Bomar, B.W. “Finite Wordlength Effects” 
*Digital Signal Processing Handbook*
Ed. Vijay K. Madisetti and Douglas B. Williams 
Boca Raton: CRC Press LLC, 1999
3

Finite Wordlength Effects

3.1 Introduction

Practical digital filters must be implemented with finite precision numbers and arithmetic. As a result, both the filter coefficients and the filter input and output signals are in discrete form. This leads to four types of finite wordlength effects.

Discretization (quantization) of the filter coefficients has the effect of perturbing the location of the filter poles and zeroes. As a result, the actual filter response differs slightly from the ideal response. This deterministic frequency response error is referred to as coefficient quantization error.

The use of finite precision arithmetic makes it necessary to quantize filter calculations by rounding or truncation. Roundoff noise is that error in the filter output that results from rounding or truncating calculations within the filter. As the name implies, this error looks like low-level noise at the filter output.

Quantization of the filter calculations also renders the filter slightly nonlinear. For large signals this nonlinearity is negligible and roundoff noise is the major concern. However, for recursive filters with a zero or constant input, this nonlinearity can cause spurious oscillations called limit cycles.

With fixed-point arithmetic it is possible for filter calculations to overflow. The term overflow oscillation, sometimes also called adder overflow limit cycle, refers to a high-level oscillation that can exist in an otherwise stable filter due to the nonlinearity associated with the overflow of internal filter calculations.

In this chapter, we examine each of these finite wordlength effects. Both fixed-point and floating-point number representations are considered.

©1999 by CRC Press LLC
3.2 Number Representation

In digital signal processing, \((B+1)\)-bit fixed-point numbers are usually represented as two's-complement signed fractions in the format

\[ b_0 \cdot b_{-1}b_{-2} \cdots b_{-B} \]

The number represented is then

\[ X = -b_0 + b_{-1}2^{-1} + b_{-2}2^{-2} + \cdots + b_{-B}2^{-B} \] (3.1)

where \(b_0\) is the sign bit and the number range is \(-1 \leq X < 1\). The advantage of this representation is that the product of two numbers in the range from \(-1\) to 1 is another number in the same range.

Floating-point numbers are represented as

\[ X = (-1)^s m 2^c \] (3.2)

where \(s\) is the sign bit, \(m\) is the mantissa, and \(c\) is the characteristic or exponent. To make the representation of a number unique, the mantissa is normalized so that \(0.5 \leq m < 1\).

Although floating-point numbers are always represented in the form of (3.2), the way in which this representation is actually stored in a machine may differ. Since \(m \geq 0.5\), it is not necessary to store the \(2^{-1}\)-weight bit of \(m\), which is always set. Therefore, in practice numbers are usually stored as

\[ X = (-1)^s (0.5 + f) 2^c \] (3.3)

where \(f\) is an unsigned fraction, \(0 \leq f < 0.5\).

Most floating-point processors now use the IEEE Standard 754 32-bit floating-point format for storing numbers. According to this standard the exponent is stored as an unsigned integer \(p\) where

\[ p = c + 126 \] (3.4)

Therefore, a number is stored as

\[ X = (-1)^s (0.5 + f) 2^{p-126} \] (3.5)

where \(s\) is the sign bit, \(f\) is a 23-b unsigned fraction in the range \(0 \leq f < 0.5\), and \(p\) is an 8-b unsigned integer in the range \(0 \leq p \leq 255\). The total number of bits is \(1 + 23 + 8 = 32\). For example, in IEEE format \(3/4\) is written \((-1)^0 (0.5 + 0.25) 2^0\) so \(s = 0\), \(p = 126\), and \(f = 0.25\). The value \(X = 0\) is a unique case and is represented by all bits zero (i.e., \(s = 0\), \(f = 0\), and \(p = 0\)). Although the \(2^{-1}\)-weight mantissa bit is not actually stored, it does exist so the mantissa has 24 b plus a sign bit.

3.3 Fixed-Point Quantization Errors

In fixed-point arithmetic, a multiply doubles the number of significant bits. For example, the product of the two 5-b numbers 0.0011 and 0.1001 is the 10-b number 00.0001 110 11. The extra bit to the left of the decimal point can be discarded without introducing any error. However, the least significant four of the remaining bits must ultimately be discarded by some form of quantization so that the result can be stored to 5 b for use in other calculations. In the example above this results in 0.0010 (quantization by rounding) or 0.0001 (quantization by truncating). When a sum of products calculation is performed, the quantization can be performed either after each multiply or after all products have been summed with double-length precision.
We will examine three types of fixed-point quantization—rounding, truncation, and magnitude truncation. If \( X \) is an exact value, then the rounded value will be denoted \( Q_r(X) \), the truncated value \( Q_t(X) \), and the magnitude truncated value \( Q_{mt}(X) \). If the quantized value has \( B \) bits to the right of the decimal point, the quantization step size is

\[
\Delta = 2^{-B}
\] (3.6)

Since rounding selects the quantized value nearest the unquantized value, it gives a value which is never more than \( \pm \Delta / 2 \) away from the exact value. If we denote the rounding error by

\[
\epsilon_r = Q_r(X) - X
\] (3.7)

then

\[
-\frac{\Delta}{2} \leq \epsilon_r \leq \frac{\Delta}{2}
\] (3.8)

