# 8

# DISCRETE-TIME SYSTEMS

## OUTLINE



## OBJECTIVES

1. Define the key signal for discrete-time systems.
2. Define a normalized frequency variable for discrete-time systems.
3. Describe a linear difference equation with constant coefficients.
4. Find the transfer function of a difference equation.
5. Determine the stability of a discrete-time system.
6. Identify basic MATLAB commands for discrete-time systems.

## INTRODUCTION

Now that we know how to sample a signal properly, we want to begin real-time digital signal processing. Each input voltage sample will have its amplitude converted to a numerical value, and the number will be delivered to a computer. The computer will quickly calculate a corresponding output number and deliver it to a port, where it will be converted back to a voltage amplitude. This processing cycle repeats continuously. The basic hardware required for this process is discussed in Appendix A. Since we do not have hardware in this text, we will have to simulate the process by filtering files or MATLAB vectors containing the numerical input data.

The development of discrete-time systems will parallel our development of continuous-time systems. The program the computer runs is a linear difference equation. These are similar to differential equations and are easily solved for sampled exponential signals. Relationships between input and output discrete-time phasors will result in transfer functions. From the transfer functions we will be able to sketch the system's pole-zero plot or determine its frequency response to sampled sinusoids. The transfer function also will reveal the system's natural response and, therefore, tell us whether or not the system is stable.

## 8.1 THE *z* DOMAIN

In working with linear continuous-time systems, we chose to specialize in exponentially increasing or decreasing sinusoids and to invoke linearity to further simplify the signals to $v(t) = \vec{V}e^{st}$. This resulted in a generalization of frequency to $s = \sigma + j\omega$ and defined an $s$ domain. Linear discrete-time systems will build on this foundation. After completing this section you will be able to:

- Define the key-signal for discrete-time systems.
- Relate the $s$ domain and the $z$ domain.

The key signal for continuous-time systems is $v(t) = \vec{V}e^{st}$. If we look at this signal only at discrete values of time given by $t = nT_s$, where $n$ is an integer and $T_s$ is the sampling interval, the fundamental discrete-time signal becomes

$$v(t) = v(nT_s) = \vec{V}e^{snT_s} = \vec{V}(e^{sT_s})^n \qquad \textbf{(8.1)}$$

Some notation changes are appropriate to reflect the fact that we will be processing numbers rather than voltages or currents. There are applications in switched capacitor circuits where the signal is indeed carried along in the form of a voltage (or charge) on capacitors. In most applications, however, the signals of interest are simply a sequence of numbers whose original units are no longer of importance. The number sequence may be created by sampling a continuous-time voltage or current, but it also may be generated by a computer program or delivered by a compact disk or other storage device. We will consequently adopt a generic $\{x\}$ to denote the input number sequence, and use $\{y\}$ to represent the output sequence.

Since $s$ is a complex constant, $e^{sT_s}$ is just another complex constant. We may as well simplify our notation and define

$$z \equiv e^{sT_s} \tag{8.2}$$

which makes our key discrete-time signal

$$x(n) = \vec{X}z^n \tag{8.3}$$

Equation 8.3 provides the transition between the time or *sample domain* and the frequency or *z domain*.

The notation changes have not affected the nature of the phasor $\vec{X}$. It still represents the initial ($t = 0$) amplitude and phase information of a sinusoidal signal whose amplitude may be changing exponentially. It could be the Euler phasor or any other phasor agreed to by defining a phasor transformation.

Consider the nature of the key discrete-time signal for several common situations:

1. For a d-c signal, $s = 0$ and $z = e^{sT_s} = 1$. The key signal is $x(n) = \vec{X}(1)^n = \vec{X}$, where the phasor is a real number.
2. For a simple exponential, $s = \sigma$, and $z = e^{\sigma T_s}$. The key signal is $x(n) = \vec{X}(e^{\sigma T_s})^n$. Again the phasor is real. If $\sigma > 0$, then $z > 1$, and the signal grows exponentially with $n$. If $\sigma < 0$, then $z < 1$ and the signal decays exponentially with $n$.
3. The signal of greatest interest is still the constant-amplitude sinusoid, for which $s = j\omega$ and $z = e^{j\omega T_s} = 1\angle\omega T_s$. The key signal becomes $x(n) = \vec{X}(e^{j\omega T_s})^n$, which is interpreted through a phasor transformation.

