Lec 2: Discrete-time signals and systems

2.1 Introduction

In this chapter, we'll look at discrete-time signals and systems. As described in the first chapter, a signal is a function of independent variables such as time, distance, position, temperature, pressure, etc. Most signals are generated naturally but a signal can also be generated artificially using a computer and signals can be in any number of dimensions (1D, 2D, 3D, etc). We'll be mainly studying time series signals, which are 1D signals with amplitude, pressure, intensity, etc as a function of time. Now, what about discrete-time systems?

Discrete-time systems operate on an input signal, according to some prescribed function and produce another output signal. For example, the input sequence could be a noisy electrocardiogram (ECG) signal and the system outputs a noise reduced ECG signal. In this example **(see Figure 2.1**), the discrete-time system is a band-pass filter that removes low frequency baseline noise and high frequency power line interference.

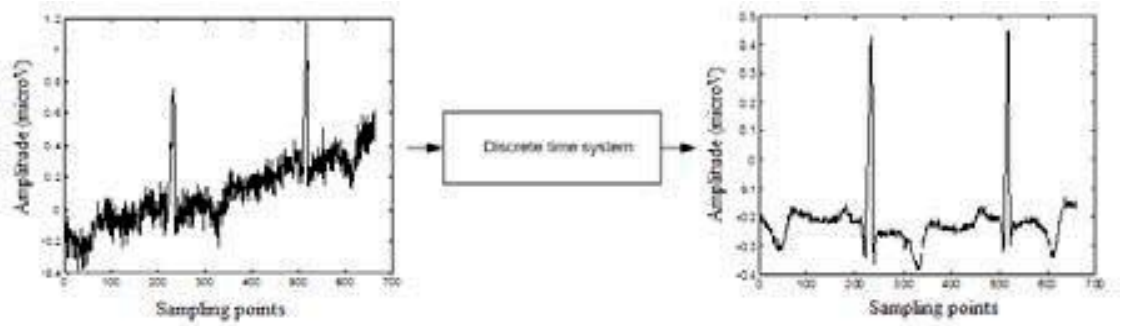

Figure 2.1: Band-pass filter as an example of a discrete time system.

There are some basic operations for discrete-time systems, which we will study in this chapter but before we do that, we need to understand that discrete-time systems operate on discrete-time signals and we need to obtain the latter from analogue signals.

2.2 Discrete-time signal

In general, biological signals that are recorded are analogue and to process analogue signals by digital means (like using a computer), we need to convert them to discrete-time form, i.e. to convert them to a sequence of numbers defined at specific uniform intervals. This process is known as sampling.

2.2.1 Sampling

Since most biological signals are 1D time series signals, we'll focus our attention to such signals where the independent variable is time. A discrete-time signal, $x[n]$ is developed by uniformly sampling an analogue signal $x(t)$ as indicated **in Figure 2.2**. It is done by taking samples at specific uniform intervals of time (every value of $x[n]$ is called a sample) and the sampling interval is the time of one of these uniform intervals while the sampling frequency is the number of uniform time intervals in one second normally specified in Hz. Then, discrete variable n can be normalized to assume integer values as a representation of t.

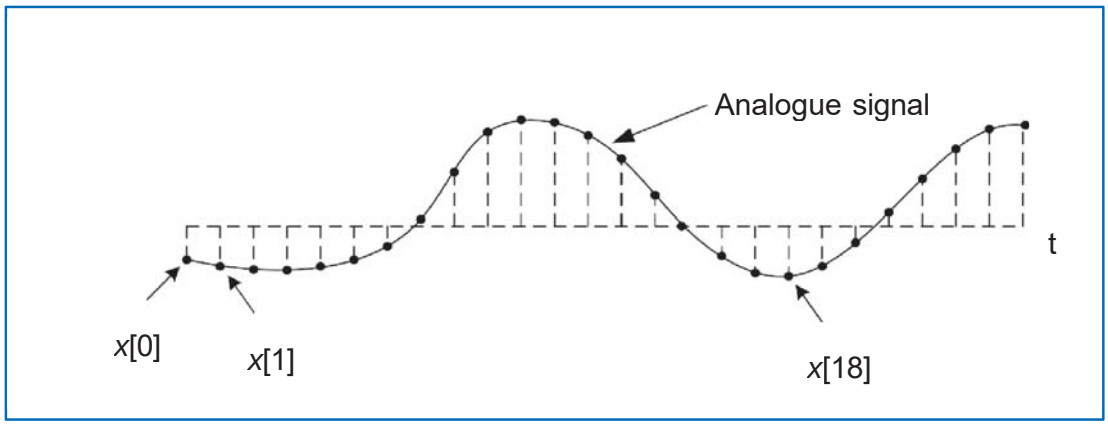

Figure 2.2: Obtaining a discrete-time signal from analogue signal through sampling.

