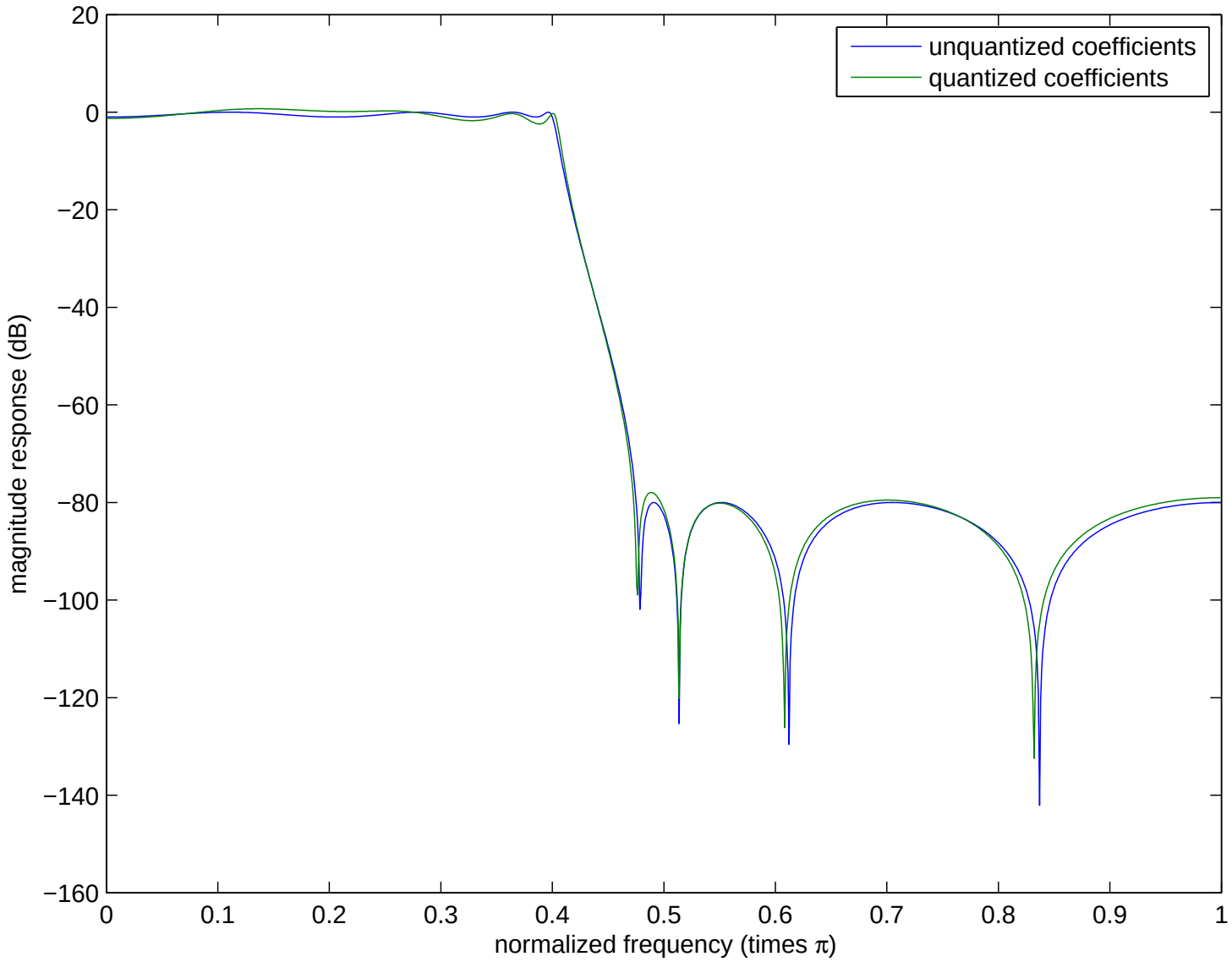
IIR Filter DF-II SOS Coefficient Quantization

When we convert the filter to second order sections, we get the following denominators, where $Q(z) = Q_1(z)Q_2(z)Q_3(z)Q_4(z)$:

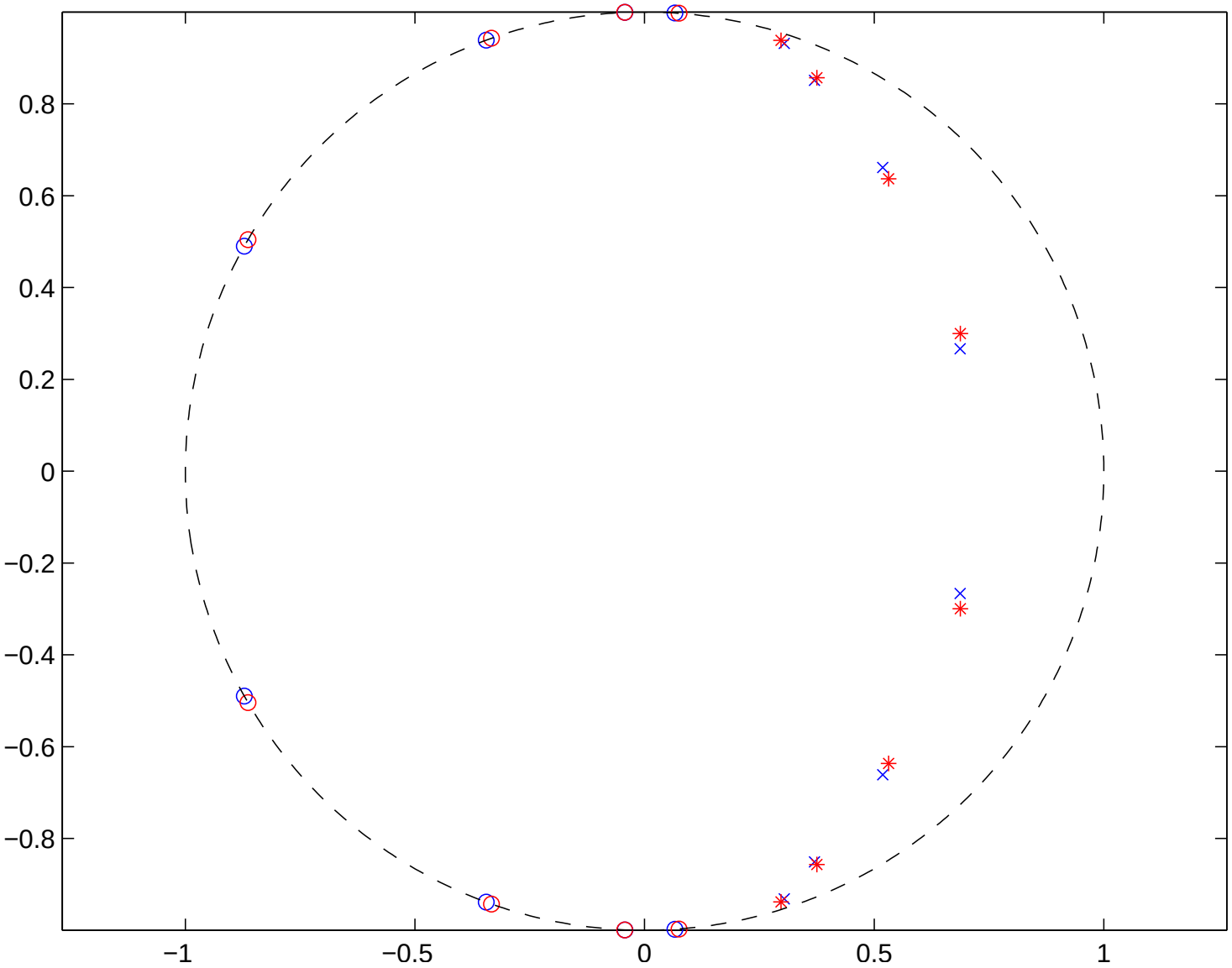
$$Q_1(z) = 1 - 0.74z^{-1} + 0.8610z^{-2}$$
$$Q_2(z) = 1 - 1.04z^{-1} + 0.7062z^{-2}$$
$$Q_3(z) = 1 - 1.37z^{-1} + 0.5431z^{-2}$$
$$Q_4(z) = 1 - 0.61z^{-1} + 0.9605z^{-2}$$

