ECE 8440 Unit 15

Interaction Between Scaling and Round-Off Noise (Example 6.13)

Consider a first order system with system function $H(z) = \dfrac{b}{1-az^{-1}}$.

An implementation of the above system with scaling added is shown below:

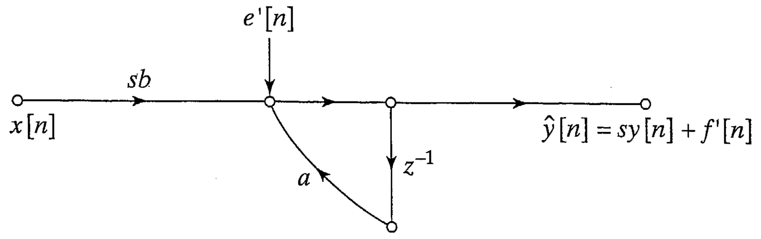


Figure 6.63 Scaled first-order system.

Assume that the input to the above system is a white noise signal with amplitudes uniformly distributed between -1 and 1.

The total average power (signal variance) of this input is

$$E(x^2(n)) = \int_{-1}^{1} \frac{1}{2} x^2 dx = \frac{1}{2} \frac{x^3}{3} \bigg|_{-1}^{1} = \frac{1}{3}.$$

The unit sample response of the above system, without scaling, is

$$h(n) = b\ a^n\ u(n).$$

Because of the simple expression for h(n), it is easy to base our scaling on the bound that involves summing the magnitude of h(n) values. (This is the most conservative bound.)

To prevent overflow, choose the scale factor to satisfy

$$s = \frac{1}{\sum_{n=0}^{\infty} |h(n)|} = \frac{1}{\sum_{n=0}^{\infty} |b||a|^n} = \frac{1-|a|}{|b|} \qquad (\text{where } X_m = 1)$$

In an example in Unit 14 which did not include any scaling (Example 6.11 from the text), we found that that the output noise power due to round-off error for the system of this example is

$$\sigma_f^2 = 2\frac{2^{-2B}}{12}\left(\frac{1}{1-|a|^2}\right).$$

Since scaling of the input does not affect the output noise power due to internal round-off error, we know that for the scaling case,

$$\sigma_{f'}^2 = \sigma_f^2 = 2\frac{2^{-2B}}{12}\left(\frac{1}{1-a^2}\right).$$

(The primed notation of $\sigma_{f'}^2$ is for the "with scaling" case, and the unprimed notation of σ_f^2 is for the "without scaling" case.)

For the current example (example 6.13) , the "signal" input is white noise with variance of (1/3) which passes though a multiplier of sb before reaching the insertion point of round-off error in the example of Unit 14. Therefore, the power of the "signal" part of the output is

$$\sigma_{y'}^2 = \frac{1}{3}s^2b^2\left(\frac{1}{1-a^2}\right).$$

If no scaling is used, the multiplier is b instead of sb , and the corresponding power of the "signal" part of the output would be

$$\sigma_y^2 = \frac{1}{3}b^2\left(\frac{1}{1-a^2}\right).$$

Since $\sigma_{y'}^2 = s^2\sigma_y^2$ and $\sigma_{f'}^2 = \sigma_{f'}^2$, the signal-to-noise ratio of the "with scaling" case is

$$\frac{\sigma_{y'}^2}{\sigma_{f'}^2} = s^2\frac{\sigma_y^2}{\sigma_f^2} = \left(\frac{1-|a|}{|b|}\right)^2\frac{\sigma_y^2}{\sigma_f^2}.$$

Note that as the pole at z = a approaches z = 1, the signal-to-noise ratio decreases toward zero.

Although the scaling prevents overflow, it reduces the size of the signal and does not affect the size of the round-off error.

Example of Analysis of a Cascade IIR Structure

Consider the following design specifications for an elliptic low-pass filter:

$$0.99 \le |H(e^{j\omega})| \le 1.01, \qquad |\omega| \le 0.5\pi$$
$$|H(e^{j\omega})| \le 0.01, \qquad 0.56\pi \le |\omega| \le \pi$$

The design process results in 6-th order filter, whose cascade implementation is represented by the following equation:

