# 1 Lecture Outline

**Reading: Chapter 3 Discrete-Time Systems (review)**

- Sample-by-sample processing vs. block processing
- Input/Output Rules
- Convolution

Note that we will only introduce a few new concepts in Chapter 3 since this chapter is mainly a review from a standard undergraduate course in signals and systems.

# 2 Discrete-Time Systems (3)

Depending on the application and hardware, digital filtering operation can be organized to operate either on a block basis or a sample-by-sample basis.

In the sample processing case, the input samples are processed one at a time as they arrive at the input. The filter operates as a state machine; that is, each input sample is used in conjunction with the current internal state of the filter to compute the current output sample and also to update the internal state of the filter in preparation for processing the next input sample.

In the block processing case, the input signal is segmented into blocks of samples or frames, each block is processed, and an output block of samples is generated. The filter's internal state is also updated appropriately.

Either approach can be used in real-time applications provided that the output rate in samples/s or blocks/s is commensurate with the input rate in samples/s or blocks/s.

As a reminder, the output of an LTI system is given by the discrete-time convolution

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

where $h(k)$ is the impulse response of the system. Alternatively, the output of an LTI system is given by the discrete-time linear, constant-coefficient difference equation (LCCDE)

$$y(n) = \sum_{k=1}^{M} a_k y(n-k) + \sum_{k=0}^{L} b_k x(n-k) \tag{2}$$

where $a_k$ are feedback coefficients and $b_k$ are feedforward coefficients.

# 3 Input/Output Rules (3.1)

Consider the DT system or Finite Impulse Response (FIR) digital filter

$$y(n) = 2x(n) + 3x(n-1) + 4x(n-2). \tag{3}$$

The impulse response of the system is simply

$$h(n) = 2\delta(n) + 3\delta(n-1) + 4\delta(n-2). \tag{4}$$

 

**Example 3.1.2:** Suppose the system is initially at rest and the first four input samples are $x_0, x_1, x_2, x_3$. The first four output samples, computed sample-by-sample are

$$\begin{aligned} y_0 &= 2x_0 \\ y_1 &= 2x_1 + 3x_0 \\ y_2 &= 2x_2 + 3x_1 + 4x_0 \\ y_3 &= 2x_3 + 3x_2 + 4x_1 \end{aligned} \tag{5}$$

The complete convolution would involve computing six output samples because the system has 2 words of memory–the memory is used to compute output samples even when the input is turned off:

$$\begin{aligned} y_0 &= 2x_0 \\ y_1 &= 2x_1 + 3x_0 \\ y_2 &= 2x_2 + 3x_1 + 4x_0 \\ y_3 &= 2x_3 + 3x_2 + 4x_1 \\ y_4 &= 3x_3 + 4x_2 \\ y_5 &= 4x_3 \end{aligned} \tag{6}$$

**Example 3.1.3:** If we treat the first four input samples as a block $[x_0, x_1, x_2, x_3]$, the output block can be expressed as

$$\mathbf{y} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 3 & 2 & 0 & 0 \\ 4 & 3 & 2 & 0 \\ 0 & 4 & 3 & 2 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \mathbf{H}\mathbf{x} \tag{7}$$

where $\mathbf{H}$ is the convolution matrix, i.e. the matrix whose rows are the flipped (time-reversed) and shifted impulse response.

**Example 3.1.4:** Example 3.1.2 can also be cast in an equivalent sample-by-sample processing form described by the following system of three equations, i.e. state-space representation:

$$\begin{aligned} y(n) &= 2x(n) + 3w_1(n) + 4w_2(n) \\ w_2(n+1) &= w_1(n) \\ w_1(n+1) &= x(n) \end{aligned} \tag{8}$$

The auxiliary signals $w_1(n)$ and $w_2(n)$ can be thought of as the "internal states" of the system. The present input sample $x(n)$ together with the knowledge of the present internal states $\{w_1(n), w_2(n)\}$ is sufficient to compute the present output $y(n)$.

Figure 1: Digital FIR filtering (state-space representation)

Consider the Infinite Impulse Response (IIR) digital filter

$$y(n) = 0.5y(n-1) + 2x(n) + 3x(n-1). \tag{9}$$

The causal impulse response of the system, $h(n)$ can be computed with Fourier transforms:

$$\begin{aligned} y(n) &= 0.5y(n-1) + 2x(n) + 3x(n-1) \\ &\updownarrow \\ Y(\omega) &= 0.5e^{-j\omega}Y(\omega) + 2X(\omega) + 3e^{-j\omega}X(\omega) \end{aligned} \tag{10}$$

or

$$Y(\omega)\left[1 - 0.5e^{-j\omega}\right] = X(\omega)\left[2 + 3e^{-j\omega}\right]. \tag{11}$$

By definition, the transfer function is given by

$$Y(\omega)/X(\omega) = H(\omega) = \frac{2 + 3e^{-j\omega}}{1 - 0.5e^{-j\omega}} \tag{12}$$

and through an inverse transformation, yields

$$h(n) = 2\left(\frac{1}{2}\right)^n u(n) + 3\left(\frac{1}{2}\right)^{n-1} u(n-1). \tag{13}$$

**Example 3.1.6:** This IIR digital filter has the state-space representation:

$$\begin{aligned} y(n) &= 0.5w_1(n) + 2x(n) + 3v_1(n) \\ w_1(n+1) &= y(n) \\ v_1(n+1) &= x(n) \end{aligned} \tag{14}$$

The auxiliary signals $w_1(n)$ and $v_1(n)$ can be thought of as the "internal states" of the system. The present input sample $x(n)$ together with the knowledge of the present internal states $\{w_1(n), v_1(n)\}$ is sufficient to compute the present output $y(n)$.

Figure 2: Digital IIR filtering (state-space representation)