Ratio of 2.53 between largest coefficient and smallest non-zero coefficient. By reducing the dynamic range, finer coefficient quantization is possible, even with less bits. This means the poles don't move as much and the SOS filter is more accurate.

IIR Coefficient Quantization Example (6-bit SOS DF-II)

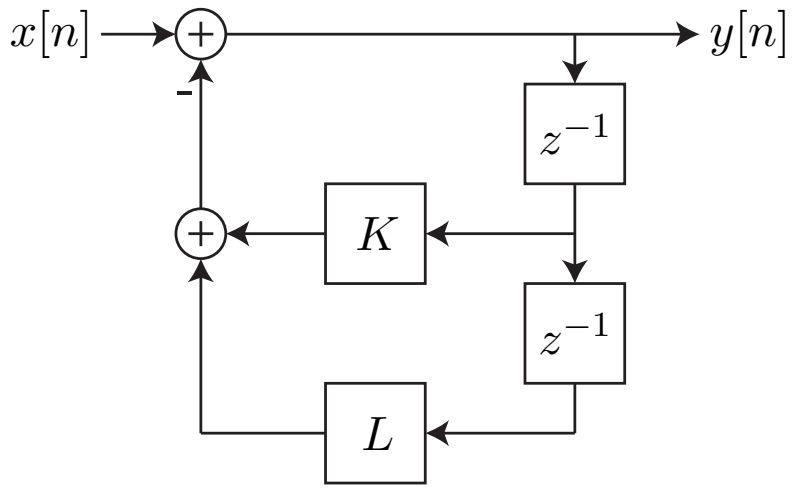


IIR Coefficient Quantization Example (6-bit SOS DF-II)



Effect of Realization Structure on Pole Locations

Consider the following DF-II realization of an all-pole second-order IIR filter:

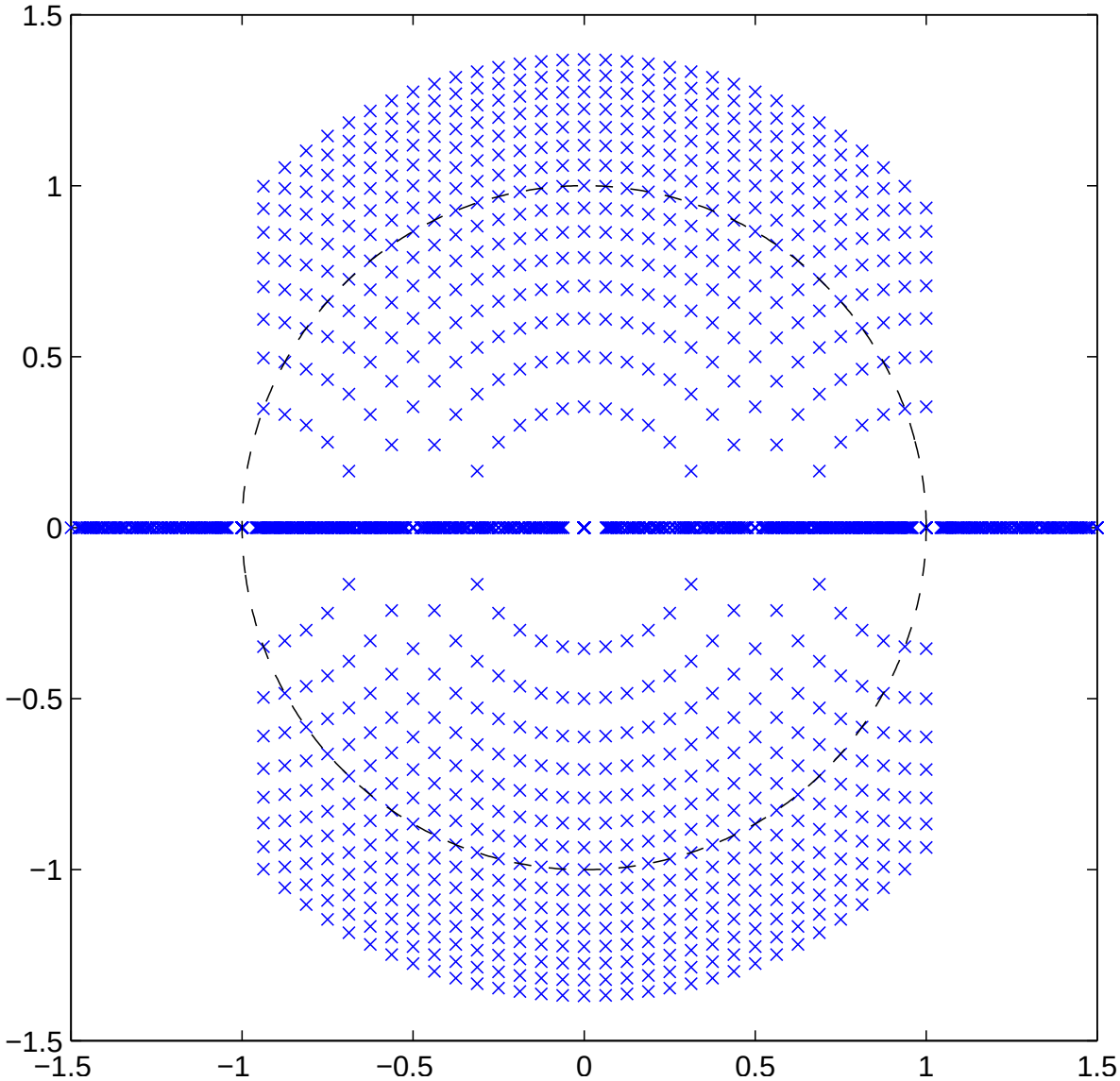


The transfer function is

$$H(z) = \frac{1}{1 + Kz^{-1} + Lz^{-2}}$$

If we assume a 5-bit word length (one sign bit, one non-fractional bit, three fractional bits) for coefficients K and L, there are only a finite number of possible pole locations that we can achieve.

