UNIT-V

EFFECT OF FINITE REGISTER LENGTH

5.1 FINITE WORD-LENGTH EFFECTS

In implementing a discrete-time system in hardware or software, it is important to consider the finite word-lengt effects. For example, if a filter is to be implemented on a fixed-point processor, the filter coefficients must b quantized to a finite number of bits. This will change the frequency response characteristics of the filter. In thi section, we look at the finite precision effects in digital filter implementations.

5.2 *Binary Representation of Numbers*

There are two basic systems for representing numbers in a digital system: fixed point and floating point. The is a trade-off in which type of representation to use. The dynamic range that is available in a floating-poi representation is much larger than with fixed-point numbers. However, fixed-point processors are typically fast and less expensive. Below, we briefly describe these number representations.

5.3 Fixed Point

In the binary representation of a real number, x, using $B+1$ bits, there are three commonly used formats: sig magnitude, one's complement, and two's complement, with two's complement being the most common. In thes systems, the only difference is in the way that negative numbers are represented.

1. *Sign magnitude*: With a sign-magnitude format, a number x is represented as

Quantization Errors in Fixed-Point Number Systems

In performing computations within a fixed- or floating-point digital processor, it is necessary to quantize numbers by either truncation or rounding from some level of precision to a lower level. For example, because multiplying two 16-bit fixed-point numbers will produce a product with up to 31 bits of precision, the product will generally need to be quantized back to 16 bits. Truncation and rounding introduce a quantization error

$$e = Q[x] - x$$

where x is the number to be quantized and $Q[x]$ is the quantized number. The characteristics of the error depend upon the number representation that is used. Truncating numbers that are represented in sign-magnitude form result in a quantization error that is negative for positive numbers and positive for negative numbers. Thus, the quantization error is symmetric about zero and falls in the range

$$-\Delta \le e \le \Delta$$

where $$\Delta = X_m 2^{-B}$$

On the other hand, for a two's complement representation, the truncation error is always negative and falls in the range

$$-\Delta \le e \le 0$$

With rounding, the quantization error is independent of the type of fixed-point representation and falls in th range

$$-\frac{\Delta}{2} \leq e \leq \frac{\Delta}{2}$$

For floating-point numbers, the mantissa is either rounded or truncated, and the size of the error depends on th value of the exponent.

5.4 *Quantization of Filter Coefficients*

In order to implement a filter on a digital processor, the filter coefficients must be converted into binary form. This conversion leads to movements in the pole and zero locations and a change in the frequency response of the filter. The accuracy with which the filter coefficients can be specified depends upon the word length of the processor, and the sensitivity of the filter to coefficient quantization depends on the structure of the filter, as well as on the locations of the poles and zeros.

For an FIR filter,

$$H(z) = \sum_{n=0}^{N} h(n) z^{-n}$$

5.5 QUANTIZATION :

Quantization, in mathematics and digital signal processing, is the process of mapping a large set of input values to a (countable) smaller set. Rounding and truncation are typical examples of quantization processes. Quantization is involved to some degree in nearly all digital signal processing, as the process of representing a signal in digital form ordinarily involves rounding. Quantization also forms the core of essentially all lossy compression algorithms. The difference between an input value and its quantized value (such as round-off error) is referred to as **quantization error**. A device or algorithmic function that performs quantization is called a **quantizer**. An analog-to-digital converter is an example of a quantizer.

It refers to the process of approximating the continuous set of values in the image data with a finite (preferably small) set of values. The input to a quantizer is the original data, and the output is always one among a finite number of levels. The quantizer is a function whose set of output values are discrete, and usually finite. Obviously, this is a process of approximation, and a good quantizer is one which represents the original signal with minimum loss or distortion. The difference between the actual analog value and quantized digital value due is called quantization error. This error is due either to rounding or truncation.

