CHAPTER 3

Sampling and Analog to Digital Conversion

3.1 INTRODUCTION

Generally, signals are representations of the behavior of physical phenomena, as well as natural or artificial processes. Those signals normally are functions of independent continuous variables, for example, muscle contraction and expansion as function of time. To be processed by digital systems, continuous signals need to be converted to a digital format. Figure 3.1 represents the steps to obtain a digital representation of an analog signal $x(t)$. The Sampler and discrete-time converter block converts the analog signal $x(t)$ to a discrete-time signal $x(n)$ by taking the amplitude values (samples) at specific time instants. Next, the Quantizer block converts real values of $x(n)$ to discrete values $x_q(n)$, which are finally converted to the digital representation, for example, binary, by the Encoder block. The encoded representation can be stored, transmitted, or processed by digital systems.

In this chapter, we detail each block of Figure 3.1 and describe how the signal is transformed in both time and frequency domains.

Figure 3.1. Block diagram of analog to digital conversion.

3.2 SIGNAL SAMPLING AND RECONSTRUCTION

Consider a continuous-time signal $x(t)$ with spectral components limited to $\Omega_x = 2\pi B\, rad/s$, that is, $X(j\Omega) = 0, |\Omega| > 2\pi B$, represented in

Figure 3.2. A sampled version of $x(t)$, $x_S(t)$, consists of a sequence of values taken from $x(t)$ at equally spaced time intervals T_S.

Signal $x_S(t)$ is obtained multiplying $x(t)$ by a periodic impulse train with period T_S, given by $s(t) = \sum_{k=-\infty}^{\infty} \delta(t - kT_s)$. Thus,

$$x_s(t) = x(t)s(t) = x(t)\sum_{k=-\infty}^{\infty} \delta\left(t - kT_s\right) = \sum_{k=-\infty}^{\infty} x(kT_s)\delta\left(t - kT_s\right), \quad (3.1)$$

represented in Figure 3.3.

In frequency, the sampling operation represented by Equation 3.1 consists of convolving $X(j\Omega)$ and the Fourier transform of $s(t)$, yielding:

$$S(j\Omega) = \frac{2\pi}{T_s}\sum_{m=-\infty}^{\infty} \delta(\Omega - m\Omega_s), \quad \Omega_s = \frac{2\pi}{T_s},$$

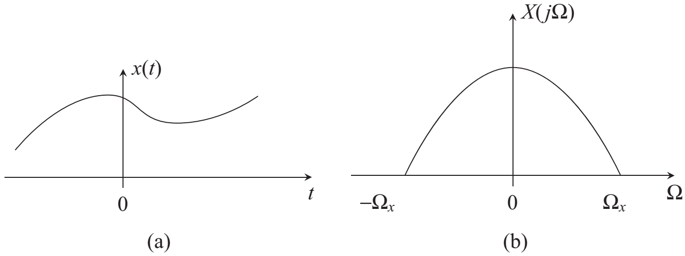


Figure 3.2. (a) Continuous-time signal $x(t)$ and (b) frequency spectrum of $x(t)$.

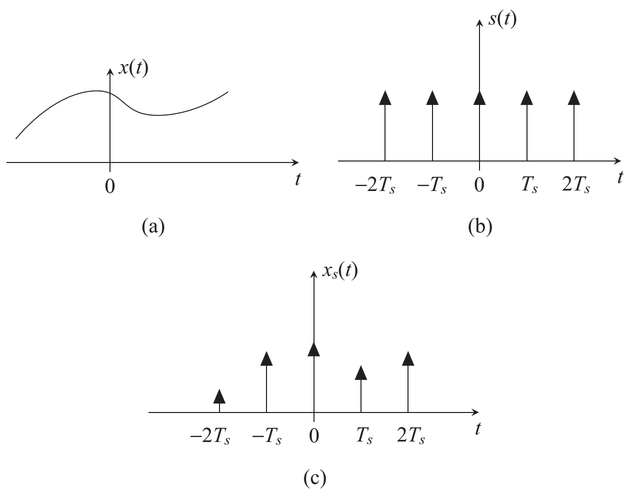


Figure 3.3. (a) Analog signal $x(t)$, (b) impulse train $s(t)$, and (c) product $x_s(t) = x(t)s(t)$.