DF-II Second Order Section Pole Locations (5-bit)



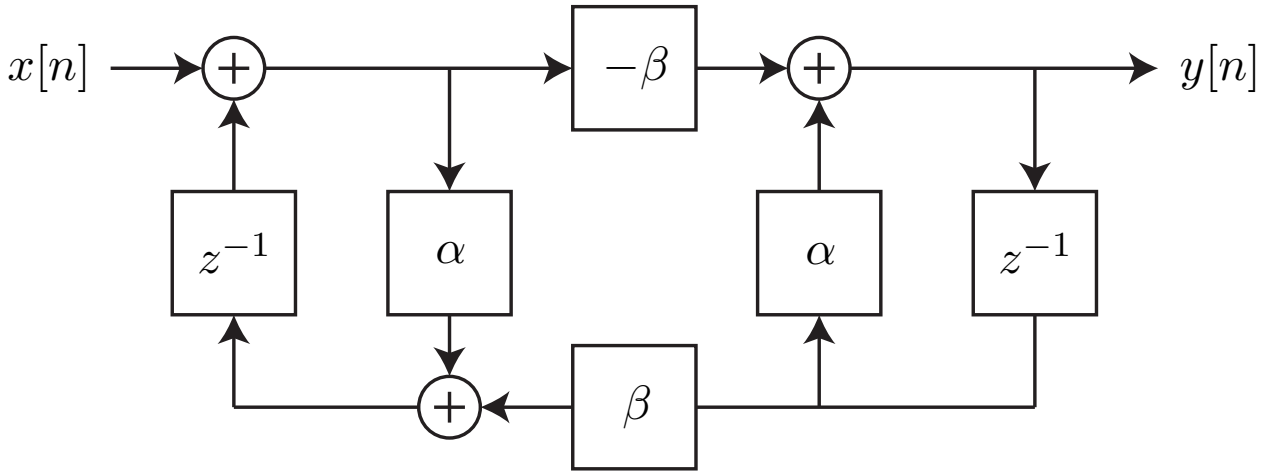
DF-II Second Order Section Pole Locations

Remarks:

1. More pole density near $z = \pm j$.
2. Less pole density near $z = \pm 1$.
3. DF-II second order structure may be highly inaccurate for lowpass or highpass filters with complex poles near $z = \pm 1$.
4. Textbook Figure 12.10 is incorrect (but the overall point is correct):
   - Doesn't show real-valued poles.
   - Uses different fixed-point representations for K (2 fractional bits) and L (3 fractional bits).

Effect of Realization Structure on Pole Locations

Now consider the following **coupled-form** realization of an all-pole second-order IIR filter:

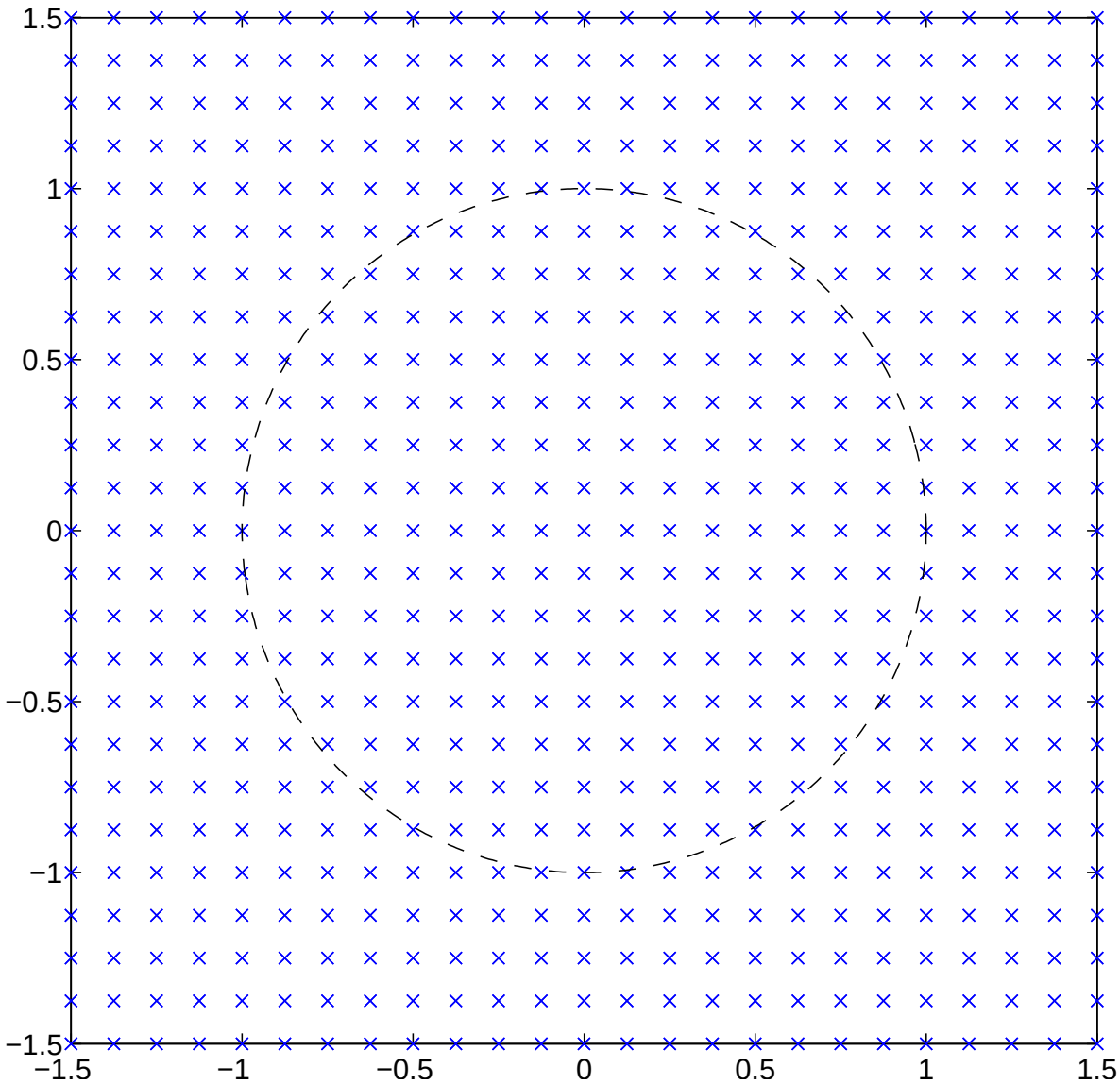


The transfer function is

$$H(z) = \frac{\beta}{1 - 2\alpha z^{-1} + (\alpha^2 + \beta^2)z^{-2}}$$

If we assume a 5-bit word length (one sign bit, one non-fractional bit, three fractional bits) for α and β, there are only a finite number of possible pole locations that we can achieve.

Coupled-Form Second Order Section Pole Locations



Coupled-Form Second Order Section Pole Locations

Remarks:

1. Pole locations are uniformly distributed on z-plane. Nice!
2. Suitable for any type of IIR filter.
3. Quantized pole displacement is easily bounded.
4. Why not always use the coupled form? Is there any disadvantage?
5. Textbook Figure 12.12 is incorrect (but the overall point is correct):
   - Doesn't show real-valued poles.