Quantization noise is a model of quantization error introduced by quantization in the analog-to-digital conversion (ADC) in telecommunication systems and signal processing. It is a rounding error between the analog input voltage to the ADC and the output digitized value. The noise is non-linear and signal-dependent. It can be modelled in several different ways. In an ideal analog-to-digital converter, where the quantization error is uniformly distributed between −1/2

LSB and +1/2 LSB, and the signal has a uniform distribution covering all quantization levels, the signal-to-noise ratio (SNR) can be calculated from The most common test signals that fulfil this are full amplitude triangle waves and sawtooth waves. In this case a 16-bit ADC has a maximum signal-to-noise ratio of 6.0206 × 16 = 96.33 dB. When the input signal is a full-amplitude sine wave the distribution of the signal is no longer uniform, and the corresponding equation is instead Here, the quantization noise is once again assumed to be uniformly distributed. When the input signal has a high amplitude and a wide frequency spectrum this is the case. In this case a 16-bit ADC has a maximum signal-to-noise ratio of 98.09 dB. The 1.761 difference in signal-to-noise only occurs due to the signal being a full-scale sine wave instead of a triangle/sawtooth.

5.6 Basic properties of quantization

Because quantization is a many-to-few mapping, it is an inherently non-linear and irreversible process (i.e., because the same output value is shared by multiple input values, it is impossible in general to recover the exact input value when given only the output value).

The set of possible input values may be infinitely large, and may possibly be continuous and therefore uncountable (such as the set of all real numbers, or all real numbers within some limited range). The set of possible output values may be finite or countably infinite. The input and output sets involved in quantization can be defined in a rather general way. For example, vector quantization is the application of quantization to multi-dimensional (vector-valued) input data

Analog-to-digital converter (ADC)

Outside the realm of signal processing, this category may simply be called rounding or scalar quantization. An ADC can be modeled as two processes: sampling and **quantization**. Sampling converts a voltage signal (function of time) into a discrete-time signal (sequence of real numbers). Quantization replaces each real number with an approximation from a finite set of discrete values (**levels**), which is necessary for storage and processing by numerical methods. Most commonly, these discrete values are represented as fixed-point words (either proportional to the waveform values or companded) or floating-point words. Common word-lengths are 8-bit (256 levels), 16-bit (65,536 levels), 32-bit (4.3 billion levels), and so on, though any number of quantization levels is possible (not just powers of two). Quantizing a sequence of numbers produces a sequence of quantization errors which is sometimes modeled as an additive random signal called **quantization noise** because of its stochastic behavior. The more levels a quantizer uses, the lower is its quantization noise power.

In general, both ADC processes lose some information. So discrete-valued signals are only an approximation of the continuous-valued discrete-time signal, which is itself only an approximation of the original continuous-valued continuous-time signal. But both types of approximation errors can, in theory, be made arbitrarily small by good design.

5.7 FIXED POINT AND FLOATING POINT NUMBER REPRESENTATIONS:

Fixed Point Representation: In computing, a fixed-point number representation is a real data type for a number that has a fixed number of digits after (and sometimes also before) the radix point (e.g., after the decimal point '.' in English decimal notation). Fixed-point number representation can be compared to the more complicated (and more computationally demanding) floating point number representation. Fixed-point numbers are useful for representing fractional values, usually in base 2 or base 10, when the executing processor has no floating point unit (FPU) or if fixed-point provides improved performance or accuracy for the application at hand. Most low-cost embedded microprocessors and microcontrollers do not have an FPU. The two most common fixed-point types are decimal and binary. Decimal fixed-point types have a scaling factor that is a power of ten, for binary fixed-point types it is a power of two. Binary fixed-point types are most commonly used, because the rescaling operations can be implemented as fast bit shifts. Binary fixed-point numbers can represent fractional powers of two exactly, but, like binary floating-point numbers, cannot exactly represent fractional powers of ten. If exact fractional powers of ten are desired, then a decimal format should be used. For example, one-tenth (0.1) and one-hundredth (0.01) can be represented only approximately by binary fixed-point or binary floating-point representations, while they can be represented exactly in decimal fixed-point or decimal floating-point representations.