represented in Figure 3.4. The result of the convolution is:

$$X_S(j\Omega) = \frac{1}{2\pi} X(j\Omega) * S(j\Omega) = \frac{1}{2\pi} X(j\Omega) * \frac{2\pi}{T_S} \sum_{m=-\infty}^{\infty} \delta\left(\Omega - m\Omega_S\right)$$

$$= \frac{1}{T_s} \sum_{m=-\infty}^{\infty} X\left[j\left(\Omega - m\Omega_s\right)\right]. \tag{3.2}$$

Thus, the spectrum of $x_s(t)$ consists of replicas of $X(j\Omega)$, the spectrum of $x(t)$, centered at integer multiples of $\Omega_s = \dfrac{2\pi}{T_s}$, as represented in Figure 3.5.

The value of sampling period T_s must be chosen such that $x(t)$ can be exactly recovered from $x_s(t)$ by lowpass filtering. For that, an ideal filter with cut-off frequency $\Omega_x < |\Omega_c| < \Omega_s - \Omega_x$ must be used, such that the filtered spectrum is the replica of $X(j\Omega)$ centered at the origin. The other replicas centered at multiples of Ω_s are eliminated.

To avoid superposition of the spectral components of $X(j\Omega)$, the value of $\Omega_x - \Omega_s$ must be greater than Ω_x, see Figure 3.5, so:

$$\Omega_s - \Omega_x > \Omega_x \Rightarrow \Omega_s > 2\Omega_x.$$

This result is established in the Nyquist theorem. When $\Omega_s < 2\Omega_x$ the replicas of $X(j\Omega)$ in $X_S(j\Omega)$ overlap, resulting in what is known as aliasing; in that case recovery of x(t) by ideal lowpass filtering is not possible.

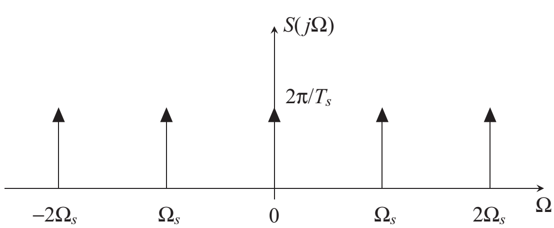


Figure 3.4. Frequency spectrum of the impulse train $s(t)$.

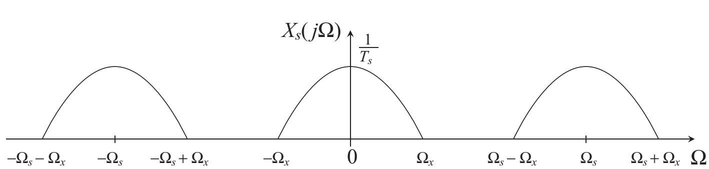


Figure 3.5. Spectrum of the sampled signal $x_s(t)$ with $\Omega_s > 2\Omega_x$.

Recovery of $x(t)$ from its samples $x_s(t)$ can be performed by an ideal lowpass filter with frequency response:

$$H_{LP}(j\Omega) = T_s\, rect\left(\frac{\Omega}{\Omega_s/2}\right),\; rect(x) = \begin{cases} 1 & |x| \leq 1, \\ 0, & |x| > 1 \end{cases}$$

and impulse response:

$$h_{LP}(t) = sinc\left(\frac{\pi t}{T_s}\right) \text{ where } sinc(x) = \sin(\pi x)/\pi x.$$