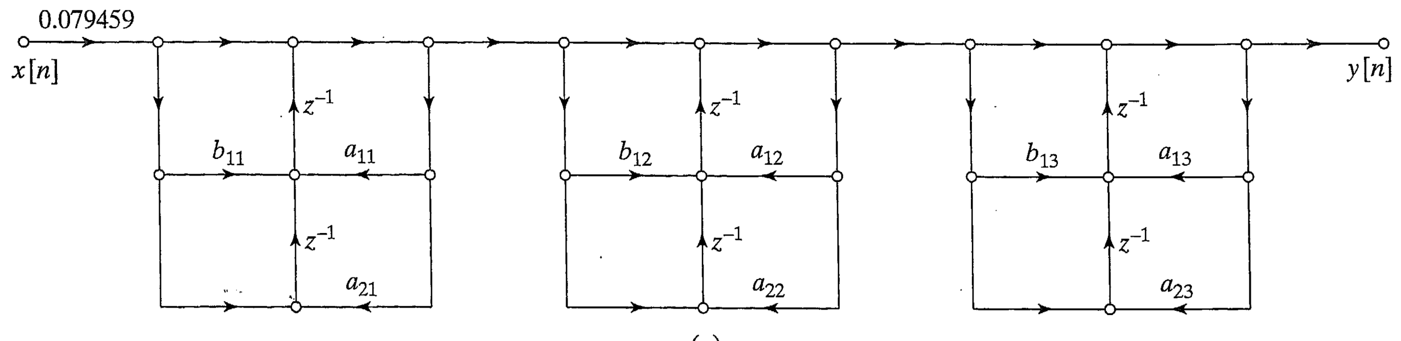
$$H(z) = 0.079459\prod_{k=1}^{3}\frac{(1-b_{1k}z^{-1}+z^{-2})}{(1-a_{1k}z^{-1}+a_{2k}z^{-2})} = 0.079459\prod_{k=1}^{3}H_k(z).$$

The coefficient values for each of the three second order sections are shown in the table below:

TABLE 6.6 COEFFICIENTS FOR ELLIPTIC LOWPASS FILTER IN CASCADE FORM

k	a_{1k}	a_{2k}	b_{1k}
1	0.478882	−0.172150	1.719454
2	0.137787	−0.610077	0.781109
3	−0.054779	−0.902374	0.411452

An implementation consisting of a cascade of three second order transposed Direct Form II filters is shown below:



(a)

Note: Putting the gain factor of 0.079459 at the input side of the implementation, as shown in the figure, decreases the signal to noise ratio, since it scales all of the signal, but none of the round-off noise.

From this perspective, it would be best to put the gain factor at the output, so that it would affect the signal and the quantization the same. However, this could cause overflow at critical internal nodes.

A better approach is to distribute the gain throughout the three stages, possibly along with additional scaling, so that overflow is barely avoided (avoided by a small margin) at each critical node.

If distributed scaling is used, we can represent the overall system function as

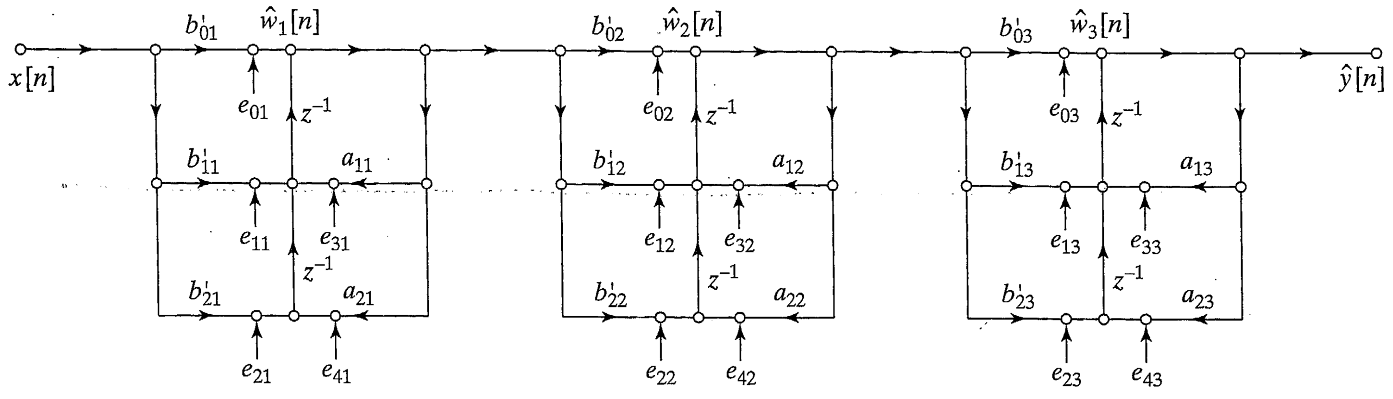
$$H(z) = s_1H_1(z)s_2H_2(z)s_3H_3(z) \text{ where } s_1s_2s_3 = 0.079459$$

Rather than introduce additional multipliers (and additional round-off error), the scaling multipliers can be combined with the filter coefficients in each of the three second order sections, as shown below:

$$H(z) = \prod_{k=1}^{3} \frac{(b'_{0k} - b'_{1k} z^{-1} + b'_{2k} z^{-2})}{(1 - a_{1k} z^{-1} + a_{2k} z^{-2})} = \prod_{k=1}^{3} H'_k(z)$$

where $b'_{0k} = b'_{2k} = s_k$ and $b'_{1k} = s_k b_{1k}$.

The resulting implementation is shown below:



(b)

<u>In each of the three second order sections, the five individual round-off noise sources can be combined into a single, "larger" noise source</u>, as shown in the following figure.

(These noise sources are still independent, although some are delayed before the point of combining them in the diagram.)