Truncation simply discards the low-order bits, giving a quantized value that is always less than or equal to the exact value so

\[
-\Delta < \epsilon_t \leq 0
\] (3.9)

Magnitude truncation chooses the nearest quantized value that has a magnitude less than or equal to the exact value so

\[
-\Delta < \epsilon_{mt} < \Delta
\] (3.10)

The error resulting from quantization can be modeled as a random variable uniformly distributed over the appropriate error range. Therefore, calculations with roundoff error can be considered error-free calculations that have been corrupted by additive white noise. The mean of this noise for rounding is

\[
m_{\epsilon_r} = E\{\epsilon_r\} = \frac{1}{\Delta} \int_{-\Delta/2}^{\Delta/2} \epsilon_r \, d\epsilon_r = 0
\] (3.11)

where \( E\{\} \) represents the operation of taking the expected value of a random variable. Similarly, the variance of the noise for rounding is

\[
\sigma_{\epsilon_r}^2 = E\{(\epsilon_r - m_{\epsilon_r})^2\} = \frac{1}{\Delta} \int_{-\Delta/2}^{\Delta/2} (\epsilon_r - m_{\epsilon_r})^2 \, d\epsilon_r = \frac{\Delta^2}{12}
\] (3.12)

Likewise, for truncation,

\[
m_{\epsilon_t} = E\{\epsilon_t\} = -\frac{\Delta}{2}
\]

\[
\sigma_{\epsilon_t}^2 = E\{(\epsilon_t - m_{\epsilon_t})^2\} = \frac{\Delta^2}{12}
\] (3.13)

and, for magnitude truncation

\[
m_{\epsilon_{mt}} = E\{\epsilon_{mt}\} = 0
\]

\[
\sigma_{\epsilon_{mt}}^2 = E\{(\epsilon_{mt} - m_{\epsilon_{mt}})^2\} = \frac{\Delta^2}{3}
\] (3.14)
3.4 Floating-Point Quantization Errors

With floating-point arithmetic, it is necessary to quantize after both multiplications and additions. The addition quantization arises because, prior to addition, the mantissa of the smaller number in the sum is shifted right until the exponent of both numbers is the same. In general, this gives a sum mantissa that is too long and so must be quantized.

We will assume that quantization in floating-point arithmetic is performed by rounding. Because of the exponent in floating-point arithmetic, it is the relative error that is important. The relative error is defined as
\[ e_r = \frac{Q_r(X) - X}{X} = \frac{\epsilon_r}{X} \] (3.15)

Since \( X = (-1)^i m 2^c \), \( Q_r(X) = (-1)^i Q_r(m) 2^c \) and
\[ e_r = \frac{Q_r(m) - m}{m} = \frac{\epsilon}{m} \] (3.16)

If the quantized mantissa has \( B \) bits to the right of the decimal point, \( |\epsilon| < \Delta / 2 \) where, as before, \( \Delta = 2^{-B} \). Therefore, since \( 0.5 < m < 1 \),
\[ |\epsilon_r| < \Delta \] (3.17)

If we assume that \( \epsilon \) is uniformly distributed over the range from \(-\Delta / 2\) to \(\Delta / 2\) and \( m \) is uniformly distributed over \(0.5\) to \(1\),
\[ m_{\epsilon_r} = E \left\{ \frac{\epsilon}{m} \right\} = 0 \]
\[ \sigma_{\epsilon_r}^2 = E \left\{ \left( \frac{\epsilon}{m} \right)^2 \right\} = \frac{2}{\Delta} \int_{-\Delta/2}^{\Delta/2} \frac{\epsilon^2}{m^2} d\epsilon dm \]
\[ = \frac{\Delta^2}{6} = (0.167)2^{-2B} \] (3.18)

In practice, the distribution of \( m \) is not exactly uniform. Actual measurements of roundoff noise in [1] suggested that
\[ \sigma_{\epsilon_r}^2 \approx 0.23 \Delta^2 \] (3.19)
while a detailed theoretical and experimental analysis in [2] determined
\[ \sigma_{\epsilon_r}^2 \approx 0.18 \Delta^2 \] (3.20)

From (3.15) we can represent a quantized floating-point value in terms of the unquantized value and the random variable \( \epsilon_r \) using
\[ Q_r(X) = X(1 + \epsilon_r) \] (3.21)

Therefore, the finite-precision product \( X_1 X_2 \) and the sum \( X_1 + X_2 \) can be written
\[ fl(X_1 X_2) = X_1 X_2 (1 + \epsilon_r) \] (3.22)
and
\[ fl(X_1 + X_2) = (X_1 + X_2) (1 + \epsilon_r) \] (3.23)
where \( \epsilon_r \) is zero-mean with the variance of (3.20).
3.5 Roundoff Noise

To determine the roundoff noise at the output of a digital filter we will assume that the noise due to a quantization is stationary, white, and uncorrelated with the filter input, output, and internal variables. This assumption is good if the filter input changes from sample to sample in a sufficiently complex manner. It is not valid for zero or constant inputs for which the effects of rounding are analyzed from a limit cycle perspective.