The recovered signal $x_r(t)$ in the frequency domain is represented in Figure 3.6 and can be expressed as:

$$X_r(j\Omega) = H_{LP}(j\Omega) X_s(j\Omega) = T_s\, rect\left(\frac{\Omega}{\Omega_s/2}\right) X_s(j\Omega),$$

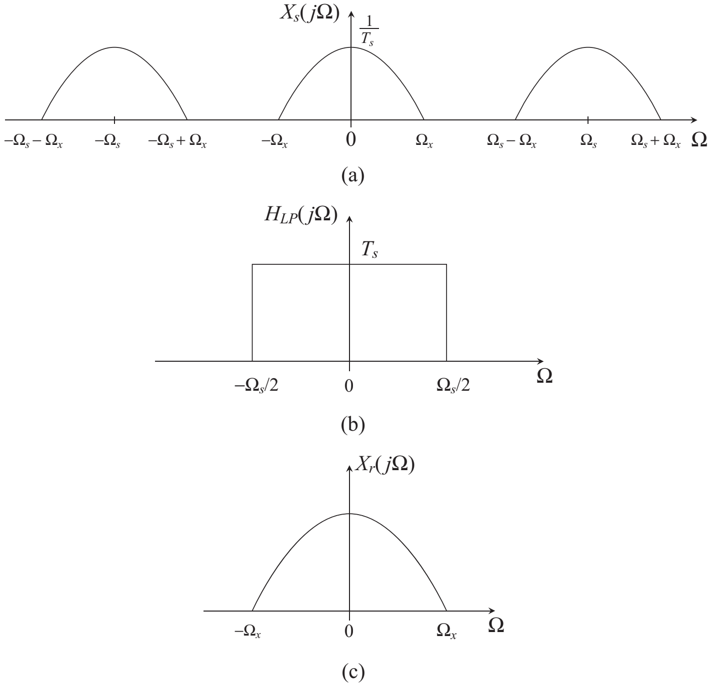
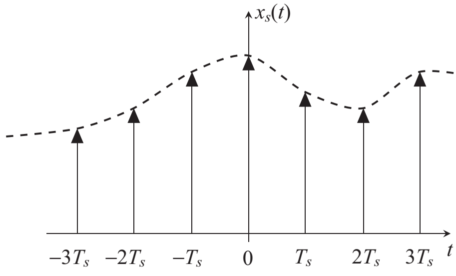
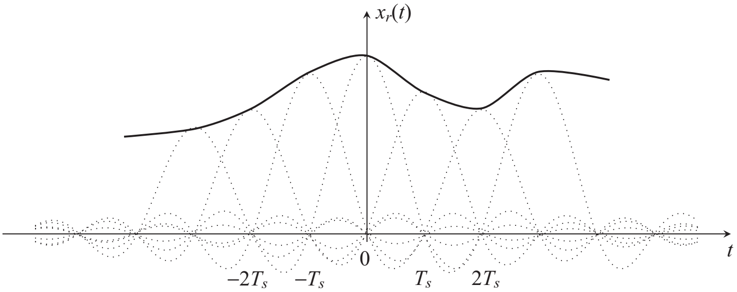


Figure 3.6. Signal reconstruction in the frequency domain (a) $X_s(j\Omega)$, (b) $H_{LP}(j\Omega)$, and (c) $X_r(j\Omega)$.



(a)



(b)

Figure 3.7. (a) Sampled signal $x_s(t)$ and (b) recovered signal $x_r(t)$.

and in time domain as:

$$x_r(t) = x_s(t) * h_{LP}(t) = \sum_{n=-\infty}^{\infty} x(nT_s)\left[\delta(t - nT_s) * h_{LP}(t)\right]$$

$$= \sum_{n=-\infty}^{\infty} x(nT_s) h_{LP}(t - nT_s) = \sum_{n=-\infty}^{\infty} x(nT_s)\, sinc\left(\frac{\pi(t - nT_s)}{T_s}\right)$$

Time operation is represented in Figure 3.7. As ideal filtering is used, $x_r(t) = x(t)$.