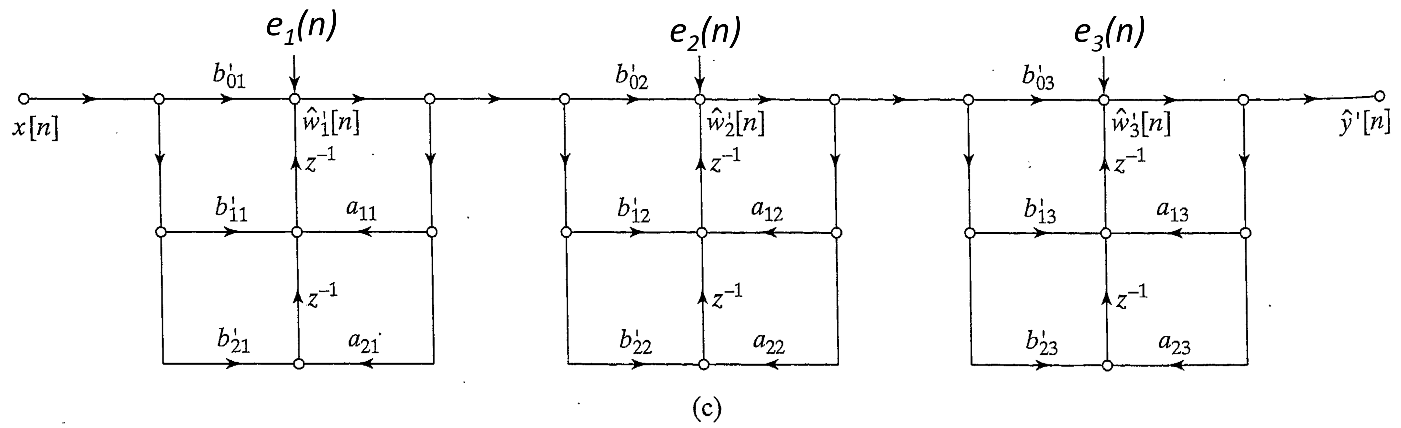


Figure 6.64 Models for 6th-order cascade system with transposed direct form II subsystems. (a) Infinite-precision model. (b) Linear-noise model for scaled system, showing quantization of individual multiplications. (c) Linear-noise model with noise sources combined.

Note that the combined round-off error injected into each of the three second order sections has a unique system function between it and the system output. The <u>power spectrum of the output noise due to all internal sources of round-off error can be expressed as</u>

$$P_{f'f'}(\omega) = 5\frac{2^{-2B}}{12}\left[\frac{s_2^2\,|H_2(e^{j\omega})|^2\, s_3^2\,|H_3(e^{j\omega})|^2}{|A_1(e^{j\omega})|^2} + \frac{s_3^2\,|H_3(e^{j\omega})|^2}{|A_2(e^{j\omega})|^2} + \frac{1}{|A_3(e^{j\omega})|^2}\right]$$

The <u>total output noise power</u> is

$$\sigma_f^2 = 5\frac{2^{-2B}}{12}\frac{1}{2\pi}\left[\int_{-\pi}^{\pi}\frac{s_2^2\,|H_2(e^{j\omega})|^2\, s_3^2\,|H_3(e^{j\omega})|^2}{|A_1(e^{j\omega})|^2}\,d\omega + \right.$$

$$\left.\frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{s_3^2\,|H_3(e^{j\omega})|^2}{|A_2(e^{j\omega})|^2}\,d\omega + \frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{1}{|A_3(e^{j\omega})|^2}\,d\omega\right]$$

(Alternatively, σ_f^2 could also be found using the z-transform approach of Appendix A5.)
If a double-length accumulator is used, it would be necessary to quantize only the output $\hat{w}_k(n)$ of each filter section and sums that feed a delay element within each filter section.

Using a double-length accumulator would reduce the factor of 5 in the above expression for σ_f^2 to 3.

In addition, if double length registers are available to implement the delay elements, only the output $\hat{w}_k(n)$ of each of the three filter sections has to be rounded off. In this case, the factor of 5 in the above equation would be reduced further to unity.

The gain factors s_1, s_2, and s_3 can be selected one at a time to prevent overflow at the critical nodes in the cascade implementation. For example, we could follow the following procedure:

Step 1: Choose s_1 so that $s_1 \max\limits_{|\omega|<\pi} |H_1(e^{j\omega})| < 1$

Step 2: Choose s_2 so that $s_1 s_2 \max\limits_{|\omega|<\pi} |H_1(e^{j\omega}) H_2(e^{j\omega})| < 1$

Step 3: Choose s_3 so that $s_1 s_2 s_3 = 0.079459$.

For the 6-th order elliptic filter whose designed coefficients are shown in Table 6.6, the values of the three scale factors determine by the above procedure are:

$s_1 = 0.186447$ $s_2 = 0.529236$ and $s_3 = 0.805267$

Recall that for a filter having N second order sections, there are N! ways to pair the poles and zeros.

In addition, there are N! ways to order the resulting second order sections.

Furthermore, each second order section can be realized using Direct Form I, Direct Form II, the transpose of Direct Form I, or the transpose of Direct Form II.