To satisfy the assumption of a sufficiently complex input, roundoff noise in digital filters is often calculated for the case of a zero-mean white noise filter input signal $x(n)$ of variance $\sigma_x^2$. This simplifies calculation of the output roundoff noise because expected values of the form $E(x(n)x(n-k))$ are zero for $k \neq 0$ and give $\sigma_x^2$ when $k = 0$. This approach to analysis has been found to give estimates of the output roundoff noise that are close to the noise actually observed for other input signals.

Another assumption that will be made in calculating roundoff noise is that the product of two quantization errors is zero. To justify this assumption, consider the case of a 16-b fixed-point processor. In this case a quantization error is of the order $2^{-15}$, while the product of two quantization errors is of the order $2^{-30}$, which is negligible by comparison.

If a linear system with impulse response $g(n)$ is excited by white noise with mean $m_x$ and variance $\sigma_x^2$, the output is noise of mean $m_y = m_x \sum_{n=-\infty}^{\infty} g(n)$ (3.24) and variance $\sigma_y^2 = \sigma_x^2 \sum_{n=-\infty}^{\infty} g^2(n)$ (3.25). Therefore, if $g(n)$ is the impulse response from the point where a roundoff takes place to the filter output, the contribution of that roundoff to the variance (mean-square value) of the output roundoff noise is given by (3.25) with $\sigma_x^2$ replaced with the variance of the roundoff. If there is more than one source of roundoff error in the filter, it is assumed that the errors are uncorrelated so the output noise variance is simply the sum of the contributions from each source.

3.5.1 Roundoff Noise in FIR Filters

The simplest case to analyze is a finite impulse response (FIR) filter realized via the convolution summation

$$y(n) = \sum_{k=0}^{N-1} h(k)x(n-k)$$

(3.26)

When fixed-point arithmetic is used and quantization is performed after each multiply, the result of the $N$ multiplies is $N$-times the quantization noise of a single multiply. For example, rounding after each multiply gives, from (3.6) and (3.12), an output noise variance of

$$\sigma_o^2 = N \frac{2^{-2B}}{12}$$

(3.27)

Virtually all digital signal processor integrated circuits contain one or more double-length accumulator registers which permit the sum-of-products in (3.26) to be accumulated without quantization. In this case only a single quantization is necessary following the summation and

$$\sigma_o^2 = \frac{2^{-2B}}{12}$$

(3.28)

©1999 by CRC Press LLC
For the floating-point roundoff noise case we will consider (3.26) for \( N = 4 \) and then generalize the result to other values of \( N \). The finite-precision output can be written as the exact output plus an error term \( e(n) \). Thus,

\[
y(n) + e(n) = \left( [h(0)x(n)[1 + \varepsilon_1(n)]
+ h(1)x(n-1)[1 + \varepsilon_2(n)][1 + \varepsilon_3(n)]
+ h(2)x(n-2)[1 + \varepsilon_4(n)][1 + \varepsilon_5(n)]
+ h(3)x(n-3)[1 + \varepsilon_6(n)][1 + \varepsilon_7(n)]\right)
\]

(3.29)

In (3.29), \( \varepsilon_1(n) \) represents the error in the first product, \( \varepsilon_2(n) \) the error in the second product, \( \varepsilon_3(n) \) the error in the first addition, etc. Notice that it has been assumed that the products are summed in the order implied by the summation of (3.26).

Expanding (3.29), ignoring products of error terms, and recognizing \( y(n) \) gives

\[
e(n) = h(0)x(n)[\varepsilon_1(n) + \varepsilon_3(n) + \varepsilon_5(n) + \varepsilon_7(n)]
+ h(1)x(n-1)[\varepsilon_2(n) + \varepsilon_3(n) + \varepsilon_5(n) + \varepsilon_7(n)]
+ h(2)x(n-2)[\varepsilon_4(n) + \varepsilon_5(n) + \varepsilon_7(n)]
+ h(3)x(n-3)[\varepsilon_6(n) + \varepsilon_7(n)]
\]

(3.30)

Assuming that the input is white noise of variance \( \sigma_x^2 \) so that \( E[x(n)x(n-k)] \) is zero for \( k \neq 0 \), and assuming that the errors are uncorrelated,

\[
E[e^2(n)] = [4h^2(0) + 4h^2(1) + 3h^2(2) + 2h^2(3)]\sigma_x^2\sigma_{\varepsilon_1}^2
\]

(3.31)

In general, for any \( N \),

\[
\sigma_o^2 = E[e^2(n)] = \left[ Nh^2(0) + \sum_{k=1}^{N-1} (N + 1 - k)h^2(k) \right] \sigma_x^2\sigma_{\varepsilon_1}^2
\]

(3.32)

Notice that if the order of summation of the product terms in the convolution summation is changed, then the order in which the \( h(k) \)'s appear in (3.32) changes. If the order is changed so that the \( h(k) \) with smallest magnitude is first, followed by the next smallest, etc., then the roundoff noise variance is minimized. However, performing the convolution summation in nonsequential order greatly complicates data indexing and so may not be worth the reduction obtained in roundoff noise.