5.8 Floating Point Representation:

In computing, floating point describes a system for numerical representation in which a string of digits (or bits) represents a rational number. The term floating point refers to the fact that the radix point (decimal point, or, more commonly in computers, binary point) can "float"; that is, it can be placed anywhere relative to the significant digits of the number. This position is indicated separately in the internal representation, and floating-point representation can thus be thought of as a computer realization of scientific notation. The advantage of floating-point representation over fixed-point (and integer) representation is that it can support a much wider range of values. For example, a fixed-point representation that has seven decimal digits, with the decimal point assumed to be positioned after the fifth digit, can represent the numbers 12345.67, 8765.43, 123.00, and so on, whereas a floating-point representation (such as the IEEE 754 decimal32 format) with seven decimal digits could in addition represent 1.234567, 123456.7, 0.00001234567, 1234567000000000, and so on. The floating-point format needs slightly more storage (to encode the position of the radix point), so when stored in the same space, floating-point numbers achieve their greater range at the expense of slightly less precision.

5.9 QUANTIZATION NOISE POWER:

Quantization refers to the process of approximating the continuous set of values in the image data with a finite (preferably small) set of values. The input to a quantizer is the original data, and the output is always one among a finite number of levels. The quantizer is a function whose set of output values are discrete, and usually finite. Obviously, this is a process of approximation, and a good quantizer is one which represents the original signal with minimum loss or distortion.

Finite register lengths and A/D converters cause errors in:- (i) Input quantisation. (ii) Coefficient (or multiplier) quantisation (iii) Products of multiplication truncated or rounded due to machine length

Truncation: simply chop off the remaining digits; also called rounding to zero.

Round to nearest: round to the nearest value, with ties broken in one of two ways. The result may **round up** or **round down**.

5.9 TRUNCATION AND ROUNDING: ROUNDING ERROR:

A round-off error, also called rounding error, is the difference between the calculated approximation of a number and its exact mathematical value. Numerical analysis specifically tries to estimate this error when using approximation equations and/or algorithms, especially when using finite digits to represent real numbers (which in theory have infinite digits). This is a form of quantization error. Truncation: simply chop off the remaining digits; also called rounding to zero. 0.142857 ≈ 0.142 (dropping all significant digits after 3rd) Round to nearest: round to the nearest value, with ties broken in one of two ways. The result may round up or round down. TRUNCATION: In mathematics, truncation is the term for limiting the number of digits right of the decimal point, by discarding the least significant ones. For example, consider the real numbers 5.6341432543653654 32.438191288 -6.3444444444444 To truncate these numbers to 4 decimal digits, we only consider the 4 digits to the right of the decimal point. The result would be: 5.6341 32.4381 -6.3444 Note that in some cases, truncating would yield the same result as rounding, but truncation does not round up or round down the digits; it merely cuts off at the specified digit. The truncation error can be twice the maximum error in rounding.

5.10 ROUND OFF:

A **round-off error**, also called **rounding error**, is the difference between the calculated approximation of a number and its exact mathematical value. Numerical analysis specifically tries to estimate this error when using approximation equations and/or algorithms, especially when using finite digits to represent real numbers (which in theory have infinite digits). This is a form of quantization error.

Rounding example:

As an example, rounding a real number x to the nearest integer value forms a very basic type of quantizer – a uniform one. A typical (mid-tread) uniform quantizer with a quantization step size equal to some value Δ can be expressed as

$$Q(x) = \Delta \cdot \left\lfloor \frac{x}{\Delta} + \frac{1}{2} \right\rfloor = \Delta \cdot \operatorname{floor}\left(\frac{x}{\Delta} + \frac{1}{2} \right),$$

where the notation $\lfloor \ \rfloor$ or $\operatorname{floor}(\ )$ depict the floor function. For simple rounding to the nearest integer, the step size Δ is equal to 1. With $\Delta = 1$ or with Δ equal to any other integer value, this quantizer has real-valued inputs and integer-valued outputs, although this property is not a necessity – a quantizer may also have an integer input domain and may also have non-integer output values. The essential property of a quantizer is that it has a countable set of possible output values that has fewer members than the set of possible input values. The members of the set of output values may have integer, rational, or real values (or even other possible values as well, in general – such as vector values or complex numbers).