Note: This technique can also be used to determine quantized zero locations.

Analysis of Quantized Root Displacements

Problem setup: We have an Nth degree polynomial $B(z)$ with simple roots and $b_N = 1$:

$$B(z) = \sum_{i=0}^{N} b_i z^i = \prod_{k=1}^{N} (z - \lambda_k)$$

Note $B(z)$ can be a numerator or denominator polynomial. We want to understand how quantizing the coefficients b_i affects the roots λ_k (which can correspond to the poles or the zeros of a transfer function).

The polynomial with quantized coefficients is

$$\hat{B}(z) = \sum_{i=0}^{N} \hat{b}_i z^i = \sum_{i=0}^{N} (b_i + \Delta b_i) z^i = B(z) + \sum_{i=0}^{N-1} (\Delta b_i) z^i = \prod_{k=1}^{N} (z - \hat{\lambda}_k)$$

Note the implicit assumption that $\Delta b_N = 0$ (since $b_N = 1$). Also note $\hat{\lambda}_k$ are the new displaced root locations after coefficient quantization.

Analysis of Quantized Root Displacements

We denote the original roots in polar coordinates as $\lambda_k = r_k e^{j\theta_k}$. We can write the displaced roots as

$$\begin{aligned}\hat{\lambda}_k &= (r_k + \Delta r_k)e^{j(\theta_k + \Delta\theta_k)} \\ &= (r_k + \Delta r_k)e^{j\theta_k}e^{j(\Delta\theta_k)} \\ &\approx (r_k + \Delta r_k)e^{j\theta_k}(1 + j(\Delta\theta_k))\end{aligned}$$

where the last line uses the first order series approximation $e^x \approx 1 + x$ (implicitly assuming the angular displacement is small). Continuing,

$$\hat{\lambda}_k \approx e^{j\theta_k}\left\{r_k(1 + j(\Delta\theta_k)) + (\Delta r_k) + j(\Delta r_k)(\Delta\theta_k)\right\}$$

If the magnitude displacement is small, we can discard the last term because it is the product of two small numbers. Hence

$$\hat{\lambda}_k \approx e^{j\theta_k}\left\{r_k + jr_k(\Delta\theta_k)) + (\Delta r_k)\right\} = \lambda_k + e^{j\theta_k}\left\{jr_k(\Delta\theta_k)) + (\Delta r_k)\right\}$$

Hence the root displacement $\hat{\lambda}_k - \lambda_k \approx e^{j\theta_k}\left\{jr_k(\Delta\theta_k)) + (\Delta r_k)\right\}$.

Analysis of Quantized Root Displacements

Sanity check. Suppose

$$B(z) = z^2 - 1.4z + 0.98$$

We can compute the roots $\lambda_1 = -0.7 + 0.7j = r_1 e^{j\theta_1}$ and $\lambda_2 = -0.7 - 0.7j = r_2 e^{j\theta_2}$ with $r_1 = r_2 = 0.9899$ and $\theta_1 = -\theta_2 = 3\pi/4$.

Now suppose the roots are displaced so that $\Delta\theta_1 = -\Delta\theta_2 = \pi/100$ and $\Delta r_1 = \Delta r_2 = -0.01$. We know the exact locations of the displaced roots are at $\hat{\lambda}_k = (r_k + \Delta r_k)e^{j(\theta_k + \Delta\theta_k)}$. We can calculate this to be

$$\lambda_1 = -0.7144 + 0.6708j \text{ and } \lambda_2 = -0.7144 - 0.6708j$$

Hence the exact root displacements are

$$\hat{\lambda}_1 - \lambda_1 = -0.0144 - 0.0292j \text{ and } \hat{\lambda}_2 - \lambda_2 = -0.0144 + 0.0292j$$

Our previous analysis says the root displacements are approximately

$$\hat{\lambda}_k - \lambda_k \approx e^{j\theta_k} \{j r_k(\Delta\theta_k)) + (\Delta r_k)\}$$

Calculation of this result with our numbers gives

$$\hat{\lambda}_1 - \lambda_1 \approx -0.0149 - 0.0291j \text{ and } \hat{\lambda}_2 - \lambda_2 \approx -0.0149 + 0.0291j$$

which is pretty accurate.

Analysis of Quantized Root Displacements

So hold the main result $\hat{\lambda}_k - \lambda_k \approx e^{j\theta_k} \{jr_k(\Delta\theta_k)) + (\Delta r_k)\}$ for a bit while we work on another related problem.

Consider the rational function (recall our simple roots assumption)

$$\frac{1}{B(z)} = \sum_{i=1}^{N} \frac{\rho_i}{z - \lambda_i}$$

If we set $z = \hat{\lambda}_k$, we get

$$\frac{1}{B(\hat{\lambda}_k)} = \sum_{i=1}^{N} \frac{\rho_i}{\hat{\lambda}_k - \lambda_i} \approx \frac{\rho_k}{\hat{\lambda}_k - \lambda_k}$$

where the approximation results from the fact that the original root λ_k is assumed to be very close to the displaced root $\hat{\lambda}_k$ and all other terms in the PFE sum will be relatively insignificant. Hence, $\hat{\lambda}_k - \lambda_k \approx \rho_k B(\hat{\lambda}_k)$.

Analysis of Quantized Root Displacements

Recall

$$\hat{B}(z) = B(z) + \sum_{i=0}^{N-1} (\Delta b_i) z^i$$

Then

$$\hat{B}(\hat{\lambda}_k) = B(\hat{\lambda}_k) + \sum_{i=0}^{N-1} (\Delta b_i)(\hat{\lambda}_k)^i = 0$$

where the last equality is because $\hat{\lambda}_k$ is a root of $\hat{B}(z)$. Hence

$$B(\hat{\lambda}_k) = -\sum_{i=0}^{N-1} (\Delta b_i)(\hat{\lambda}_k)^i.$$

Plug this in to the result from the previous slide ($\hat{\lambda}_k - \lambda_k \approx \rho_k B(\hat{\lambda}_k)$) to get

$$\hat{\lambda}_k - \lambda_k \approx -\rho_k \sum_{i=0}^{N-1} (\Delta b_i)(\hat{\lambda}_k)^i \approx -\rho_k \sum_{i=0}^{N-1} (\Delta b_i)(\lambda_k)^i$$

where the second approximation uses the assumption that $\hat{\lambda}_k$ is very close to λ_k.

Analysis of Quantized Root Displacements

So we have two useful approximations now:

$$\hat{\lambda}_k - \lambda_k \approx e^{j\theta_k} \{jr_k(\Delta\theta_k)) + (\Delta r_k)\}$$

and

$$\hat{\lambda}_k - \lambda_k \approx -\rho_k \sum_{i=0}^{N-1} (\Delta b_i)(\lambda_k)^i$$

We can equate these and do a little rearranging to write

$$\boxed{\Delta r_k + jr_k(\Delta\theta_k) \approx -e^{-j\theta_k}\rho_k \sum_{i=0}^{N-1} (\Delta b_i)(r_k e^{j\theta_k})^i}$$

In other words, given the original root magnitudes $\{r_i\}$, original root angles $\{\theta_i\}$, coefficient displacements $\{\Delta b_i\}$, and partial fraction expansion residues $\{\rho_i\}$, we can calculate the magnitude and angle displacements ($\{\Delta r_i\}$ and $\{\Delta\theta_i\}$) for all of the roots of $B(z)$.