## EXAMPLE 8.1

The signal $x(t) = 12e^{-t}\cos(4t + 20°)$ is sampled at $t = n\pi/5$ seconds. Determine the Euler phasor, $z$, and the key signal, $x(n)$.

### Solution

The Euler phasor is defined as $\vec{X} = 6\angle 20°$. Since $s = -1 + j4$,

$$z = e^{sT_s} = e^{(-1+j4)\pi/5} = e^{-\pi/5}\, e^{j0.80\pi} = 0.5335e^{j0.80\pi}$$

Converting to proper units, $20^\circ = 0.3491$ radians, so

$$\vec{X} = 6\angle 20^\circ = 6e^{j0.3491}$$

and

$$x(n) = 6e^{j0.3491}(e^{-\pi/5}e^{j0.80\pi})^n \qquad \text{or} \qquad x(n) = 6(0.5335)^n\, e^{j(0.80\pi n + 0.3491)}$$

The signal is a sequence of samples that will be interpreted the same way regardless of whether they were obtained using an appropriate sampling rate or not.

## 8.2 NORMALIZED FREQUENCY

In sampled systems, the signal frequency is only unique relative to the sampling frequency. Furthermore, system frequency response must be periodic, since sampled signal spectra are periodic in the range $f_s$. These facts suggest working with a normalized frequency variable so that the results are independent of the sampling frequency used. After completing this section you will be able to:

- Define a normalized frequency variable for discrete-time systems.
- Express a sinusoidal signal in terms of the normalized frequency.

The signal frequency, $\omega$, and the sampling interval, $T_s$, always appear together as $e^{j\omega T_s}$ in the expressions for sampled waveforms. This also gives the sampled signal a frequency spectrum that is periodic in the interval $\omega T_s = 2\pi$ or $\omega = \omega_s$. These factors suggest shifting to a normalized frequency variable and using a standard range for the frequency domain information. Two normalized frequency variables are suggested, as indicated in Equation 8.4. (Also see Figure 8.1.) Both are used in the literature. We will prefer the $\upsilon$ variable, which measures the frequency relative to the folding frequency of $f_s/2$.

$$\omega T_s = 2\pi f T_s = 2\pi\underbrace{(f/f_s)}_{\Omega} = \pi\underbrace{(2f/f_s)}_{\upsilon} \tag{8.4}$$

As with the DFT, the frequency range used for signal spectra may be taken as $0 \le \upsilon < 2$, since it represents one full period of the spectrum, or it may be shifted to show the baseband region of $-1 \le \upsilon < 1$. Usually the frequency response of a discrete-time system is given for the phasor of the positive exponent only, $z = e^{+j\omega T_s} = e^{j\pi\upsilon}$, just as was done for continuous-time systems. This phasor exists over the baseband range $0 \le \upsilon < 1$.

Because we always work with baseband frequencies, there is a tendency to think $\upsilon$ is confined to that range. Actually, $\upsilon$ can take on any value.