### 3.5.2 Roundoff Noise in Fixed-Point IIR Filters

To determine the roundoff noise of a fixed-point infinite impulse response (IIR) filter realization, consider a causal first-order filter with impulse response

\[
h(n) = a^n u(n)
\]

(3.33)

realized by the difference equation

\[
y(n) = ay(n-1) + x(n)
\]

(3.34)

Due to roundoff error, the output actually obtained is

\[
y(n) = Q[ay(n-1) + x(n)] = ay(n-1) + x(n) + e(n)
\]

(3.35)
where $e(n)$ is a random roundoff noise sequence. Since $e(n)$ is injected at the same point as the input, it propagates through a system with impulse response $h(n)$. Therefore, for fixed-point arithmetic with rounding, the output roundoff noise variance from (3.6), (3.12), (3.25), and (3.33) is

$$
\sigma_o^2 = \frac{\Delta^2}{12} \sum_{n=0}^{\infty} h^2(n) = \frac{\Delta^2}{12} \sum_{n=0}^{\infty} a^{2n} = \frac{2^{-2B}}{12} \frac{1}{1 - a^2}
$$

(3.36)

With fixed-point arithmetic there is the possibility of overflow following addition. To avoid overflow it is necessary to restrict the input signal amplitude. This can be accomplished by either placing a scaling multiplier at the filter input or by simply limiting the maximum input signal amplitude. Consider the case of the first-order filter of (3.34). The transfer function of this filter is

$$
H(e^{j\omega}) = \frac{Y(e^{j\omega})}{X(e^{j\omega})} = \frac{1}{e^{j\omega} - a}
$$

(3.37)

so

$$
|H(e^{j\omega})|^2 = \frac{1}{1 + a^2 - 2a \cos(\omega)}
$$

(3.38)

and

$$
|H(e^{j\omega})|_{\text{max}} = \frac{1}{1 - |a|}
$$

(3.39)

The peak gain of the filter is $1/(1 - |a|)$ so limiting input signal amplitudes to $|x(n)| \leq 1 - |a|$ will make overflows unlikely.

An expression for the output roundoff noise-to-signal ratio can easily be obtained for the case where the filter input is white noise, uniformly distributed over the interval from $-(1 - |a|)$ to $(1 - |a|)$ [4, 5]. In this case

$$
\sigma_s^2 = \frac{1}{2(1 - |a|)} \int_{-(1 - |a|)}^{1 - |a|} x^2 \, dx = \frac{1}{3}(1 - |a|)^2
$$

(3.40)

so, from (3.25),

$$
\sigma_s^2 = \frac{1}{3} \frac{(1 - |a|)^2}{1 - a^2}
$$

(3.41)

Combining (3.36) and (3.41) then gives

$$
\frac{\sigma_o^2}{\sigma_s^2} = \left( \frac{2^{-2B}}{12} \frac{1}{1 - a^2} \right) \left( \frac{3}{1 - (|a|)^2} \right) = \frac{2^{-2B}}{12} \frac{3}{(1 - |a|)^2}
$$

(3.42)

Notice that the noise-to-signal ratio increases without bound as $|a| \to 1$.

Similar results can be obtained for the case of the causal second-order filter realized by the difference equation

$$
y(n) = 2r \cos(\theta)y(n - 1) - r^2y(n - 2) + x(n)
$$

(3.43)

This filter has complex-conjugate poles at $re^{\pm j\theta}$ and impulse response

$$
h(n) = \frac{1}{\sin(\theta)} r^n \sin((n + 1)\theta) u(n)
$$

(3.44)

Due to roundoff error, the output actually obtained is

$$
\hat{y}(n) = 2r \cos(\theta)y(n - 1) - r^2y(n - 2) + x(n) + e(n)
$$

(3.45)
There are two noise sources contributing to $e(n)$ if quantization is performed after each multiply, and there is one noise source if quantization is performed after summation. Since

$$
\sum_{n=-\infty}^{\infty} h^2(n) = \frac{1 + r^2}{1 - r^2} \frac{1}{(1 + r^2)^2 - 4r^2\cos^2(\theta)}
$$

(3.46)

the output roundoff noise is

$$
\sigma_{\delta}^2 = v \frac{2^{-2B}}{12} \frac{1 + r^2}{1 - r^2} \frac{1}{(1 + r^2)^2 - 4r^2\cos^2(\theta)}
$$

(3.47)

where $v = 1$ for quantization after summation, and $v = 2$ for quantization after each multiply.

To obtain an output noise-to-signal ratio we note that

$$
H(e^{j\omega}) = \frac{1}{1 - 2r\cos(\theta)e^{-j\omega} + r^2e^{-j2\omega}}
$$

(3.48)

and, using the approach of [6],

$$
|H(e^{j\omega})|^2_{\text{max}} = \frac{1}{4r^2} \left\{ \text{sat}\left(\frac{1+r^2}{2r}\cos(\theta)\right) - \frac{1+r^2}{2r}\cos(\theta) \right\}^2 + \left\{ \frac{1-r^2}{2r}\sin(\theta) \right\}^2
$$

(3.49)

where

$$
\text{sat}(\mu) = \begin{cases} 
1 & \mu > 1 \\
\mu & -1 \leq \mu \leq 1 \\
-1 & \mu < -1 
\end{cases}
$$

(3.50)