Therefore, the total number of ways to implement a cascade of N second order sections, using the above options, is $4(N!)^2$. For the above example for which N = 3, the total number of configurations is thus

$$4(N!)^2 = 4(3!)^2 = 144$$

For **N** = 4, this number would be

$$4(4!)^2 = 2{,}304$$

For **N** = 5, the number of configurations of second order sections would become

$$4(5!)^2 = 57{,}600$$

<u>"Rules of thumb" for selecting the desired configuration of second order section for a filter</u>:

1. Pair the pole that is closest to the unit circle with the zero that is closest to it.
2. Repeat Rule 1 until all poles and zeros have been paired.
3. Order the resulting second-order sections according to either increasing pole closeness to the unit circle OR decreasing pole closeness to the unit circle.

Rules 1 and 2 (the pairing rules) are based the fact that that poles near the unit circle cause high peak gain, and this can cause two types of problems:

- overflow
- amplification of quantization error

By pairing a nearby zero to a pole close to the unit circle, we minimize the high gain due to that pole.

Rule 3 is based on the following two contradictory considerations:

- Having a "high Q" pole near the input means that most of the internally generated round-off noise doesn't have to pass through it.

-On the other hand, a high Q pole near the input means that the input will probably have to be reduced by major scaling, to prevent overflow, and this will degrade the signal to noise ratio at the output.

Which of the above two considerations should be followed depends on the characteristics of the filter being implemented.

An example of pairing zeros with poles and of ordering the resulting second order sections for the 6th order elliptic filter previously discussed are shown in the figure below. In this figure, ordering is based on the "least peaked" to "most peaked" option.

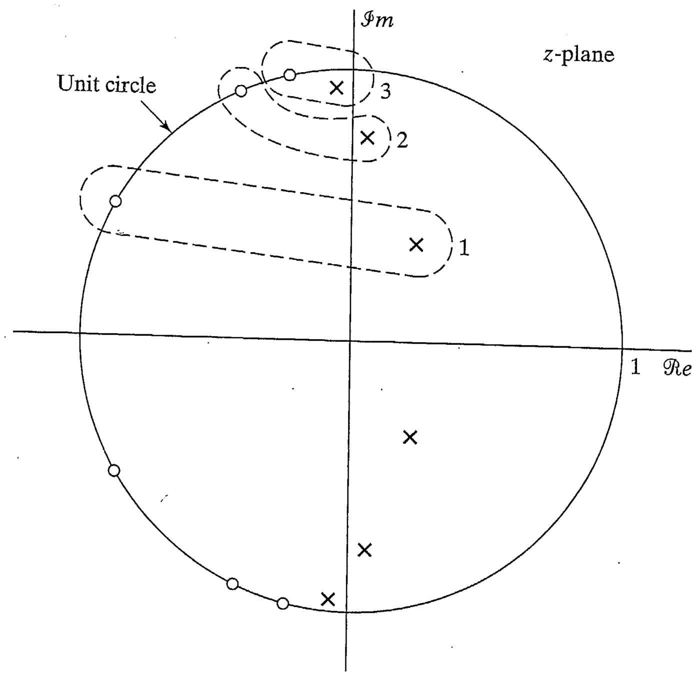
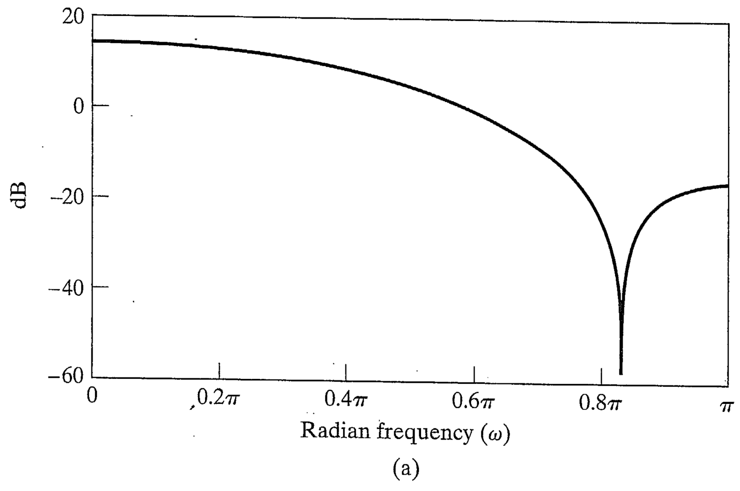
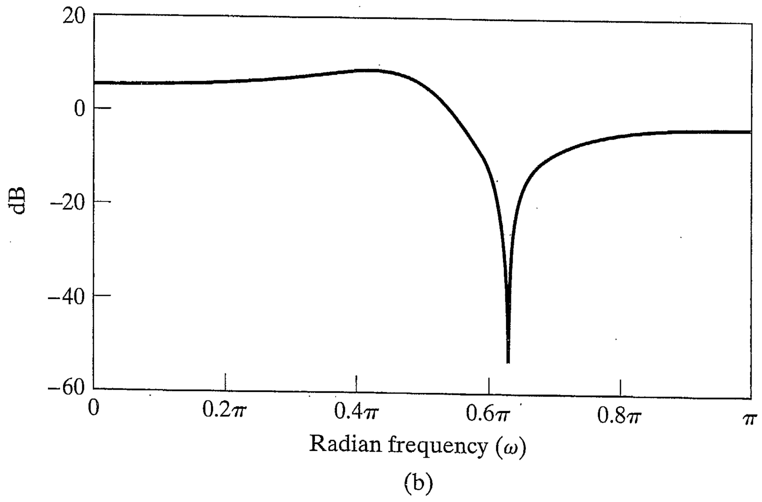


Figure 6.65 Pole-zero plot for sixth-order system if Figure 6.64, showing pairing of poles and zeros.