Analysis of Quantized Root Displacements

To make the main result

$$\Delta r_k + j r_k(\Delta\theta_k) \approx -e^{-j\theta_k}\rho_k \sum_{i=0}^{N-1} (\Delta b_i)(r_k e^{j\theta_k})^i$$

more explicit, we can denote the PFE coefficients $\rho_k = \alpha_k + j\beta_k$, equate the real and imaginary parts on each side, and do a bit of algebra to write

$$\Delta r_k = (-\alpha_k \boldsymbol{P}_k + \beta_k \boldsymbol{Q}_k)\,\Delta\boldsymbol{B}$$

$$\Delta\theta_k = -\frac{1}{r_k}(\beta_k \boldsymbol{P}_k + \alpha_k \boldsymbol{Q}_k)\,\Delta\boldsymbol{B}$$

where

$$\boldsymbol{P}_k = \begin{bmatrix}\cos\theta_k & r_k & r_k^2\cos\theta_k & \dots & r_k^{N-1}\cos((N-2)\theta_k)\end{bmatrix} \in \mathbb{R}^{1\times N}$$

$$\boldsymbol{Q}_k = \begin{bmatrix}-\sin\theta_k & 0 & r_k^2\sin\theta_k & \dots & r_k^{N-1}\sin((N-2)\theta_k)\end{bmatrix} \in \mathbb{R}^{1\times N}$$

$$\Delta\boldsymbol{B} = \begin{bmatrix}\Delta b_0 \\ \vdots \\ \Delta b_{N-1}\end{bmatrix} \in \mathbb{R}^{N\times 1}$$

Analysis of Quantized Root Displacements: Example

Let's apply our result to look at the pole sensitivity of

$$H(z) = \frac{1}{1 + Kz^{-1} + Lz^{-2}} = \frac{z^2}{z^2 + Kz + L} = \frac{1}{B(z)}$$

with $N = 2$, $b_0 = L$, $b_1 = K$, and $b_2 = 1$.

Note $B(z)$ has two roots $\lambda_1 = re^{j\theta}$ and $\lambda_2 = re^{-j\theta}$. We want to understand how changes in K and L affect r and θ.

Using our prior analysis, we can write

$$\begin{aligned} \boldsymbol{P}_1 &= \begin{bmatrix} \cos\theta & r \end{bmatrix} = \boldsymbol{P}_2 \\ \boldsymbol{Q}_1 &= \begin{bmatrix} -\sin\theta & 0 \end{bmatrix} = -\boldsymbol{Q}_2 \\ \Delta\boldsymbol{B} &= \begin{bmatrix} \Delta L \\ \Delta K \end{bmatrix} \end{aligned}$$

Continued...

Analysis of Quantized Root Displacements: Example

We can write the partial fractional expansion of

$$\frac{1}{B(z)} = \frac{\rho_1}{z-\lambda_1} + \frac{\rho_2}{z-\lambda_2}$$

yields

$$\rho_1 = -\frac{j}{2r\sin\theta} \text{ and } \rho_2 = \frac{j}{2r\sin\theta}$$

Hence $\alpha_1 = \alpha_2 = 0$ and $\beta_1 = -\beta_2 = -\frac{1}{2r\sin\theta}$.

Putting it all together, we can determine the sensitivity of the first pole at $\lambda_1 = re^{j\theta}$ as

$$\Delta r = \beta_1 \boldsymbol{Q}_1 \Delta \boldsymbol{B} = -\frac{1}{2r\sin\theta}\begin{bmatrix}-\sin\theta & 0\end{bmatrix}\begin{bmatrix}\Delta L \\ \Delta K\end{bmatrix} = \frac{\Delta L}{2r}$$

$$\Delta\theta = -\frac{1}{r}(\beta_1 \boldsymbol{P}_1 \Delta \boldsymbol{B}) = -\frac{1}{r}\left(-\frac{1}{2r\sin\theta}\begin{bmatrix}\cos\theta & r\end{bmatrix}\begin{bmatrix}\Delta L \\ \Delta K\end{bmatrix}\right) = \frac{\Delta L}{2r^2\tan\theta} + \frac{\Delta K}{2r\sin\theta}$$

Observations:

- The pole displacements are large when r is small.
- The pole displacements are also large when θ is close to zero or $\pm\pi$.
- These results are consistent with the figure on slide 23.

Analysis of Quant. Root Displacements: Other Structures

The prior analysis can be extended to other structures as follows:

1. Denote the coefficients in the structure as $\gamma_1, \ldots, \gamma_R$.
2. Coefficient quantization changes each coefficient to $\hat{\gamma}_k = \gamma_k + \Delta\gamma_k$.
3. Since $B(z)$ is a function of these coefficients, this also indirectly changes the coefficients of the polynomial $B(z)$. We can relate changes in the structure coefficients γ_k to the polynomial coefficients b_k with

$$\Delta b_k = \sum_{i=1}^{R} \frac{\partial b_k}{\partial \gamma_i} \Delta\gamma_i \text{ for } k = 0, \ldots, N-1$$

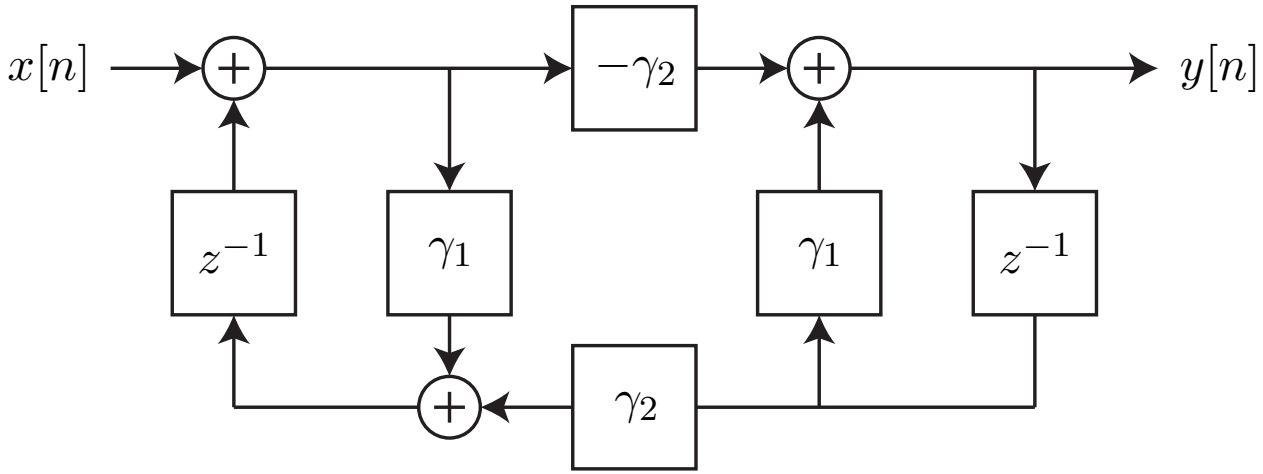
4. Note that $\Delta\boldsymbol{B} = \boldsymbol{C}\Delta\boldsymbol{\gamma}$ where $\boldsymbol{C} \in \mathbb{R}^{N \times R}$ is a gradient matrix.
5. Hence

$$\Delta r_k = \left(-\alpha_k \boldsymbol{P}_k + \beta_k \boldsymbol{Q}_k\right) \boldsymbol{C}\Delta\boldsymbol{\gamma}$$

$$\Delta\theta_k = -\frac{1}{r_k}\left(\beta_k \boldsymbol{P}_k + \alpha_k \boldsymbol{Q}_k\right) \boldsymbol{C}\Delta\boldsymbol{\gamma}$$