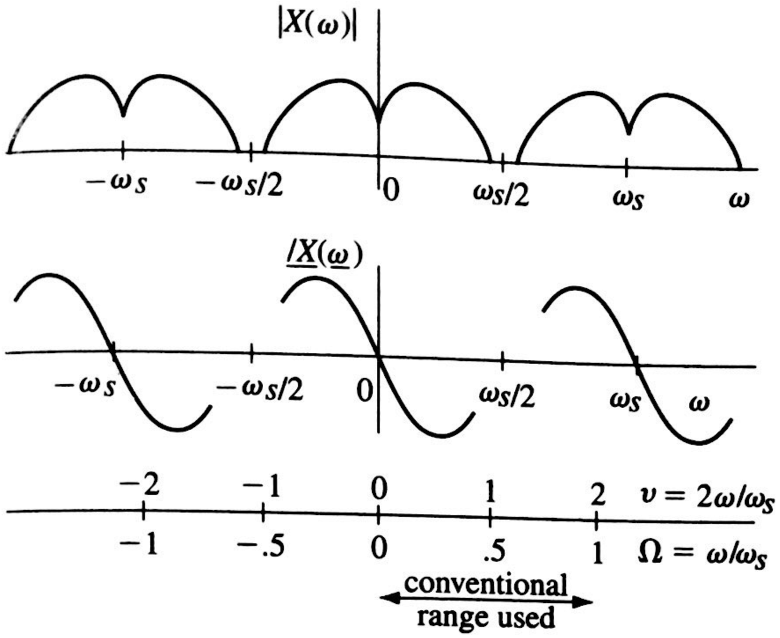



**Figure 8.1** Spectrum of a sampled real $x(t)$ and the normalized frequency variables $v$ and $\Omega$ found in the literature. The spectrum is conventionally given for $0 \le v < 2$ or for half that range. The frequency response of discrete-time systems is normally given over the range $0 \le v < 1$.

## EXAMPLE 8.2

Find the values of $z$ and $v$ associated with the sampled sinewave $x(n) = 6\sin(3n + 30°)$ and recommend a phasor for the key signal.

### Solution

A phasor transformation with the sine as the base function is

$$X_m e^{-\sigma T_s n} \sin(\omega T_s n + \theta) \;\leftrightarrow\; \vec{X} z^n = [X_m e^{j\theta}] z^n$$

The phasor is $6e^{j30°}$ using this transformation.

To identify the normalized frequency, we note that

$$\omega t = \omega T_s n = 2\pi f T_s n = \pi(2f/f_s)n = \pi v n$$

Then the sampled sinusoid always takes the general form $\cos(\pi v n + \phi)$, making $v$ obvious by inspection. In this case, $\pi v=3$.

By definition,

$$z \equiv e^{sT_s} = e^{j\omega T_s} = e^{j\pi v} = e^{j3}$$

and the key signal is

$$x(n) = \bar{X}e^{j\pi v n} = [6e^{j30^\circ}]e^{j3n}$$

## 8.3 THE DIFFERENCE EQUATION

In continuous-time systems, a circuit ultimately defined the time domain relationship between input and output through a differential equation. In discrete-time systems, a computer program defines that relationship directly through a difference equation. After completing this section you will be able to:

- Describe a linear difference equation with constant coefficients.
- Classify the difference equation according to its impulse response.
- Solve a linear difference equation by iteration.
- Provide an overview of the processing cycle.

Of the many possible prescriptions that might be conceived for creating the output sequence, only *linear* difference equations with *constant coefficients*, as defined by Equation 8.5, will be considered. This equation describes how the present output $y(n)$ is to be constructed from present and past inputs and past outputs. The functional notation $y(n - i)$ indicates the value of $y(n)$ that occurred $i$ samples earlier. Such a system is both *causal* and *time-invariant*.

$$y(n) = \sum_{i=0}^{M} a_i x(n - i) - \sum_{i=1}^{N} b_i y(n - i) \tag{8.5}$$

Systems in which the $b_i$ are all zero are called $M$th-order *finite impulse response* (FIR) systems. They are inherently stable, and are especially interesting because they have no counterpart in the continuous-time world except for the trivial case of $M = 0$ (a resistive circuit).

If at least one of the $b_i$ is nonzero, Equation 8.5 describes an $N$th-order *infinite impulse response* (IIR) system. These are similar to continuous-time systems, and it will be found that many well established designs of continuous-time systems may be adapted for use in IIR filters. Such systems involve feedback, because past outputs influence the current output. Unfortunately, with feedback comes the potential for instability.

Difference equations have one significant advantage over differential equations. Any difference equation is solvable by iteration. The process simply involves creating a table, putting in the starting conditions, and filling in the successive lines in the table from the known input signal $x(n)$.

### EXAMPLE 8.3

Find the first five output values for the difference equation subjected to a discrete unit step input:

$$y(n) = x(n) - x(n-2) + \frac{1}{2}y(n-1) \qquad \text{if } x(n) = \begin{cases} 0 & n < 0 \\ 1 & n \geq 0 \end{cases}$$

#### *Solution*

Since the input is zero prior to $n = 0$, we insist that the output $y(n) = 0$ for $n < 0$. This is the property of *causality*. The iteration table is shown below. It is helpful to summarize the effects of the input terms in an $f(n)$ column. In this case, only two columns remain to be combined to determine each $y(n)$. The values in the first row should be self-evident. Values in succeeding rows depend only on the input signal and values found in previous rows of the table.