Following the same approach as for the first-order case then gives

$$
\frac{\sigma_{\delta}^2}{\sigma_{\gamma}^2} = v \frac{2^{-2B}}{12} \frac{1 + r^2}{1 - r^2} \frac{3}{(1 + r^2)^2 - 4r^2\cos^2(\theta)}
$$

$$
\times \frac{1}{4r^2} \left\{ \text{sat}\left(\frac{1+r^2}{2r}\cos(\theta)\right) - \frac{1+r^2}{2r}\cos(\theta) \right\}^2 + \left\{ \frac{1-r^2}{2r}\sin(\theta) \right\}^2
$$

(3.51)

Figure 3.1 is a contour plot showing the noise-to-signal ratio of (3.51) for $v = 1$ in units of the noise variance of a single quantization, $2^{-2B}/12$. The plot is symmetrical about $\theta = 90^\circ$, so only the range from $0^\circ$ to $90^\circ$ is shown. Notice that as $r \to 1$, the roundoff noise increases without bound. Also notice that the noise increases as $\theta \to 0^\circ$.

It is possible to design state-space filter realizations that minimize fixed-point roundoff noise [7]– [10]. Depending on the transfer function being realized, these structures may provide a roundoff noise level that is orders-of-magnitude lower than for a nonoptimal realization. The price paid for this reduction in roundoff noise is an increase in the number of computations required to implement the filter. For an $N$th-order filter the increase is from roughly $2N$ multiplies for a direct form realization to roughly $(N + 1)^2$ for an optimal realization. However, if the filter is realized by the parallel or cascade connection of first- and second-order optimal subfilters, the increase is only to about $4N$ multiplies. Furthermore, near-optimal realizations exist that increase the number of multiplies to only about $3N$ [10].

©1999 by CRC Press LLC
3.5.3 Roundoff Noise in Floating-Point IIR Filters

For floating-point arithmetic it is first necessary to determine the injected noise variance of each quantization. For the first-order filter this is done by writing the computed output as

\[ y(n) + e(n) = (ay(n-1)(1 + \varepsilon_1(n)) + x(n))(1 + \varepsilon_2(n)) \]  (3.52)

where \( \varepsilon_1(n) \) represents the error due to the multiplication and \( \varepsilon_2(n) \) represents the error due to the addition. Neglecting the product of errors, (3.52) becomes

\[ y(n) + e(n) \approx ay(n-1) + x(n) + ay(n-1)\varepsilon_1(n) + ay(n-1)\varepsilon_2(n) + x(n)\varepsilon_2(n) \]  (3.53)

Comparing (3.34) and (3.53), it is clear that

\[ e(n) = ay(n-1)\varepsilon_1(n) + ay(n-1)\varepsilon_2(n) + x(n)\varepsilon_2(n) \]  (3.54)

Taking the expected value of \( e^2(n) \) to obtain the injected noise variance then gives

\[ E\{e^2(n)\} = a^2E\{y^2(n-1)\}E\{\varepsilon_1^2(n)\} + a^2E\{y^2(n-1)\}E\{\varepsilon_2^2(n)\} + E\{x^2(n)\}E\{\varepsilon_2^2(n)\} + E\{x(n)\}E\{y(n-1)\}E\{\varepsilon_2^2(n)\} \]  (3.55)

To carry this further it is necessary to know something about the input. If we assume the input is zero-mean white noise with variance \( \sigma_x^2 \), then \( E\{x^2(n)\} = \sigma_x^2 \) and the input is uncorrelated with past values of the output so \( E\{x(n)\}E\{y(n-1)\} = 0 \) giving

\[ E\{e^2(n)\} = 2a^2\sigma_x^2\sigma_{\varepsilon_1}^2 + a^2\sigma_x^2\sigma_{\varepsilon_2}^2 \]  (3.56)
and

\[
\sigma_o^2 = \left(2a^2\sigma_\varepsilon^2 + \sum_{n=-\infty}^{\infty} h^2(n)\right)
\]

\[
= \frac{2a^2\sigma_\varepsilon^2 + \sigma_y^2}{1-a^2}
\]

(3.57)

However,

\[
\sigma_y^2 = \sigma_x^2 \sum_{n=-\infty}^{\infty} h^2(n) = \frac{\sigma_x^2}{1-a^2}
\]

(3.58)

so

\[
\sigma_o^2 = \frac{1+a^2}{1-a^2}\sigma_y^2 = \frac{1+a^2}{1-a^2}\sigma_e^2
\]

(3.59)

and the output roundoff noise-to-signal ratio is

\[
\frac{\sigma_o^2}{\sigma_y^2} = \frac{1+a^2}{1-a^2}
\]

(3.60)

Similar results can be obtained for the second-order filter of (3.43) by writing

\[
y(n) + e(n) = ([2r\cos(\theta)y(n-1)(1+\varepsilon_1(n)) - r^2y(n-2)(1+\varepsilon_2(n))] \\
\times [1+\varepsilon_3(n)] + x(n)(1+\varepsilon_4(n))
\]

(3.61)

Expanding with the same assumptions as before gives

\[
e(n) \approx 2r\cos(\theta)y(n-1)[\varepsilon_1(n) + \varepsilon_3(n) + \varepsilon_4(n)] \\
- r^2y(n-2)[\varepsilon_2(n) + \varepsilon_3(n) + \varepsilon_4(n)] + x(n)\varepsilon_4(n)
\]