Considering the diagram of Figure 3.1, the sampled version $x_s(t)$ is passed through a discrete-time converter producing $x(n) = x_s(nT_s)$, and the discrete-time signal is processed to produce a digital version, as shown in next sections.

3.3 ANALOG-TO-DIGITAL CONVERSION

In the previous section, a signal $x(n)$ was obtained by uniformly sampling and converting to discrete-time a continuous-time signal $x(t)$. To process such signal using digital computers it is necessary to discretize its amplitude values, as digital processors can only process digital values represented by binary words (zeros and ones). In this section, we examine how analog-to-digital conversion is done and discuss the related issues.

3.3.1 UNIFORM AND NON-UNIFORM QUANTIZATION

The initial step of converting $x_s(t)$ from analog to digital consists of discretizing its range of amplitude values. Consider $x(n)$ a signal for which $Max[x(n)] = -Min[x(n)] = A$, that is, the amplitudes of $x(n)$ vary between A and $-A$.

Discretization starts by choosing the number L of steps that the range of amplitude values will be divided into. This process is called quantization, and each step is called quantization level. In each quantization level, values of amplitude are approximated by the upper or lower limit of the level. As an example, in Figure 3.8, for an input signal range between -4Δ and 3Δ and $L = 8$ quantization levels, values of $x(n)$ between $\frac{\Delta}{2}$ and 1.5Δ are converted into Δ.

Quantization provides discrete amplitude representation $x_q(n)$ of the discrete-time signal $x(n)$. As values of $x_q(n)$ are approximations of $x(n)$, quantization implies in increasing error in signal representation.

If the quantization levels are equally spaced, quantization is said to be uniform. However, if the analog signal has small amplitude variations followed by greater variation at other instants, it may be interesting not

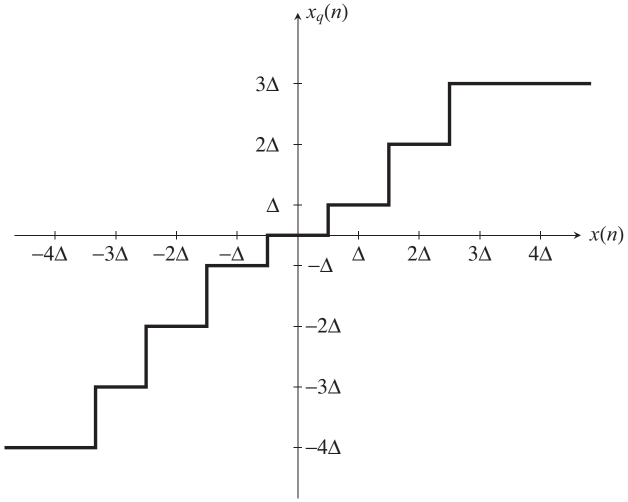


Figure 3.8. Quantization levels for uniform quantization.

to use uniform quantization, as all small values can be associated to the same level. In this case, it can be used a companding (short name for compress and expand) system that processes the input signal $x(n)$ using a non-linearity to obtain equally distributed amplitudes that are passed through an uniform quantizer.

Definition of how to distribute quantization levels in non-uniform quantization depends on the application. For voice transmission systems, performance improvement is obtained by concentrating the quantization levels in low amplitude regions. In some cases non-linear distribution of quantization levels can be utilized, as represented in Figure 3.9. Definition of non-linear distribution depends on the application, one example being the μ-law given by Equation 3.3, used in voice telephony systems:

$$F(x) = \frac{\operatorname{sgn}(x(n))\log(1+\mu\,|\,x(n)\,|)}{\log(1+\mu)}. \tag{3.3}$$

After quantization, samples of $x(t)$ are discrete in time and amplitude. The discrete amplitudes can be represented by a finite alphabet. For example, if a binary representation is chosen, each quantization step is coded by a binary word of $\log_2(L)$ bits. Finally, the digital samples can be processed by a digital signal processor or a field gate array (FPGA) or another digital processor.