| n | $-x(n-2)$ | $+ 0x(n-1)$ | $+ x(n) = f(n)$ | $f(n)$ | $+ 0.5y(n-1)$ | $= y(n)$ |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | +0.5(1) | +3/2 |
| 2 | −1 | 0 | 1 | 0 | +0.5(3/2) | +3/4 |
| 3 | −1 | 0 | 1 | 0 | +0.5(3/4) | +3/8 |
| 4 | −1 | 0 | 1 | 0 | +0.5(3/8) | +3/16 |

Note, in this case, that the solution might be expressed as $y(n) = 0$ if $n < 0$, $y(0) = 1$, and $y(n) = 3(\frac{1}{2})^n$ if $n \geq 1$.

Although the iteration procedure is fundamental and straightforward, it does not provide much insight into the system characteristics. In addition, if the iterative procedure is done by hand calculation, an error in any row will propagate through all subsequent rows. The iterative approach also does not generally lead to a closed-form solution for $y(n)$, but it is sometimes possible to deduce a closed-form solution from the results. This was the case in Example 8.3 for $n \geq 1$. Finally, most difference equations involve many more terms than our example, making the iteration procedure impractical. Since the difference equation is a program that a processor will run, we can write the program and let the processor do the iterations.

It may be helpful to consider how the processing takes place and what attributes are needed by the processor. In real-time digital signal processing, the processor begins a sampling cycle by directing that the input signal be sampled. The sample is presented to an analog-to-digital (A/D) converter, which turns the signal amplitude into a binary number. The A/D unit delivers its latest conversion $x(n)$, directly to an input port or to an internal processor register, and it raises a flag to signal that the new sample is ready. Sometimes an interrupt is initiated by the flag. The processor collects the latest input sample and begins calculating the output specified by the difference equation.

The processor will need to have placed its most recent inputs and outputs in storage and have a fast and easy way to recover them. The linear difference equation requires that the latest input, along with past inputs and outputs, be multiplied by fixed constants and summed to calculate the next output, $y(n)$. This *multiply and accumulate* operation is fundamental to digital signal processing, and the processing chips are optimized to accomplish it efficiently. Once the latest output has been calculated, it is sent to its next destination. This often is an output port for digital-to-analog (D/A) conversion back to a continuous-time signal. The processor waits for the sampling interval to expire and then calls for the next sample to be collected. This process repeats forever.

In some applications real-time processing is not necessary. Photos from Mars, for instance, may be processed enough to show that the camera system is working properly, but fine details and precise coloring can await later processing if the signal is stored on tape. Any input number sequence may be stored in memory and processed like any other file. A general-purpose computer can be used instead of costly high-speed processors. Since this also does not require special sampling hardware, we will eventually use this technique to demonstrate some digital filtering.

## 8.4 THE TRANSFER FUNCTION

The linear difference equation is easily solved for the key signal $x(n) = \hat{X}z^n$. The result is a transfer function relating the output phasor to the input phasor as a function of $z$. After completing this section you will be able to:

- Find the transfer function of a difference equation.
- State the delay theorem.
- Find the frequency response of a discrete-time system.

In linear continuous-time systems, a voltage $v(t) = \vec{V}e^{st}$ is applied to each circuit element, and a current $i(t) = \vec{I}e^{st}$ results. This leads to the definition of *impedance* and turns differential equations into algebraic equations. The equivalent concept needed for linear difference equations is the delay theorem. For our key signal, if $x(n) = \vec{X}z^n$, then $x(n - m) = \vec{X}z^{(n-m)} = z^{-m}\vec{X}z^n = z^{-m}x(n)$.

**Delay Theorem**

$$x(n - m) = z^{-m}x(n) \tag{8.6}$$

A block diagram representation of the delay theorem, incorporating an amplitude change, is shown in Figure 8.2. The multiplication by $k$ may occur before or after the delay.

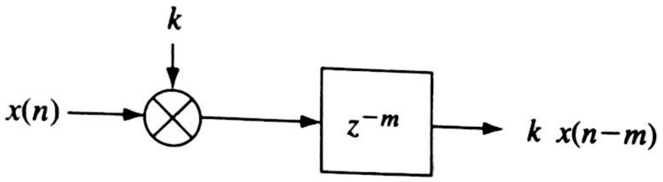



**Figure 8.2** The delay theorem may be characterized in a simple block diagram. Combining a delay with multiplication by a constant, discrete-time signals are modified in magnitude and phase, exactly as impedances modified continuous-time signals.