Analysis of Quantized Root Displacements: Example

We can illustrate the process for the coupled form second order IIR filter



with $\gamma_1 = r\cos\theta$ and $\gamma_2 = r\sin\theta$ to get the original roots $\lambda_1 = re^{j\theta}$ and $\lambda_2 = re^{-j\theta}$. Note $R = 2$. The transfer function is

$$H(z) = \frac{-\gamma_2}{1 - 2\gamma_1 z^{-1} + (\gamma_1^2 + \gamma_2^2)z^{-2}}$$

Hence $B(z) = z^2 - 2\gamma_1 z + (\gamma_1^2 + \gamma_2^2)$ with $N = 2$, $b_0 = \gamma_1^2 + \gamma_2^2$, $b_1 = -2\gamma_1$ and $b_2 = 1$. Continued...

Analysis of Quantized Root Displacements: Example

The next step is to compute the $\boldsymbol{C}$ matrix

$$\boldsymbol{C} = \begin{bmatrix} \frac{\partial b_0}{\partial \gamma_1} & \frac{\partial b_0}{\partial \gamma_2} \\ \frac{\partial b_1}{\partial \gamma_1} & \frac{\partial b_1}{\partial \gamma_2} \end{bmatrix} = \begin{bmatrix} 2\gamma_1 & 2\gamma_2 \\ -2 & 0 \end{bmatrix} = \begin{bmatrix} 2r\cos\theta & 2r\sin\theta \\ -2 & 0 \end{bmatrix}$$

Putting this together with our previous result yields

$$\begin{aligned} \Delta r &= \beta_1 \boldsymbol{Q}_1 \boldsymbol{C} \Delta\boldsymbol{\gamma} \\ &= -\frac{1}{2r\sin\theta} \begin{bmatrix} -\sin\theta & 0 \end{bmatrix} \begin{bmatrix} 2r\cos\theta & 2r\sin\theta \\ -2 & 0 \end{bmatrix} \begin{bmatrix} \Delta\gamma_1 \\ \Delta\gamma_2 \end{bmatrix} \\ &= \Delta\gamma_1 \cos\theta + \Delta\gamma_2 \sin\theta \end{aligned}$$

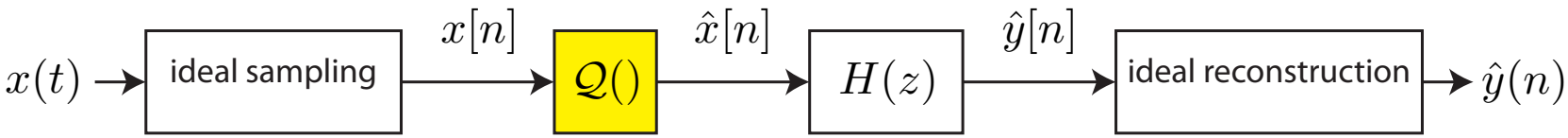
and

$$\Delta\theta = -\frac{\Delta\gamma_1}{r}\sin\theta + \frac{\Delta\gamma_2}{r}\cos\theta \qquad \text{(homework problem)}$$

Observations:

- Angular pole displacement is still sensitive when r is small (intuitive).
- Neither displacement is sensitive to angle of original poles.
- These results are consistent with the figure on slide 26.

Part II: Effect of Input Quantization (Section 12.5)



Here we assume $H(z)$ is an LTI system (no overflow or significant quantization errors) and focus our attention on understanding the effect of input quantization.

Recall the ideal sampling relationship

$$x[n] = x(nT)$$

and the quantizer relationship

$$\hat{x}[n] = \mathcal{Q}(x[n]) = x[n] + \epsilon[n]$$

where $\epsilon[n]$ is the quantization error.

Statistical Input Quantization Error Analysis (1 of 4)

Under the assumptions

1. the ADC is a rounding quantizer with step size δ and
2. the input is bounded to the ADC full-scale range (no input overflow)

we can bound the quantization error $\epsilon[n] = \hat{x}[n] - x[n]$ as

$$-\frac{\delta}{2} < \epsilon[n] \leq \frac{\delta}{2}.$$

We further assume the quantization error sequence $\{\epsilon[n]\}$ is a random sequence with the following statistical properties:

1. Each quantization error $\epsilon[n]$ is uniformly distributed on $\left[-\frac{\delta}{2}, \frac{\delta}{2}\right]$.
2. The quantization error $\epsilon[n]$ is independent of $\epsilon[m]$ for all $n \neq m$.
3. The quantization error $\epsilon[n]$ is independent of $x[m]$ for all n and m.

In other words, the quantization error sequence $\{\epsilon[n]\}$ is an independent and identically distributed random sequence, also independent of $\{x[n]\}$.

Statistical Input Quantization Error Analysis (2 of 4)

What is the mean of $\epsilon[n]$?

What is the variance of $\epsilon[n]$?

Note that the variance of a zero-mean random sequence is sometimes called the "power" of the sequence.

We define the **signal to quantization noise** ratio as

$$\begin{aligned}\text{SQNR(dB)} &= 10\log_{10}\left(\frac{\text{signal power}}{\text{quantization noise power}}\right) \\ &= 10\log_{10}\left(\frac{\sigma_x^2}{\sigma_\epsilon^2}\right)\end{aligned}$$

Signal power: If $x[n]$ is reasonably-modeled as a independent, identically distributed random sequence with each sample uniformly distributed on $[-A, A]$ with $0 < A \leq A_{max}$, what is the variance of $x[n]$?

Statistical Input Quantization Error Analysis (3 of 4)

Putting it all together, we have:

$$\sigma_\epsilon^2 = \frac{\delta^2}{12} = \frac{1}{12} \cdot \left(\frac{R_{FS}}{2^{b+1}} \right)^2 = \frac{R_{FS}^2}{48 \cdot 2^{2b}}$$

and

$$\sigma_x^2 = \frac{A^2}{3}$$

hence

$$\text{SQNR(dB)} = 10 \log_{10} \left(\frac{16 A^2 2^{2b}}{R_{FS}^2} \right)$$

If the input signal is scaled so that $A = \frac{\alpha}{2} R_{FS}$ with $0 < \alpha \leq 1$, then $R_{FS} = \frac{2A}{\alpha}$ and we have

$$\text{SQNR(dB)} = 10 \log_{10} \left(4 \cdot 2^{2b} \cdot \alpha^2 \right) = 6.02b + 6.02 + 20 \log_{10}(\alpha)$$

Statistical Input Quantization Error Analysis (4 of 4)

$$\boxed{\text{SQNR(dB)} = 6.02b + 6.02 + 20\log_{10}(\alpha)}$$

where α represents the input scaling with $\alpha = 1$ corresponding to full scale.