(3.62)

and

\[
E[e^2(n)] = 4r^2\cos^2(\theta)\sigma_\varepsilon^2\sigma_y^2 + r^2\sigma_\varepsilon^2\sigma_y^2 \\
+ \sigma_\varepsilon^2\sigma_y^2 - 8r^3\cos(\theta)\sigma_\varepsilon^2 E[y(n-1)y(n-2)]
\]

(3.63)

However,

\[
E[y(n-1)y(n-2)] \\
= E[[2r\cos(\theta)y(n-2) - r^2y(n-3) + x(n-1)y(n-2)]y(n-2)] \\
= 2r\cos(\theta)E[y^2(n-2)] - r^2E[y(n-2)y(n-3)] \\
= 2r\cos(\theta)E[y^2(n-2)] - r^2E[y(n-1)y(n-2)] \\
= \frac{2r\cos(\theta)}{1+r^2}\sigma_y^2
\]

(3.64)

so

\[
E[e^2(n)] = \sigma_\varepsilon^2\sigma_y^2 + \left[3r^4 + 12r^2\cos^2(\theta) - \frac{16r^4\cos^2(\theta)}{1+r^2}\right]\sigma_\varepsilon^2\sigma_y^2
\]

(3.65)

and

\[
\sigma_o^2 = E[e^2(n)] \sum_{n=-\infty}^{\infty} h^2(n) \\
= \xi \left[\sigma_\varepsilon^2\sigma_y^2 + \left[3r^4 + 12r^2\cos^2(\theta) - \frac{16r^4\cos^2(\theta)}{1+r^2}\right]\sigma_\varepsilon^2\sigma_y^2\right]
\]

(3.66)
where from (3.46),

$$\xi = \sum_{n=\infty}^{\infty} h^2(n) = \frac{1 + r^2}{1 - r^2} \frac{1}{(1 + r^2)^2 - 4r^2 \cos^2(\theta)}$$  \hspace{1cm} (3.67)

Since $\sigma^2_Y = \xi \sigma^2_x$, the output roundoff noise-to-signal ratio is then

$$\frac{\sigma^2_Y}{\sigma^2_x} = \xi \left[ 1 + \xi \left[ 3r^4 + 12r^2 \cos^2(\theta) - \frac{16r^4 \cos^2(\theta)}{1 + r^2} \right] \right] \sigma^2_{\epsilon_x}$$ \hspace{1cm} (3.68)

Figure 3.2 is a contour plot showing the noise-to-signal ratio of (3.68) in units of the noise variance of a single quantization $\sigma^2_{\epsilon_x}$. The plot is symmetrical about $\theta = 90^\circ$, so only the range from $0^\circ$ to $90^\circ$ is shown. Notice the similarity of this plot to that of Fig. 3.1 for the fixed-point case. It has been observed that filter structures generally have very similar fixed-point and floating-point roundoff characteristics [2]. Therefore, the techniques of [7] - [10], which were developed for the fixed-point case, can also be used to design low-noise floating-point filter realizations. Furthermore, since it is not necessary to scale the floating-point realization, the low-noise realizations need not require significantly more computation than the direct form realization.

FIGURE 3.2: Normalized floating-point roundoff noise variance.

### 3.6 Limit Cycles

A limit cycle, sometimes referred to as a multiplier roundoff limit cycle, is a low-level oscillation that can exist in an otherwise stable filter as a result of the nonlinearity associated with rounding (or truncating) internal filter calculations [11]. Limit cycles require recursion to exist and do not occur in nonrecursive FIR filters.

©1999 by CRC Press LLC
As an example of a limit cycle, consider the second-order filter realized by

\[ y(n) = Q_r \left\{ \frac{7}{8} y(n-1) - \frac{5}{8} y(n-2) + x(n) \right\} \]  

(3.69)

where \( Q_r \) represents quantization by rounding. This is stable filter with poles at \( 0.4375 \pm j0.6585 \). Consider the implementation of this filter with 4-b (3-b and a sign bit) two's complement fixed-point arithmetic, zero initial conditions \( y(-1) = y(-2) = 0 \), and an input sequence \( x(n) = \frac{3}{8} \delta(n) \), where \( \delta(n) \) is the unit impulse or unit sample. The following sequence is obtained:

\[
\begin{align*}
    y(0) &= Q_r \left\{ \frac{3}{8} \right\} = \frac{3}{8} \\
    y(1) &= Q_r \left\{ \frac{21}{64} \right\} = \frac{3}{8} \\
    y(2) &= Q_r \left\{ \frac{3}{32} \right\} = \frac{1}{8} \\
    y(3) &= Q_r \left\{ -\frac{1}{8} \right\} = -\frac{1}{8} \\
    y(4) &= Q_r \left\{ -\frac{3}{16} \right\} = -\frac{1}{8} \\
    y(5) &= Q_r \left\{ -\frac{1}{32} \right\} = 0 \\
    y(6) &= Q_r \left\{ \frac{5}{64} \right\} = \frac{1}{8} \\
    y(7) &= Q_r \left\{ \frac{7}{64} \right\} = \frac{1}{8} \\
    y(8) &= Q_r \left\{ \frac{1}{32} \right\} = 0 \\
    y(9) &= Q_r \left\{ -\frac{5}{64} \right\} = -\frac{1}{8} \\
    y(10) &= Q_r \left\{ -\frac{7}{64} \right\} = -\frac{1}{8} \\
    y(11) &= Q_r \left\{ -\frac{1}{32} \right\} = 0 \\
    y(12) &= Q_r \left\{ \frac{5}{64} \right\} = \frac{1}{8} \\
    \vdots
\end{align*}
\]

(3.70)

Notice that while the input is zero except for the first sample, the output oscillates with amplitude 1/8 and period 6.