We hypothesize that an input signal of the form $x(n) = \vec{X}z^n$ will result in a forced response of the form $y(n) = \vec{Y}z^n$ in a linear system. Then for the class of signals having this form, a transfer function may *always* be found by starting with the difference equation

$$y(n) = \sum_{i=0}^{M} a_i x(n - i) - \sum_{i=1}^{N} b_i y(n - i)$$

substituting $x(n) = \vec{X}z^n$ and $y(n) = \vec{Y}z^n$ and applying the delay theorem

$$\vec{Y}z^n = \sum_{i=0}^{M} a_i z^{-i}\vec{X}z^n - \sum_{i=1}^{N} b_i z^{-i}\vec{Y}z^n$$

The $z^n$ terms cancel, leaving

$$\vec{Y}\left(1 + \sum_{i=0}^{N} b_i z^{-i}\right) = \sum_{i=0}^{M} a_i z^{-i}\vec{X}$$

Finally, we see that the output and input phasors are related by a ratio of polynomials in $z$:

$$\frac{\vec{Y}}{\vec{X}} = \frac{\sum_{i=0}^{M} a_i z^{-i}}{1 + \sum_{i=1}^{N} b_i z^{-i}} \tag{8.7}$$

To find the frequency response of the discrete-time system, we simply substitute $z = e^{j\pi v}$.

## EXAMPLE 8.4

A signal $x(n) = 12\cos(\pi n/4 + 30°)$ is input to a processor running the program

$$y(n) = x(n) - x(n-2) + \frac{1}{2}y(n-1)$$

Find the output sinusoid.

### *Solution*

Making the standard substitutions, we have

$$\vec{Y}z^n = \vec{X}z^n - z^{-2}\vec{X}z^n + \frac{1}{2}z^{-1}\vec{Y}z^n$$

Canceling $z^n$ terms and regrouping gives

$$\vec{Y}\left(1 - \frac{1}{2}z^{-1}\right) = \vec{X}(1 - z^{-2})$$

For the input signal given, we define a phasor transformation

$$12\cos\left(\frac{n\pi}{4} + 30°\right) \leftrightarrow 12\angle 30° \quad \left(v = \frac{1}{4}\right)$$

and substitute for $z$:

$$z = e^{j\pi/4} = 1\angle 45° = \frac{1}{\sqrt{2}}(1 + j1)$$

The result is

$$\vec{Y}\left(1 - \frac{1}{2}e^{-j\pi/4}\right) = (12\angle 30°)\,(1 - e^{-j\pi/2})$$

$$\vec{Y}\left(1 - \frac{1}{2\sqrt{2}} + j\frac{1}{2\sqrt{2}}\right) = (12\angle 30°)(1 + j)$$

which we continue to systematically reduce to a polar-form number

$$\vec{Y} = \frac{(12\angle 30°)(\sqrt{2}\angle 45°)}{(0.6465 + j0.3536)} = \frac{16.97\angle 75°}{0.7369\angle 28.68°} = 23.03\angle 46.32°$$

Finally, we reverse the phasor transformation to obtain

$$y(n) = 23.03\cos\left(\frac{\pi n}{4} + 46.32°\right)$$

Phasor calculations in the $z$ domain are the same as those in the $s$ domain, but they involve an extra layer of calculations to replace the exponential terms with Euler's identity.

---

Example 8.4 is equivalent to the type of problem initially faced in an a-c circuits course; it helps reinforce the phasor concept. By now, however, we recognize that the system's transfer function gives all the important information and does not require us to state a phasor transformation or even specify details about the input signal.

The transfer function takes the form of a ratio of polynomials in $z$, and factoring the polynomials identifies the poles and zeros of the system in the $z$ plane. Furthermore, by letting $z = e^{j\pi v} = 1\angle\pi v$ we obtain the frequency response of the system. Note that as $v$ varies from 0 to 2, it takes us once around the unit circle of the $z$ plane and through the landscape created by the poles and zeros. As $v$ continues to increase, it just keeps making additional trips through the same landscape, showing again that the frequency response of the discrete-time system is periodic in the range $0 \le v < 2$. Since the frequency response is unique only in the range $0 \le v < 1$, it is particularly easy to have a computer make the calculations.