2.2.2 Aliasing

Aliasing is a problem in the sampling process as it causes ambiguities in reconstruction, i.e. distorts the sampled signal unrepresentative of the original signal. To avoid aliasing and be able to reconstruct the original signal without errors, the sampling frequency has to be more than twice of the highest frequency contained in $x(t)$. This is known as Nyquist theorem and the minimum frequency known as Nyquist frequency (rate).

Consider the following sampling example of an analogue signal with sufficient sampling frequency.

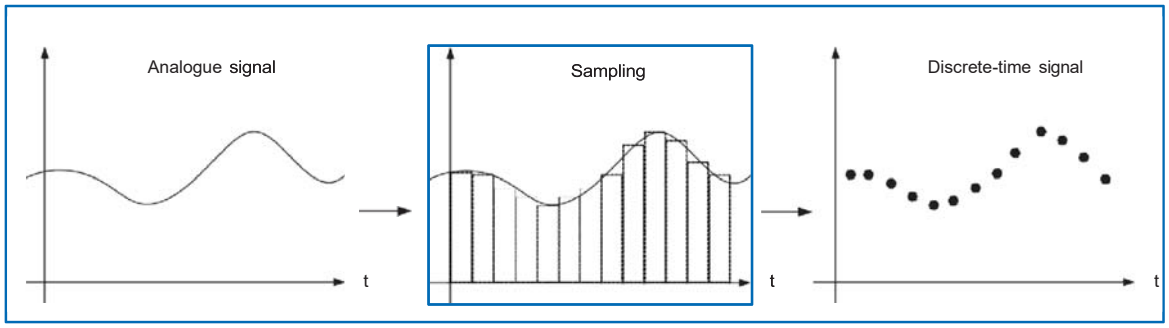

Figure 2.3: Sampling with high enough frequency – the signal is *correctly* represented.

Figure 2.4 shows the problem of aliasing with insufficient sampling frequency. It can be seen that the analogue signal in Figure 2.4 is not represented correctly by the discrete-time version.

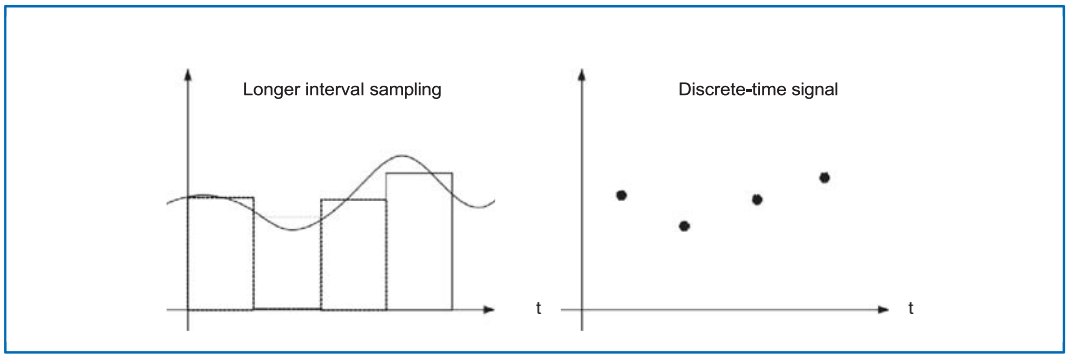

Figure 2.4: Aliasing problem – effects of sampling frequency below Nyquist frequency.

Example: Consider the analogue signal $x(t) = 3\cos 50\pi t + 10\sin 300\pi t - \cos 100\pi t$. What is the Nyquist frequency for this signal?