Limit cycles are primarily of concern in fixed-point recursive filters. As long as floating-point filters are realized as the parallel or cascade connection of first- and second-order subfilters, limit cycles will generally not be a problem since limit cycles are practically not observable in first- and second-order systems implemented with 32-b floating-point arithmetic [12]. It has been shown that such systems must have an extremely small margin of stability for limit cycles to exist at anything other than underflow levels, which are at an amplitude of less than \( 10^{-38} \) [12].

©1999 by CRC Press LLC
There are at least three ways of dealing with limit cycles when fixed-point arithmetic is used. One is to determine a bound on the maximum limit cycle amplitude, expressed as an integral number of quantization steps [13]. It is then possible to choose a word length that makes the limit cycle amplitude acceptably low. Alternately, limit cycles can be prevented by randomly rounding calculations up or down [14]. However, this approach is complicated to implement. The third approach is to properly choose the filter realization structure and then quantize the filter calculations using magnitude truncation [15, 16]. This approach has the disadvantage of producing more roundoff noise than truncation or rounding [see (3.12)–(3.14)].

3.7 Overflow Oscillations

With fixed-point arithmetic it is possible for filter calculations to overflow. This happens when two numbers of the same sign add to give a value having magnitude greater than one. Since numbers with magnitude greater than one are not representable, the result overflows. For example, the two's complement numbers 0.101 (5/8) and 0.100 (4/8) add to give 1.001 which is the two's complement representation of $-7/8$.

The overflow characteristic of two's complement arithmetic can be represented as $R()$ where

$$R(X) = \begin{cases} X - 2 & X \geq 1 \\ X & -1 \leq X < 1 \\ X + 2 & X < -1 \end{cases} \quad (3.71)$$

For the example just considered, $R(9/8) = -7/8$.

An overflow oscillation, sometimes also referred to as an adder overflow limit cycle, is a high-level oscillation that can exist in an otherwise stable fixed-point filter due to the gross nonlinearity associated with the overflow of internal filter calculations [17]. Like limit cycles, overflow oscillations require recursion to exist and do not occur in nonrecursive FIR filters. Overflow oscillations also do not occur with floating-point arithmetic due to the virtual impossibility of overflow.

As an example of an overflow oscillation, once again consider the filter of (3.69) with 4-b fixed-point two's complement arithmetic and with the two's complement overflow characteristic of (3.71):

$$y(n) = Q_r \left\{ R \left[ \frac{7}{8} y(n-1) - \frac{5}{8} y(n-2) + x(n) \right] \right\} \quad (3.72)$$

In this case we apply the input

$$x(n) = \begin{cases} -\frac{3}{4} \delta(n) - \frac{5}{8} \delta(n-1) \\ \{-\frac{3}{4}, -\frac{5}{8}, 0, 0, \ldots\} \end{cases} \quad (3.73)$$

giving the output sequence

$$\begin{align*}
y(0) &= Q_r \left\{ R \left[ -\frac{3}{4} \right] \right\} = Q_r \left\{ -\frac{3}{4} \right\} = -\frac{3}{4} \\
y(1) &= Q_r \left\{ R \left[ -\frac{41}{32} \right] \right\} = Q_r \left\{ -\frac{23}{32} \right\} = \frac{3}{4} \\
y(2) &= Q_r \left\{ R \left[ \frac{9}{8} \right] \right\} = Q_r \left\{ -\frac{7}{8} \right\} = -\frac{7}{8} \\
y(3) &= Q_r \left\{ R \left[ -\frac{79}{64} \right] \right\} = Q_r \left\{ -\frac{49}{64} \right\} = \frac{3}{4} 
\end{align*}$$

©1999 by CRC Press LLC
This is a large-scale oscillation with nearly full-scale amplitude.

There are several ways to prevent overflow oscillations in fixed-point filter realizations. The most obvious is to scale the filter calculations so as to render overflow impossible. However, this may unacceptably restrict the filter dynamic range. Another method is to force completed sums-of-products to saturate at \( \pm 1 \), rather than overflowing [18, 19]. It is important to saturate only the completed sum, since intermediate overflows in two's complement arithmetic do not affect the accuracy of the final result. Most fixed-point digital signal processors provide for automatic saturation of completed sums if their saturation arithmetic feature is enabled. Yet another way to avoid overflow oscillations is to use a filter structure for which any internal filter transient is guaranteed to decay to zero [20]. Such structures are desirable anyway, since they tend to have low roundoff noise and be insensitive to coefficient quantization [21].