## EXAMPLE 8.5

Find the frequency response of the difference equation $y(n) = x(n) - x(n-2) + \frac{1}{2}y(n-1)$.

### Solution

To find the frequency response, we need the transfer function form of the difference equation. Substituting $x(n) = \vec{X}z^n, y(n) = \vec{Y}z^n$, and using the delay theorem gives

$$y(n) - \frac{1}{2}y(n-1) = x(n) - x(n-2)$$

$$\vec{Y}\left(1 - \frac{1}{2}z^{-1}\right) = \vec{X}(1 - z^{-2})$$

$$H(z) = \frac{\vec{Y}}{\vec{X}}(z) = \frac{(1 - z^{-2})}{\left(1 - \frac{1}{2}z^{-1}\right)} = \frac{z^2 - 1}{z(z - 0.5)}$$

The pole-zero diagram for the system is shown in Figure 8.3a. It shows that the frequency response starts and ends on zeros. The pole at the origin affects only the phase of the response, since it is equidistant from all points on the unit circle. The pole at $z = 0.5$ should produce a peaking in the frequency response somewhere between $v = 0$ and $v = 0.5$.

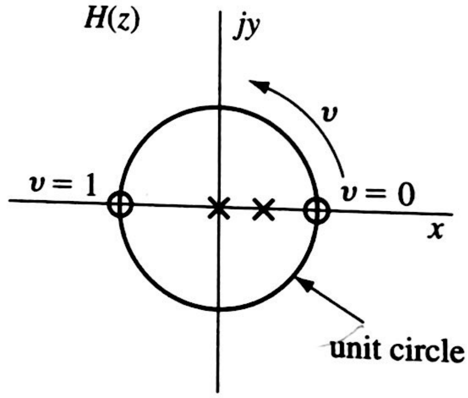
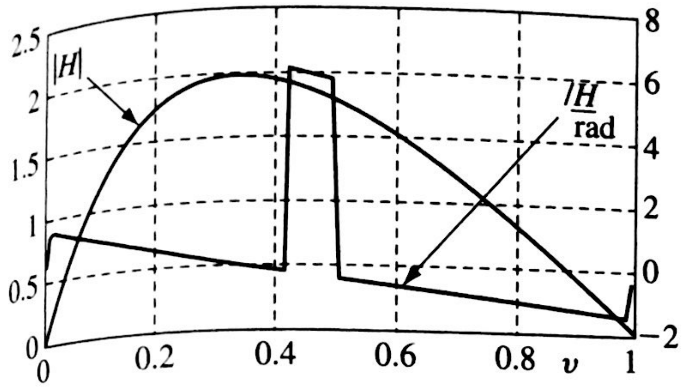



**Figure 8.3a** The pole-zero diagram for the system of Example 8.5.

If we want an expression for the frequency response in terms of $v$, we try to simplify:

$$H(e^{j\pi v}) = \frac{e^{j2\pi v} - 1}{e^{j\pi v}\,(e^{j\pi v} - 0.5)}$$

No special rules have been developed for sketching these types of functions, but we can always calculate the frequency response. We only need to provide the $H(z)$ polynomials. To a computer, calculating the frequency response is not much different in the $s$ or $z$ domain (Figure 8.3b).




```
>v=linspace(0.01,0.99);
>z=exp(j*pi*v);
>num=[1 0 -1];den=[1 -0.5 0];
>p1=polyval(num,z);
>p2=polyval(den,z);
>pha=angle(p1)-angle(p2);
>plot(v,abs(p1./p2))
>hold
>plot(v,unwrap(pha))
```

**Figure 8.3b** The frequency response is shown for the system of Example 8.5. The programming provided produces an unwrapped phase version of this response.

---

If we have a transfer function to implement or a difference equation whose frequency response is in question, the relationship between them is always the same; it is summarized here for convenience.