When the quantization step size is small (relative to the variation in the signal being measured), it is relatively simple to show[3][4][5][6][7][8] that the mean squared error produced by such a rounding operation will be approximately $\Delta^2/12$. Mean squared error is also called the quantization **noise power**. Adding one bit to the quantizer halves the value of Δ, which reduces the noise power by the factor ¼. In terms of decibels, the noise power change is $10 \cdot \log_{10}\left(\frac{1}{4}\right) = -6 \text{ dB}.$

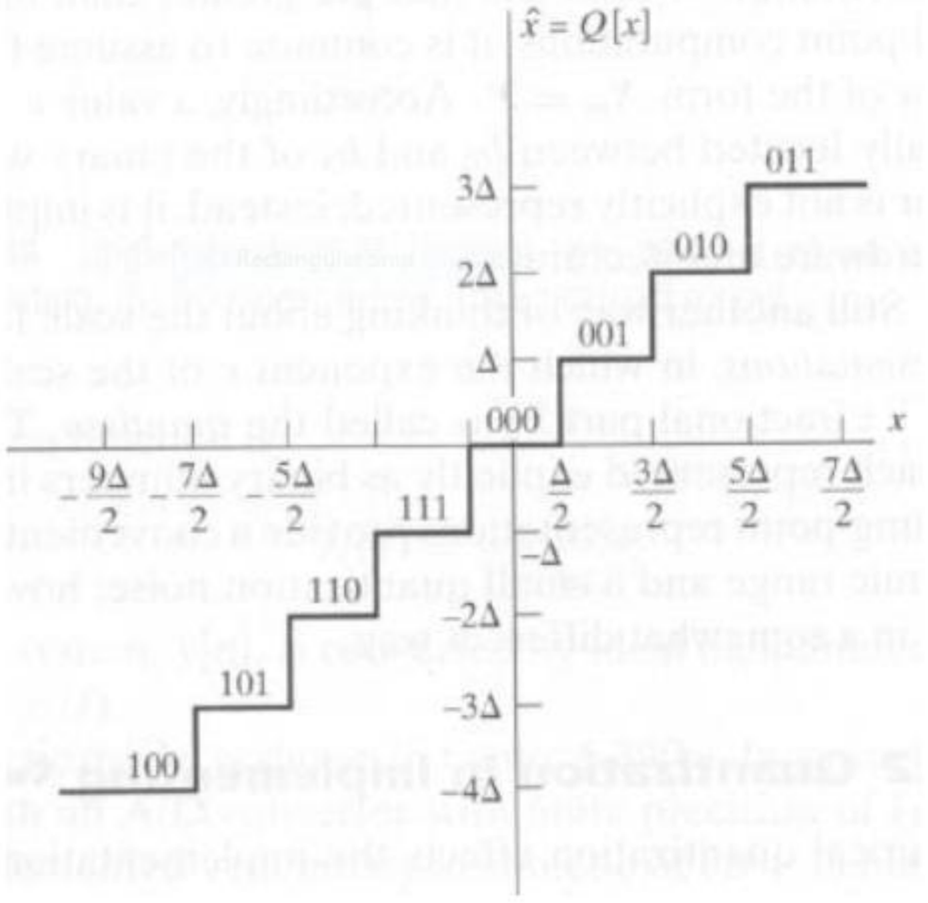
Because the set of possible output values of a quantizer is countable, any quantizer can be decomposed into two distinct stages, which can be referred to as the classification stage (or forward quantization stage) and the reconstruction stage (or inverse quantization stage), where the classification stage maps the input value to an integer quantization index k and the reconstruction stage maps the index k to the reconstruction value y_k that is the output approximation of the input value. For the example uniform quantizer described above, the forward quantization stage can be expressed as