Answer: Use the generic term, $A\cos 2\pi t$ or $A\sin 2\pi t$ to compute the frequencies present in the signal, which are 25 Hz, 150 Hz and 50 Hz. So, Nyquist frequency is 2* highest frequency= 2*150 Hz = 300 Hz. In practise, we normally sample at a much higher rate than Nyquist frequency.

2.3 Sequences

Sometimes, a discrete-time signal is known as a sequence and vice versa. A discrete-time signal may be a finite length or an infinite-length sequence, e.g. $x[n]=1-2n^4$, $-5 \le n \le 2$ is a finite length sequence with length $2-(-5)+1=8$ but $x[n]=\sin(0.1n)$ is an infinite-length sequence.

An important and perfectly valid operation with sequences is zero padding. An N length sequence can be increased by padding with zeros in the beginning or in the end. For example, a sequence with length 3 can be changed to length 7 by padding with 4 zeros:

$$x[n] = n^4, \quad 1 \le n < 4; \tag{2.1}$$

$$x_{pad}[n] = \begin{cases} n^4, & 1 \le n < 4; \\ 0, & 4 \le n \le 7. \end{cases}$$

2.4 Basic Discrete-time System Operations

There are several common operations that we'll frequently encounter for discrete-time systems and we'll look at a few here.

2.4.1 Product (modulation)

Product operation is given as:

$$y[n] = w[n].x[n]. \tag{2.2}$$

It is frequently used in windowing, where a discrete-time finite signal $y[n]$ is obtained from a discrete-time infinite signal $x[n]$ using a window, $w[n]$.

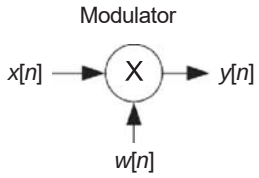

Figure 2.5: Product operation.

2.4.2 Addittion

Addition operation can be performed as given below to add two sequences,

$$y[n] = x[n] + w[n] \tag{2.3}$$

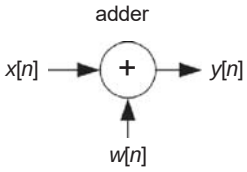

Figure 2.6: Addition operation.

2.4.3 Multiplication

Multiplication operation is used to amplify or to attenuate a signal:

$$y[n] = A.x[n] \quad or \quad y[n] = Ax[n]. \tag{2.4}$$

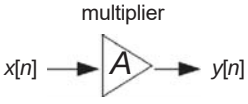

Figure 2.7: Multiplication operation.

2.4.4 Time reversal (folding)

Folding operation is important for filtering and is given by:

$$y[n] = x[-n]. \tag{2.5}$$

The input sequence is flipped at n=0 to produce the output sequence.

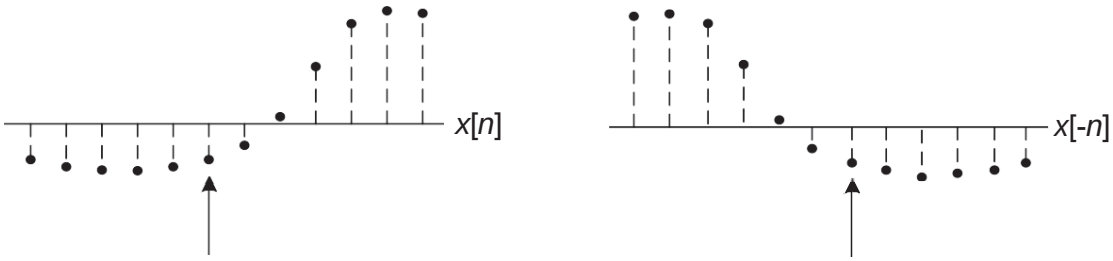

Figure 2.8: Folding operation (arrow points to n=0).

2.4.5 Branching

Branching is used to provide multiple copies of the input sequence:

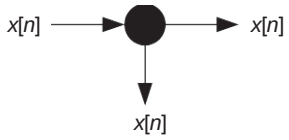

Figure 2.9: Branching operation.

2.4.6 Time shifting

Time shifting denotes delaying or advancing the input by N samples:

$$\begin{aligned} y[n] &= x[n-N] \quad \text{(delay)} \\ y[n] &= x[n+N] \quad \text{(advance)} \end{aligned} \tag{2.6}$$

Figure 2.8: Folding operation (arrow points to n=0).