Amplitude quantization naturally introduces an error in the signal representation, as a real value is approximated by a discrete value. Also, all values in a quantization step are converted to the same discrete value.

The whole process of sampling, quantization, and finite alphabet representation is named analog-to-digital conversion, and the system performing this conversion is the analog-to-digital converter (ADC), specified as described in the next section.

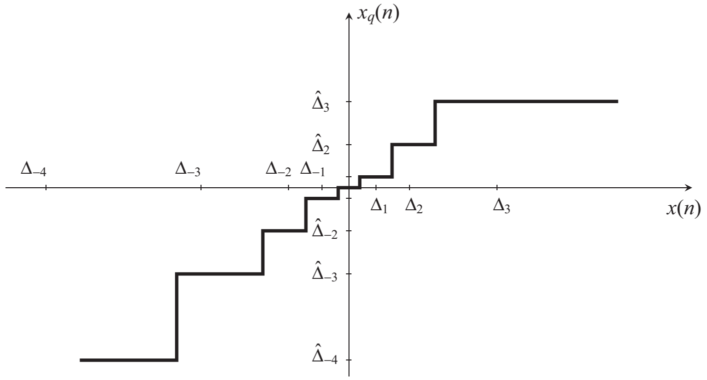


Figure 3.9. Quantization levels for non-uniform quantization.

3.3.2 ANALOG-TO-DIGITAL CONVERTER SPECIFICATIONS

For a Nyquist-based ADC, the first parameter to be specified is the sampling rate. Given in Hertz, the sampling rate is the maximum value of the ADC conversion rate, being twice the maximum frequency of the input signal. For example, an ADC with sampling rate $F_S = 8\ kHz$ samples signals of maximum frequency $F_{max} = 4\ kHz$.

The second parameter is the dynamic range given in volts, which specifies the maximum and minimum values of input signal amplitude. It is common to have a non-symmetrical dynamic range, for example, from 0 to 3 V.

The third parameter is the number of bits to represent each sample value. This parameter is the resolution of the ADC and is a tradeoff with the sampling rate: when the rate increases, the number of bits per sample decreases.

3.3.3 DECIMATION AND INTERPOLATION OF DISCRETE-TIME SIGNALS

In some applications, it is necessary to increase or decrease the sampling rate of a discrete-time signal $x(n)$. Those operations are called decimation and interpolation, respectivelly, as presented in this section.

Decimation of a discrete-time signal $x(n)$ consists of multiplying the signal by a discrete-time impulse sequence $s_N(n) = \sum_{k=-\infty}^{\infty} \delta(n-kN)$. For each set of N samples, this operation maintains one and discards the other $N-1$ samples.

The decimated sequence can be written as:

$$x_{dec}(n) = x(n)s_N(n) = \sum_{k=-\infty}^{\infty} x(kN)\delta(n-kN).$$

If $x(n)$ was obtained by sampling a continuous-time signal $x_c(t)$ with a sampling period T_s, decimation is equivalent to reducing the original sample rate by an integer value. Considering that originally $x(n) = x_c(nT_s)$, sampling the discrete signal by N implies in obtaining a signal $x_{dec}(n) = x(nN) = x_c(nNT_s)$. This is equivalent to sampling $x_c(t)$ with sampling period NT_s.

In the frequency domain, decimation of $x(n)$ with spectrum $X(e^{j\omega})$ will produce the spectrum given by:

$$X_{dec}(e^{j\omega}) = \frac{1}{N}\sum_{k=0}^{N-1} X\left(e^{j\left(\omega - \frac{2\pi k}{N}\right)}\right). \tag{3.4}$$

Equation 3.4 means that decimating a signal by N implies to produce copies of $X(e^{j\omega})$ centered at frequencies $\frac{2\pi k}{N}$. Even for band-limited $x(n)$, that is, $X(e^{j\omega}) = 0$ for $\omega_x \le |\omega| \le \pi$, depending on the chosen N, aliasing may occur. In general, to avoid aliasing after decimation, it must be guaranteed that:

$$N\omega_x < \pi.$$

This condition is satisfied by filtering the signal $x(n)$ before decimating using a lowpass filter with cut-off frequency π / N, as represented in Figure 3.10.