Remarks:

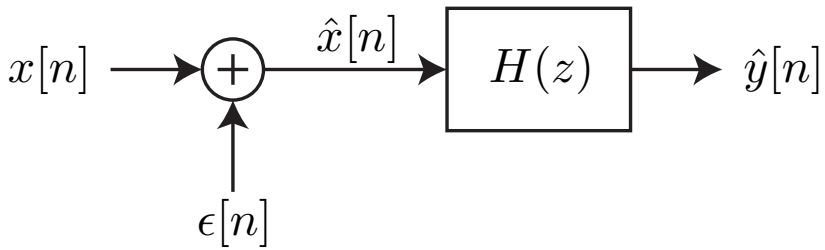
1. SQNR is maximized when $\alpha = 1$. This corresponds to scaling the input so that the signal uses the full range of the quantizer but does not overflow.
2. Setting $\alpha > 1$ appears to make SQNR even better, but our analysis does not account for overflow which actually causes SQNR to become bad very quickly for $\alpha > 1$.
3. Adding one more bit to your quantizer increases the SQNR by approximately 6dB. So a 12-bit quantizer has an SQNR about 24dB better than an 8-bit quantizer.
4. Compact discs use 16-bit quantization. If the signal uses half of the full scale, i.e. $\alpha = 1/2$, what is the SQNR?

Propagation of Input Quantization Noise to Filter Output

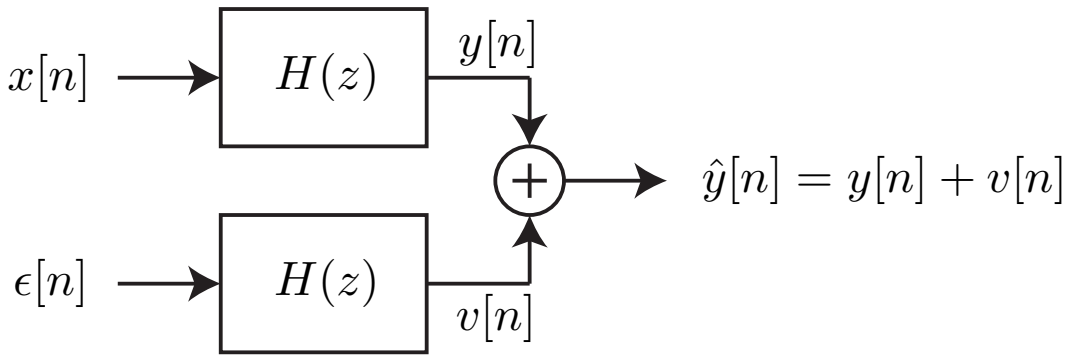
Since the quantized input to $H(z)$ can be written as

$$\hat{x}[n] = x[n] + \epsilon[n]$$

we can think of input quantization as shown in below.



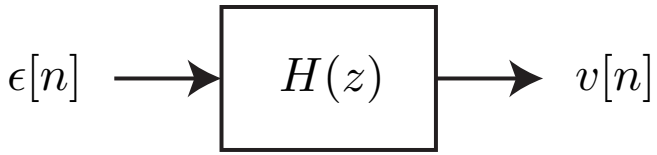
Since $H(z)$ is linear, we can use the principle of superposition to analyze the effect of $x[n]$ and $\epsilon[n]$ separately. In other words, we can analyze



and look specifically at the properties of $v[n]$.

Propagation of Input Quantization Noise to Filter Output

We now focus on analyzing



Since $H(z)$ is an LTI system and $\{\epsilon[n]\}$ is a zero-mean independent random sequence, i.e. white noise, we have the results (ECE502):

- Output mean:

$$\mu_v = 0$$

- Output variance:

$$\sigma_v^2 = \sigma_\epsilon^2 \cdot \frac{1}{2\pi} \int_{-\pi}^{\pi} |H(\omega)|^2 \, d\omega = \sigma_\epsilon^2 \cdot \sum_{n=-\infty}^{\infty} |h[n]|^2$$

Your textbook also provides method for "Algebraic Computation of Output Noise Variance" in Section 12.5.5 that might be easier to compute in some cases.

Propagation of Input Quantization Noise to Filter Output

Example:

$$H(z) = \frac{1}{1 - az^{-1}}$$

with $|a| < 1$ and ROC $|z| > |a|$. We can compute the output variance

$$\begin{aligned}
\sigma_v^2 &= \sigma_\epsilon^2 \cdot \frac{1}{2\pi} \int_{-\pi}^{\pi} |H(\omega)|^2 \, d\omega \\
&= \sigma_\epsilon^2 \cdot \frac{1}{2\pi} \int_{-\pi}^{\pi} \left| \frac{1}{1 - ae^{-j\omega}} \right|^2 d\omega \\
&= \text{difficult integral}
\end{aligned}$$

It is easier to note that $h[n] = a^n \mu[n]$ and compute

$$\sigma_v^2 = \sigma_\epsilon^2 \cdot \sum_{n=-\infty}^{\infty} |h[n]|^2 = \frac{\sigma_\epsilon^2}{1 - |\alpha|^2}.$$

What happens when α gets close to the unit circle?

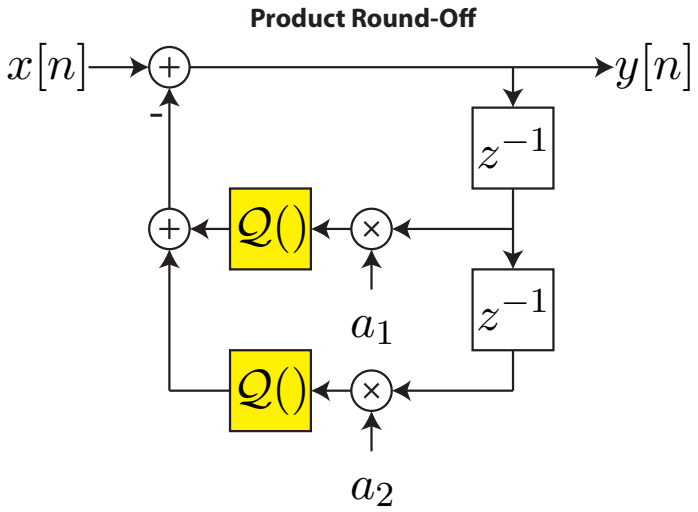
Matlab Simulation of Input Quantization Noise Propagation