The following figure shows the individual frequency response of each of the second order sections of the overall filter:





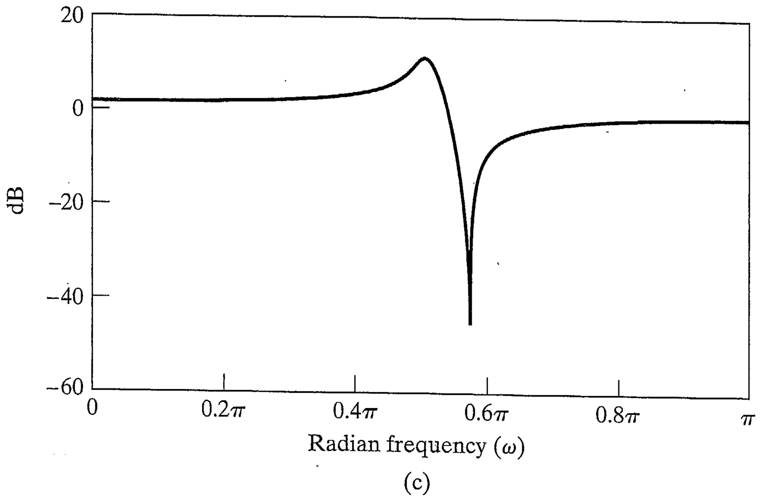
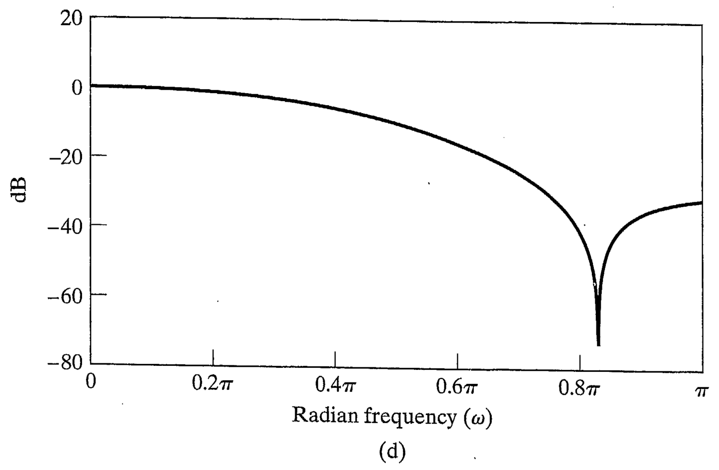
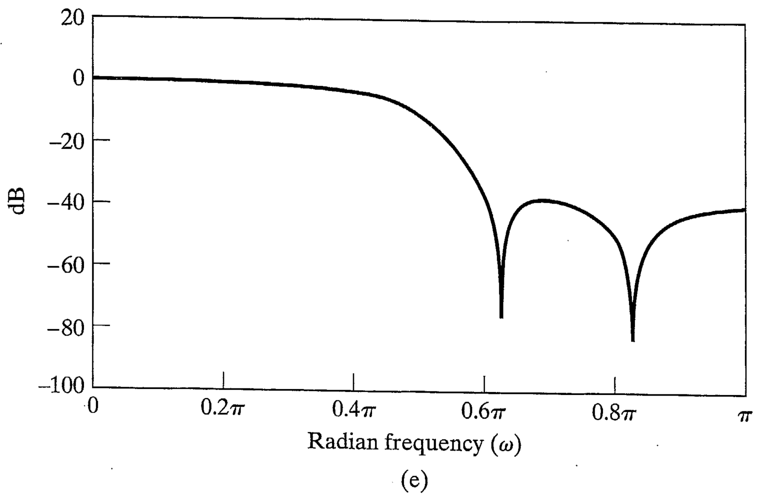


Figure 6.66 Frequency-response functions for example system

(a) $20\log_{10}\left|H_1\left(e^{j\omega}\right)\right|$
(b) $20\log_{10}\left|H_2\left(e^{j\omega}\right)\right|$
(c) $20\log_{10}\left|H_3\left(e^{j\omega}\right)\right|$

The following figure shows how the overall frequency response of the 6-th order filter is "built up" from the second order components:





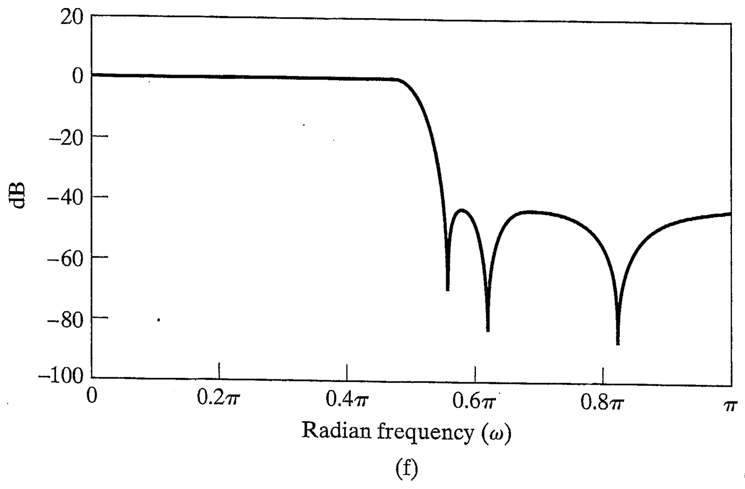


Figure 6.66 (continued)

(d) $20\log_{10}\left|H_1'\left(e^{j\omega}\right)\right|$

(e) $20\log_{10}\left|H_1'\left(e^{j\omega}\right)H_2'\left(e^{j\omega}\right)\right|$

(f) $20\log_{10}\left|H_1'\left(e^{j\omega}\right)H_2'\left(e^{j\omega}\right)H_3'\left(e^{j\omega}\right)\right| = 20\log_{10}\left|H'\left(e^{j\omega}\right)\right|$

In the previous figure, note that the previous method of scaling has successfully kept the maximum gain from the input to the output of any subsystem less than unity.

Figure 6.67 below shows the power spectrum of the output noise for a "123" ordering (least peaked to most peaked) and for a "321" ordering (most peaked to least peaked).

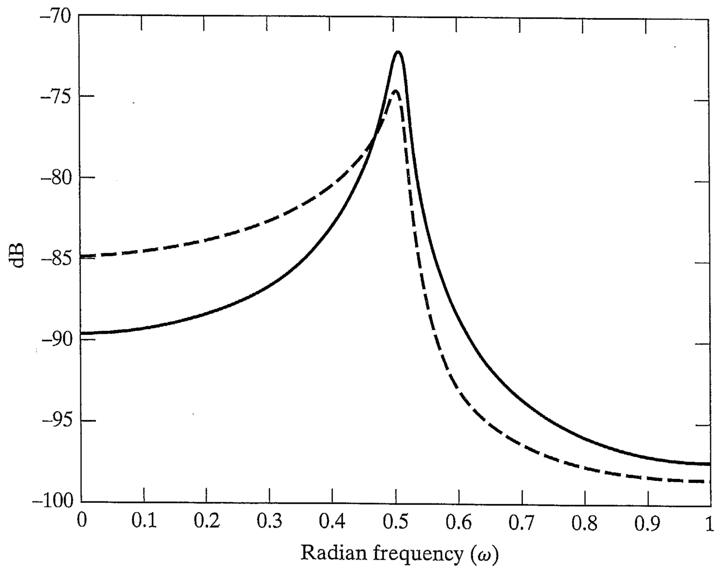


Figure 6.67 Output noise power Spectrum for 123 ordering (solid line) And 321 ordering of second-order Sections.

The total noise power for the two above orderings is based on the integral of each curve in the above figure, and is therefore about the same, in this example.

Parallel Form

It is been shown that an implementation consisting of a parallel second order sections has total output noise power similar to that of the best pairing and order for the cascade form. However, due to implementation issues, the cascade form has been more popular in most applications.

Round-Off Noise in FIR Filters

The effect of round-off in FIR filters is a special case of the noise analysis for IIR filters for the case of N = 0. A signal flow diagram for an FIR filter showing injection of round-off noise is shown below:

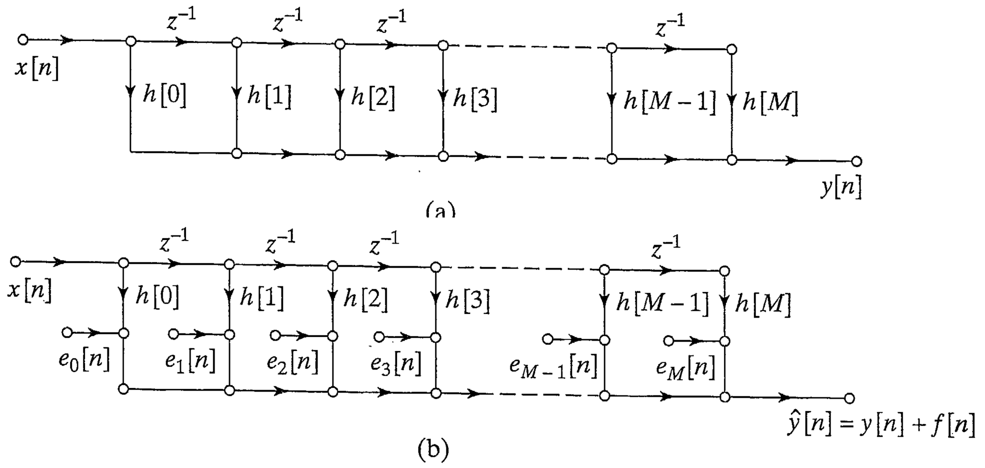


Figure 6.68 Direct-form of an FIR system. (a) Infinite-precision model. (b) Linear-noise model.