$$k = \left\lfloor \frac{x}{\Delta} + \frac{1}{2} \right\rfloor,$$

and the reconstruction stage for this example quantizer is simply

$$y_k = k \cdot \Delta.$$

This decomposition is useful for the design and analysis of quantization behavior, and it illustrates how the quantized data can be communicated over a communication channel – a source encoder can perform the forward quantization stage and send the index information through a communication channel (possibly applying entropy coding techniques to the quantization indices), and a decoder can perform the reconstruction stage to produce the output approximation of the original input data. In more elaborate quantization designs, both the forward and inverse quantization stages may be substantially more complex. In general, the forward quantization stage may use any function that maps the input data to the integer space of the quantization index data, and the inverse quantization stage can conceptually (or literally) be a table look-up operation to map each quantization index to a corresponding reconstruction value. This two-stage decomposition applies equally well to vector as well as scalar quantizers.



rounding

5.11 TRUNCATION:

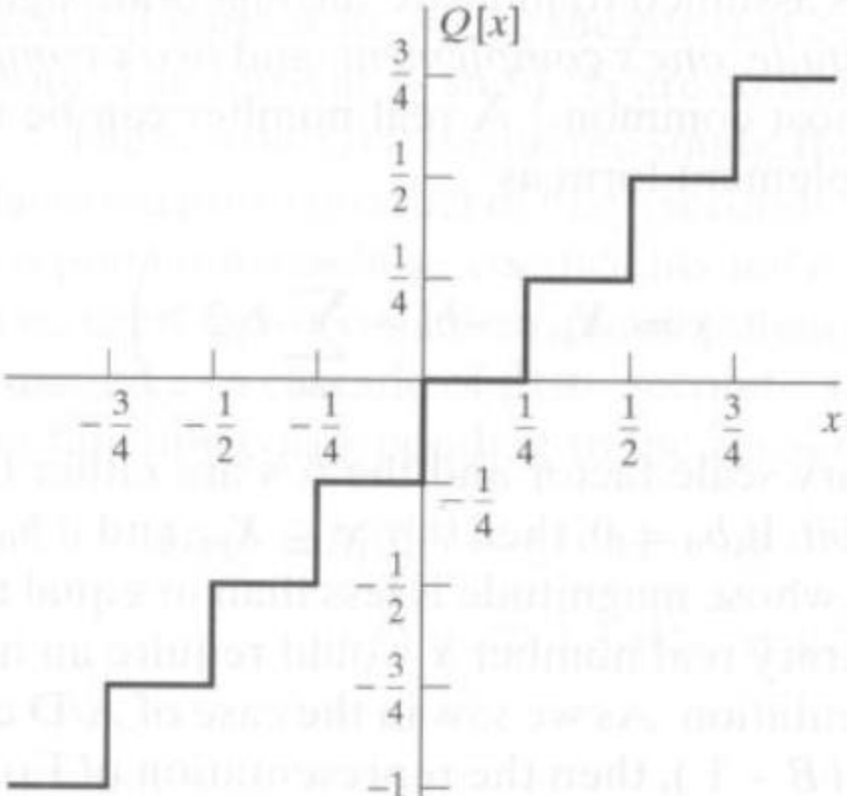
In mathematics, **truncation** is the term for limiting the number of digits right of the decimal point, by discarding the least significant ones.

For example, consider the real numbers

5.6341432543653654

32.438191288

-6.3444444444444
To truncate these numbers to 4 decimal digits, we only consider the 4 digits to the right of the decimal point.
The result would be
5.6341
32.4381
-6.3444
Note that in some cases, truncating would yield the same result as rounding, but truncation does not round up or round down the digits; it merely cuts off at the specified digit. The truncation error can be twice the maximum error in rounding.