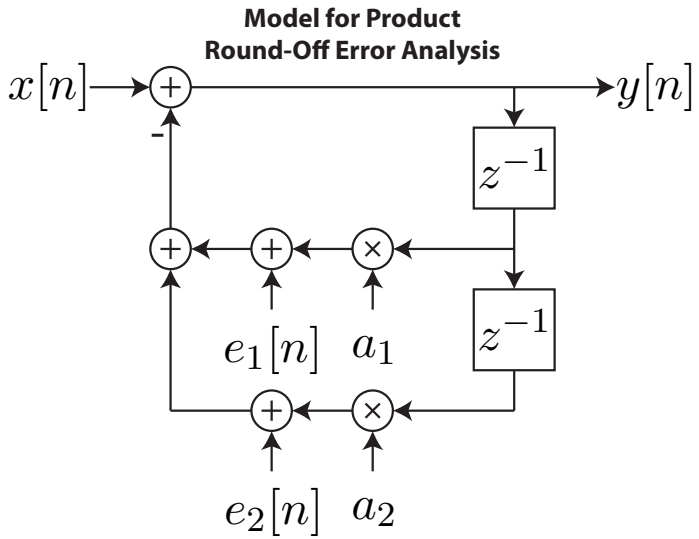
```
% output noise variance via simulation
% DRB ECE503 Spring 2012
delta = 0.1;              % quantizer step size
a = 0.8;                  % filter parameter
num = [1 0];
den = [1 -a];
N = 1e5;

% generate input quantization noise sequence
e = rand(1,N)*delta-delta/2;
disp(['Input noise variance          : ' num2str(var(e))]);

% filter
v = filter(num,den,e);

% compute output noise variance
disp(['Output noise variance         : ' num2str(var(v))]);
disp(['Ratio                         : ' num2str(var(v)/var(e))]);
```

Part III: Effect of Product Round-Off

When you take the product of two b-bit fixed point numbers, the result requires $2b$ bits to store. We typically round off the least significant bits of the product, which leads to another source of quantization error in finite precision filters. How does this error affect the filter output?

To illustrate the main idea, consider the following DF-II realization of an all-pole second-order IIR filter:





Effect of Product Round-Off

To facilitate analysis, the product round-off quantization errors are modeled as random sequences just like input quantization errors:

1. Each product round-off error $e_\ell[n]$ is uniformly distributed on $\left[-\frac{\delta}{2}, \frac{\delta}{2}\right]$.
2. The product round-off error $e_\ell[n]$ is independent of $e_\ell[m]$ for all $n \neq m$.
3. The product round-off error $e_\ell[n]$ is independent of $x[m]$ for all n and m.
4. The product round-off error $e_\ell[n]$ is independent of $e_k[m]$ for all $\ell \neq k$.

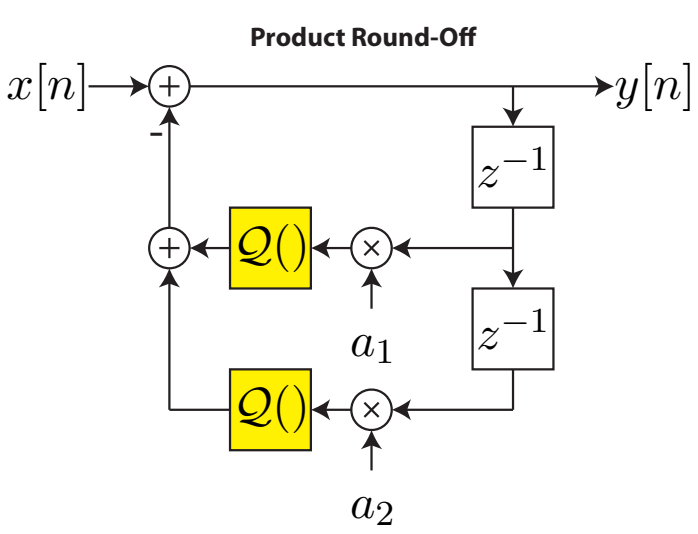
Procedure: Denote the round-off noise variance σ_0^2 (assumed the same for all products). For each product round-off error source $\ell = 1, \ldots, L$:

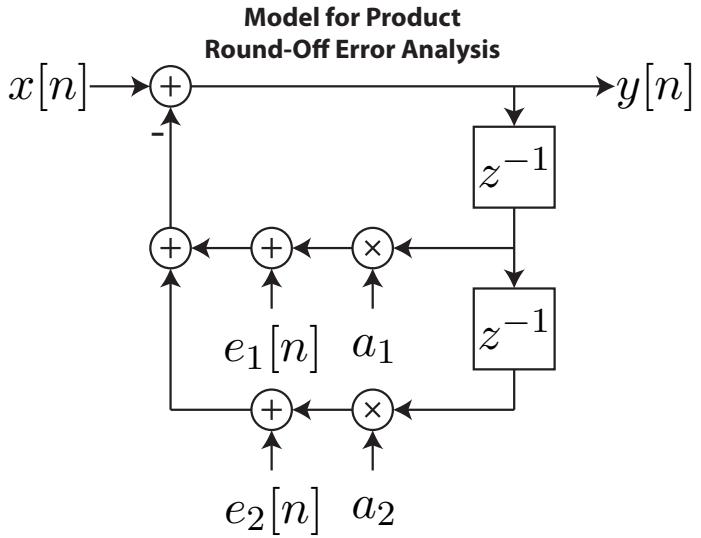
1. Determine the "noise transfer function" $G_\ell(z) = \frac{Y(z)}{E_\ell(z)}$ (you should assume all other sources including the input are set to zero).
2. Compute the output noise variance σ_ℓ^2 using same approach as before

$$\sigma_\ell^2 = \sigma_0^2 \cdot \frac{1}{2\pi} \int_{-\pi}^{\pi} |G_\ell(\omega)|^2 \, d\omega = \sigma_0^2 \cdot \sum_{n=-\infty}^{\infty} |g_\ell[n]|^2$$

The total round-off noise variance at output is then $\sigma_{tot}^2 = \sum_{\ell=1}^{L} \sigma_\ell^2$.

Effect of Product Round-Off: Example





We denote the round-off noise variance for both products as σ_0^2. By inspection, we can determine

$$G_1(z) = G_2(z) = \frac{1}{1 + a_1 z^{-1} + a_2 z^{-2}} = H(z)$$

Hence, assuming a_1 and a_2 are such that $H(z)$ is stable,

$$\sigma_{tot}^2 = 2\sigma_0^2 \frac{1}{2\pi} \int_{-\pi}^{\pi} |G_1(\omega)|^2 \, d\omega = 2\sigma_0^2 \left(\frac{1 + a_2}{1 - a_2} \right) \left(\frac{1}{1 + 2a_2 + a_2^2 - a_1^2} \right)$$

Conclusions

1. Several sources of quantization error can affect the behavior of filters when realized with finite precision arithmetic:
   - Coefficient quantization error.
   - Input quantization error.
   - Product round-off quantization error.
   - Overflow in sums (not covered).
   - Output quantization (not covered).
2. Analytical techniques can inform a good design.
3. All of our analysis assumed no overflow. Overflow causes massive quantization errors that are usually devastating to filter performance.
4. Simulation techniques can be used to confirm analysis and test effect of simultaneous error sources.