In the previous figure, all M + 1 noise sources are injected directly to the output. Therefore, the output noise variance is

$$\sigma_f^2 = (M+1)\frac{2^{-2B}}{12}.$$

However, if a double length accumulator is used, it is only necessary to round-off the output, and the factor of M+1 in the above expression would be reduced to unity.

<u>Overflow in FIR Filters</u>

Overflow in FIR filters can be prevented using the same scaling methods developed for IIR filters.

However, if two's complement arithmetic is used, we only need to be concerned with the size of the output, since all of the other sums are partial sums.

The scaling constant can be determined using any of the three methods developed for IIR filters, as shown in the following example.

<u>Example 6.14 (Scaling Considerations for the FIR System in Section 6.8.5)</u>

For this filter, a value of M = 27 was required to meet the design specifications. For this 27th order filter, the three candidate methods for selecting the scale factor are shown below:

$$\sum_{n=0}^{27} |h(n)| = 1.751352 \qquad \left(\sum_{n=0}^{27} |h(n)|^2\right)^{\frac{1}{2}} = 0.6794421$$

$$\max_{|\omega|<\pi} |H(e^{j\omega}) \approx 1.009$$

Based on the above, overflow is probably unlikely for most inputs, and scaling might not be used.
However, if absolutely no chance of overflow can be tolerated, scaling could be implemented by dividing each value of h(n) by 1.751352. (This assume $X_m = 1$.)

Note: We have seen that for generalized linear phase, the number of multiplications (and therefore, potential sources of round-off error) can be reduced by a factor of 2 by taking advantage of the symmetry of h(n).
However, most DSP processors have double-length accumulators with pipelined multipliers, so that direct form of implementation (without taking advantage of the symmetry of h(n)) is typically used.

Round-Off Error When Floating Point Arithmetic is Used

Floating point representation of numbers provides a means of representing a large range of values and involving a low level of quantization noise.

In floating point arithmetic, each value is represented using the format
$x = 2^c x_M$
The exponent c is called the characteristic.

The other component, x_M, is a fraction and is call the mantissa.
Both c and x_M are represented as fixed-point binary numbers.

When floating point numbers are multiplied, their characteristics are added and their mantissas are multiplied. Round-off error occurs when these mantissas are multiplied.

When floating point number are added, their characteristics must be adjusted to be the same which involves moving the binary point for one of the numbers.

This adjustment can introduce round-off error, and this error is inherently scaled by the value of the characteristic.

Thus, a quantized floating point number $\hat{x}$ can be represented in term of the desired value x and the round-off error ε as

$$\begin{aligned}\hat{x} &= 2^{c}(x_{M} + \varepsilon) \\ &= x + 2^{c}\varepsilon\end{aligned}$$

Zero-Input Limit Cycles in Fixed-Point Realizations of Digital Filters

A zero-input limit cycle occurs when the input to a LTI system goes to zero and remains at zero, but the output oscillates indefinitely in a periodic pattern or stays at a constant, non-zero value, due to round-off noise or overflow .

Example: (Limit cycle due to round-off error)

Consider the first-order system described by the following difference equations.

$y(n) = ay(n-1) + x(n)$ where a < 1.

Signal flow diagrams (a) with ideal arithmetic and (b) with round-off effects are shown below:

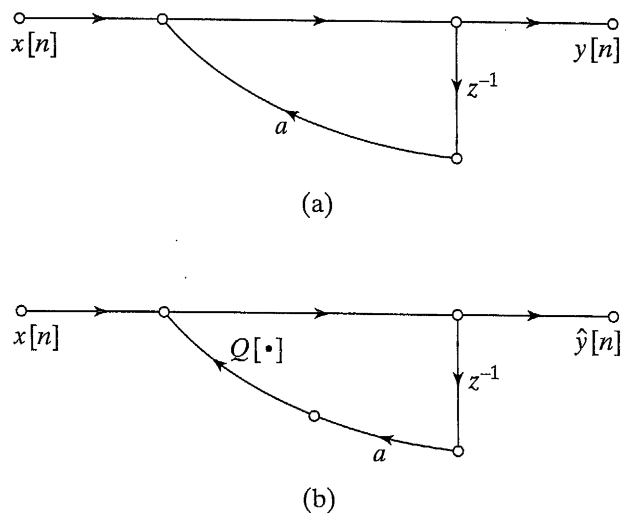


Figure 6.69 1st-order IIR system. (a) Infinite-precision linear system. (b)Nonlinear system due to quantization.

Because of round-off effects, the actual input/output relation for the system is

$\hat{y}(n) = Q[a\hat{y}(n-1)] + x(n)$

Assume that 4-bit fixed-point arithmetic is used to implement this filter and that when a number to be rounded is half way between representable values, the number is rounded up to the higher value.

Consider the case where $a = (1/2) = 0.100$, when represented as a two's complement fraction.