(b)

truncation

.

5.12 LIMIT CYCLE OSCILLATIONS:

For an IIR filter implemented with infinite precision arithmetic the output should approach zero in the steady state if the input is zero and it should approach a constant value if the input is a constant. However , with an implementation using a finite length register an output can occur even with zero input. The output may be a fixed value or it may oscillate between finite positive and negative values. This effect is referred to as (zero input) limit cycle oscillation.
A limit cycle, sometimes referred to as a multiplier round off limit cycle, is a low-level oscillations that can exist in an otherwise stable filter as a result of the nonlinearity associated with rounding (or truncating) internal filter calculations . Limit cycles require recursion to exist and do not occur in non-recursive FIR filters. 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. 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. 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 approach. This approach has the disadvantage of producing more round off noise than truncation or rounding.

5.13 LIMIT CYCLE OVERFLOW:

The addition of two fixed point arithmetic numbers cause overflow when the sum exceeds the word ze available to store the sum. This overflow caused by adder make the filter output to oscillate between maximum amplitude limits. Such limit cycles have been referred to as overflow oscillations. The limit cycles occur as a result of quantization effect in multiplication. The amplitudes of the output during a limit cycle are confined to a range of values called the dead band of the filter.

5.14 GIBB'S OSCILLATION:

The truncation of Fourier series is known to introduce the unwanted ripples in the frequency response characteristics H(w) due to non uniform convergence of Fourier series at a discontinuity .These ripples or oscillatory behaviour near the band edge of the filter is known as "Gibb's phenomenon or Gibb's oscillation ".

METHODS TO REDUCE GIBB'S OSCILLATION:

There are two methods to reduce Gibb's phenomenon

1. The discontinuity between pass band and stop band in the frequency response is avoided by introducing the transition between the pass band and stop band.

2. Another technique used for the reduction of Gibb's phenomenon is by using window function that contains a taper which decays towards zero gradually instead abruptly.

5.15 SCALING:

Saturation arithmetic eliminates limit cycle due to overflow, but it causes undesirable signal distortion due to the non-linearity of the clipper. In order to limit the amount of non-linear distortion, it is important to scale the input signal and the unit sample response between the input and any internal summing node in the system such that overflows becomes a rare event.

5.16 DYNAMIC RANGE:

The dynamic range of a signal processing system can be defined as the maximum dB level sustainable without overflow (or other distortion) minus the dB level of the ``noise floor''.

Similarly, the dynamic range of a signal can be defined as its maximum decibel level minus its average ``noise level'' in dB. For digital signals, the limiting noise is ideally quantization noise.

Quantization noise is generally modeled as a uniform random variable between plus and minus half the least significant bit (since rounding to the nearest representable sample value is normally used). If q denotes the quantization interval, then the maximum quantization-error magnitude is $q/2$, and its variance (``noise power'') is $\sigma_q^2 = q^2/12$ (see §G.3 for a derivation of this value).

The rms level of the quantization noise is therefore $\sigma_q = q/(2\sqrt{3}) \approx 0.3q$, or about 60% of the maximum error.

The number system (see Appendix G and number of bits chosen to represent signal samples determines their available dynamic range. Signal processing operations such as digital filtering may use the same number system as the input signal, or they may use extra bits in the computations, yielding an increased ``internal dynamic range''.

Since the threshold of hearing is near 0 dB SPL, and since the ``threshold of pain'' is often defined as 120 dB SPL, we may say that the dynamic range of human hearing is approximately 120 dB.

The dynamic range of magnetic tape is approximately 55 dB. To increase the dynamic range available for analog recording on magnetic tape, companding is often used. ``Dolby A'' adds approximately 10 dB to the dynamic range that will fit on magnetic tape (by compressing the signal dynamic range by 10 dB), while DBX adds 30 dB (at the cost of more ``transient distortion''). In general, any dynamic range can be mapped to any other dynamic range, subject only to noise limitations.