The inverse of decimation is discrete interpolation, which increases the sampling rate by introducing $L-1$ samples between consecutive samples of a discrete-time signal $x(n)$. Ideally, the inserted values correspond to samples of the continuous signal $x_c(t)$ obtained using a sampling rate $T_i = T_s/L$. The interpolated signal is expressed as $x_i(n) = x_c(nT_i)$. Figure 3.11(a) represents the original discrete-time signal $x(n)$ and Figure 3.11(b) the interpolated signal $x_i(n)$ for $L = 4$.

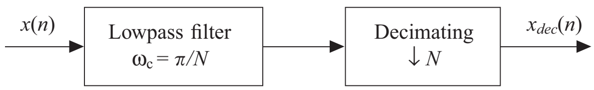


Figure 3.10. Block diagram for decimation.

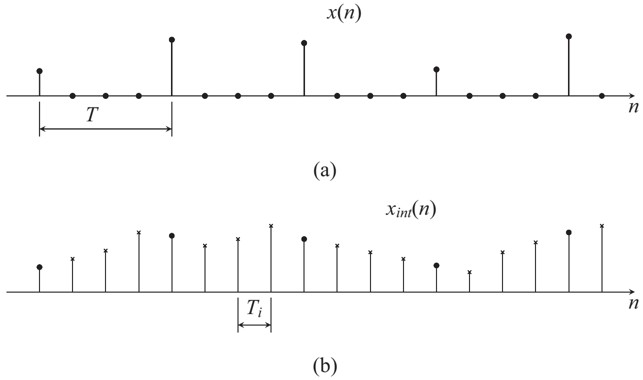


Figure 3.11. Discrete-time interpolation: (a) original $x(n)$, (b) interpolated $x_i(n)$, using $L = \frac{T}{T_i} = 4$.

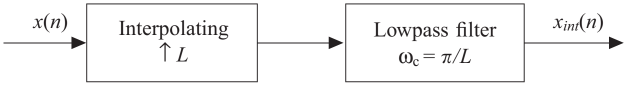


Figure 3.12. Block diagram for interpolation.

The spectrum of the interpolated signal is a scaled version of $X(e^{j\omega})$, the spectrum of $x(n)$, that is, $X_i(e^{j\omega}) = X(e^{j\omega L})$. To produce a periodic spectrum with period 2π, a lowpass filter is applied to $X(e^{j\omega})$, as represented in Figure 3.12.

Interpolation quality depends on the values introduced between samples. Those values can be obtained by zero-order interpolation, linear interpolation, or any other interpolation method. The choice of interpolation method is dictated by the tradeoff between approximation quality and computational cost. Increasing the approximation order increases the computation cost.

3.4 CHAPTER OVERVIEW

In this chapter, we present the process of converting a band-limited analog signal to a digital representation. Starting from the mathematical aspects of taking uniformly spaced samples of an analog signal, the fundamental limit of the Nyquist theorem is established. Next, we define the phenomenum known as aliasing and show the need for anti-aliasing filtering. Having presented the sampling process, we described the inverse operation, that is, the reconstruction of the analog signal from its samples. We show that, in the frequency domain, this operation is implemented by lowpass filtering, which corresponds to interpolation in the time domain.

The steps of quantizing and encoding a sequence of signals samples in order to obtain a digital representation is described. We present uniform and non-uniform quantization and the binary representation of the quantized samples. Some error sources present in analog-to-digital conversion are discussed, and some parameters for ADCs described.

The chapter ends with definition of decimation and interpolation for, respectively, reducing and increasing the sampling rate of discrete-time signals. Those operations are analyzed in both time and frequency domains.