### 3.8 Coefficient Quantization Error

Each filter structure has its own finite, generally nonuniform grids of realizable pole and zero locations when the filter coefficients are quantized to a finite word length. In general the pole and zero locations desired in filter do not correspond exactly to the realizable locations. The error in filter performance (usually measured in terms of a frequency response error) resulting from the placement of the poles and zeroes at the nonideal but realizable locations is referred to as coefficient quantization error.

Consider the second-order filter with complex-conjugate poles

\[
\lambda = r e^{\pm j \theta} = \lambda_r \pm j \lambda_i = r \cos(\theta) \pm j r \sin(\theta)
\]

and transfer function

\[
H(z) = \frac{1}{1 - 2r \cos(\theta)z^{-1} + r^2 z^{-2}}
\]

realized by the difference equation

\[
y(n) = 2r \cos(\theta)y(n - 1) - r^2 y(n - 2) + x(n)
\]

Figure 3.3 from [5] shows that quantizing the difference equation coefficients results in a nonuniform grid of realizable pole locations in the \( z \) plane. The grid is defined by the intersection of vertical lines corresponding to quantization of \( 2 \lambda_r \) and concentric circles corresponding to quantization of \( -r^2 \).
The sparseness of realizable pole locations near $z = \pm 1$ will result in a large coefficient quantization error for poles in this region.

Figure 3.4 gives an alternative structure to (3.77) for realizing the transfer function of (3.76). Notice that quantizing the coefficients of this structure corresponds to quantizing $\lambda_r$ and $\lambda_i$. As shown in Fig. 3.5 from [5], this results in a uniform grid of realizable pole locations. Therefore, large coefficient quantization errors are avoided for all pole locations.

It is well established that filter structures with low roundoff noise tend to be robust to coefficient quantization, and visa versa [22]–[24]. For this reason, the uniform grid structure of Fig. 3.4 is also popular because of its low roundoff noise. Likewise, the low-noise realizations of [7]–[10] can be expected to be relatively insensitive to coefficient quantization, and digital wave filters and lattice filters that are derived from low-sensitivity analog structures tend to have not only low coefficient sensitivity, but also low roundoff noise [25, 26].

It is well known that in a high-order polynomial with clustered roots, the root location is a very sensitive function of the polynomial coefficients. Therefore, filter poles and zeros can be much more accurately controlled if higher order filters are realized by breaking them up into the parallel or cascade connection of first- and second-order subfilters. One exception to this rule is the case of linear-phase FIR filters in which the symmetry of the polynomial coefficients and the spacing of the filter zeros around the unit circle usually permits an acceptable direct realization using the convolution summation.

Given a filter structure it is necessary to assign the ideal pole and zero locations to the realizable locations. This is generally done by simply rounding or truncating the filter coefficients to the available number of bits, or by assigning the ideal pole and zero locations to the nearest realizable locations. A more complicated alternative is to consider the original filter design problem as a problem in discrete
FIGURE 3.4: Alternate realization structure.

FIGURE 3.5: Realizable pole locations for the alternate realization structure.
optimization, and choose the realizable pole and zero locations that give the best approximation to
the desired filter response [27]- [30].

3.9 Realization Considerations

Linear-phase FIR digital filters can generally be implemented with acceptable coefficient quantization
sensitivity using the direct convolution sum method. When implemented in this way on a digital
signal processor, fixed-point arithmetic is not only acceptable but may actually be preferable to
floating-point arithmetic. Virtually all fixed-point digital signal processors accumulate a sum of
products in a double-length accumulator. This means that only a single quantization is necessary to
compute an output. Floating-point arithmetic, on the other hand, requires a quantization after every
multiply and after every add in the convolution summation. With 32-b floating-point arithmetic
these quantizations introduce a small enough error to be insignificant for many applications.

When realizing IIR filters, either a parallel or cascade connection of first- and second-order sub-
filters is almost always preferable to a high-order direct-form realization. With the availability of
very low-cost floating-point digital signal processors, like the Texas Instruments TM 5320C32, it is
highly recommended that floating-point arithmetic be used for IIR filters. Floating-point arithmetic
simultaneously eliminates most concerns regarding scaling, limit cycles, and overflow oscillations.
Regardless of the arithmetic employed, a low roundoff noise structure should be used for the second-
order sections. Good choices are given in [2] and [10]. Recall that realizations with low fixed-point
roundoff noise also have low floating-point roundoff noise. The use of a low roundoff noise structure
for the second-order sections also tends to give a realization with low coefficient quantization
sensitivity. First-order sections are not as critical in determining the roundoff noise and coefficient
sensitivity of a realization, and so can generally be implemented with a simple direct form structure.

References

[1] Weinstein, C. and Oppenheim, A.V., A comparison of roundoff noise in floating-point and
of second-order state-space digital filter structures, IEEE Trans. Circuits Syst. II, 39, 90-98,
[3] Proakis, G.J. and Manolakis, D.J., Introduction to Digital Signal Processing, New York, Macmil-
Hall, 1975.
[6] Bomar, B.W. and Joseph, R.D., Calculation of \( L_\infty \) norms for scaling second-order state-space
[9] Barnes, C.W., On the design of optimal state-space realizations of second-order digital filters,
[10] Bomar, B.W., New second-order state-space structures for realizing low roundoff noise digital