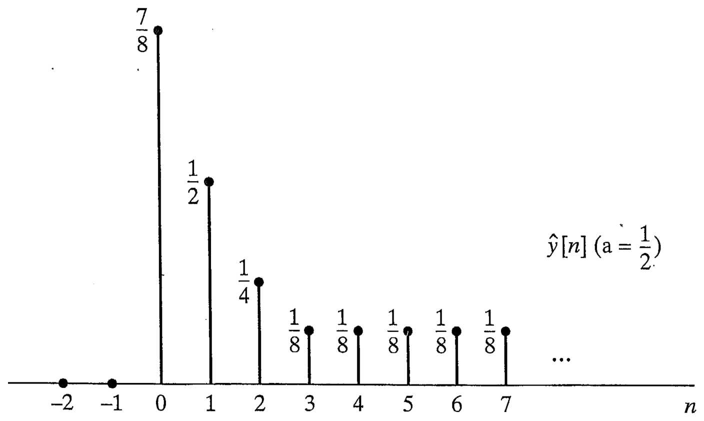
Consider the following input to the system:

$x(n) = (7/8)\ \delta(n) = 0.111\ \delta(n)$

Response of the filter to this input is as follows:

n	x(n)	$a\hat{y}(n-1)$	$Q[a\hat{y}(n-1)]$	$\hat{y}(n)$
0	0.111			0.111
1	0.000	0.011100	0.100	0.100
2	0.000	0.010000	0.010	0.010
3	0.000	0.001000	0.001	0.001
4	0.000	0.000100	0.001	0.001
5	0.000	0.000100	0.001	0.001

The limit cycle output corresponding to the above table is plotted below:



(a)

If the filter parameter "a" were equal to -(1/2) instead of (1/2), the following limit cycle output would occur:

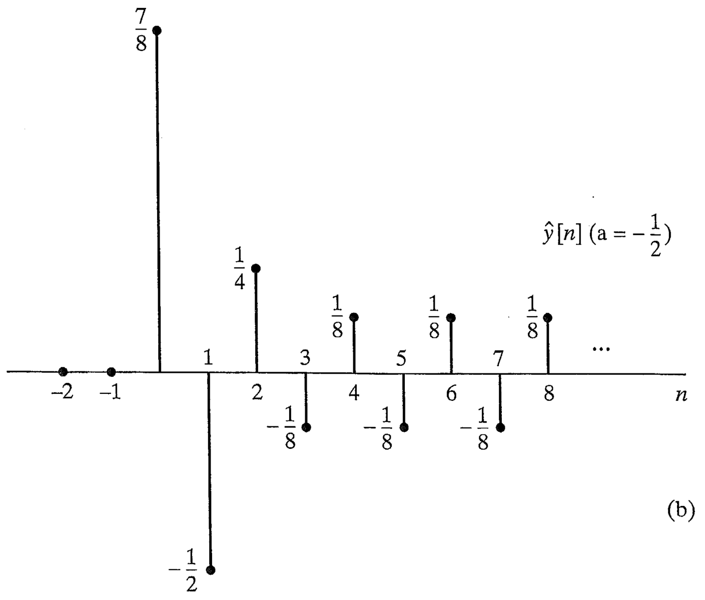


Figure 6.70 Response of the 1st-order System of Figure 6.69 to an impulse. (a) a = ½ (b) a = - ½

<u>Limit Cycles due to Overflow</u>

Consider the system represented by the following difference equation:

$$\hat{y}(n) = x(n) + Q[a_1\hat{y}(n-1)] + Q[a_2\hat{y}(n-2)$$

Assume the following values for filter parameters:

$a_1 = (3/4) = 0.110$ and $a_2 = -(3/4) = 1.010$

Also assume that input x(n) = 0 for all n ≥ 0 and that the following initial conditions are present:

$\hat{y}(-1) = (3/4) = 0.110$ and $\hat{y}(-2) = -(3/4) = 1.010$

Assume that if the value to be rounded is halfway between two available quantizer output values, the value which is larger <u>in magnitude</u> is selected as output. For example, a value of -9/16 is rounded to -10/16 = -5/8, not to -8/16 = -1/2.

The table below shows how limit cycles output of this filter for $0 \le n \le 3$.

n	$a_1\hat{y}(n-1)$	$Q[a_1\hat{y}(n-1)]$	$a_2\hat{y}(n-2)$	$Q[a_2\hat{y}(n-2]$	$\hat{y}(n)$
–2					1.010
–1					0.110
0	0.100100	0.101	0.100100	0.101	1.010 *
1	1.011100	1.011	1.011100	1.011	0.110 **
2	0.100100	0.101	0.100100	0.101	1.010 *
3	1.011100	1.011	1.011100	1.011	0.110 **

* The sum of two positive values produces a negative result $\Rightarrow$ Overflow

** The sum of two negative values produces a positive result $\Rightarrow$ Overflow

<u>Ways to Reduce the Likelihood of Limit Cycles in Digital Filters</u>

1. Use longer word length
2. Use double-length accumulator
3. Use implementation structures that do not support limit cycles.

<u>Note</u>: Limit cycles cannot occur in FIR filters, since FIR filters have no feedback paths