***Theorem***

The Difference Equation/Transfer Function Relationship

| **The *n* domain** | **The *z* domain** |
|---|---|

$$y(n) + \sum_{i=1}^{N} b_i\, y(n-i) = \sum_{i=0}^{M} a_i\, x(n-i) \Leftrightarrow \frac{\vec{Y}}{\vec{X}} = \frac{\sum_{i=0}^{M} a_i z^{-i}}{1 + \sum_{i=1}^{N} b_i z^{-i}}$$

As with a-c circuits, it is considered poor practice to mix domains. The $n$ and $z$ variables should never appear together except in the definition that relates the two domains, $x(n) = \vec{X} z^n$, or temporarily as the algebraic transition is being made from one domain to the other.

## 8.5 STABILITY IN THE *z* DOMAIN

The frequency response of a difference equation is meaningless if it has a natural response that is unstable. The numerical values generated by an unstable difference equation increase steadily until they exceed the capacity of the accumulator and registers. Any forcing function present is simply overwhelmed by the runaway natural response. After completing this section you will be able to:

- Determine the stability of a discrete-time system.
- Describe the natural response of a linear difference equation.

There are basically two ways to check the stability of a program. In the time domain, the simplest test is to apply a discrete unit impulse to the difference equation and see if the output dies out. The discrete unit impulse was defined as

$$\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases} \tag{7.16}$$

and since it forces the equation only when $n = 0$, it activates the system's natural response. During this test, it is unnecessary to retain delayed input terms in the difference equation, since only the feedback of the output terms affects stability.

## EXAMPLE 8.6

Determine if the following difference equation is stable:

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

### *Solution*

The iteration table is set up with only the $x(n)$ input term. The input terms are irrelevant in establishing stability. The test signal is $x(n) = \delta(n)$. It is necessary to continue the iterations either until a mathematical expression for the output is obtained or until some pattern is established that can be shown to continue indefinitely. In complicated difference equations, this may require an extensive iteration table.

| n | x(n) | $+\frac{3}{4}y(n-2)$ | $-y(n-1)$ | = | y(n) |
|---|---|---|---|---|---|
| 0 | 1 | 3/4(0) | 0 | | 1 |
| 1 | 0 | 3/4(0) | −(1) | | −1 |
| 2 | 0 | 3/4(1) | −(−1) | | 7/4 |
| 3 | 0 | 3/4(−1) | −(7/4) | | −10/4 |
| 4 | 0 | 3/4(7/4) | −(−10/4) | | 61/16 |

Stability ☆

In this case we see that for $n > 1$, the $y(n-2)$ and $y(n-1)$ terms are always the same sign and that the $y(n-1)$ term is always as large as the last output, so the equation is unstable. It has an oscillatory and ever-increasing natural response.

Even though we have used the simplest possible input and eliminated any delayed input terms, using an iteration table is still very prone to numerical and interpretation errors. A better approach is available in the $z$ domain. The denominator of a transfer function is the *characteristic equation* of the system, and its poles represent the natural $z$ values of the system. We will designate these poles as $z_p$. Turning on any input creates infinitesimal amounts of signal at all $z$ values. Those that receive infinite gain show up as the system's natural response. (If multiple poles occur, terms like $n^k(z_p)^n$ may also be generated.)

If the natural response terms take the form $y_n(n) = \vec{Y} z_p^n$, then the natural response terms will grow or decay with the sample number, $n$, according to the *magnitude* of the pole value,

$$|y_n(n)| \propto |z_p|^n$$

Reinforcing this concept is the fact that the locations of the $z$ plane poles are related to $s$ plane locations through the defining relationship

$$z = e^{sTs} = e^{(\sigma + jw)Ts} = e^{\sigma Ts} e^{j\omega Ts} = e^{\sigma Ts} \angle \pi v = |z_p| \angle \pi v \qquad \textbf{(8.8)}$$

from which it is seen that $|z| > 1$ corresponds to $\sigma > 0$, which is the right half of the $s$ plane (RHP). Allowed pole locations for stable systems are shown in Figure 8.4.

To summarize:

*If* $|z_p| > 1$: The natural response grows without bound. Such a system is *unstable* and it is not useful.

*If* $|z_p| = 1$: The natural response neither grows nor decays and is classified as *conditionally stable*. This condition corresponds to $\sigma = 0$ in the $s$ plane. (If such roots are repeated, however, the response grows without bound.)

*If* $|z_p| < 1$: The natural response is a transient response, and the system is *stable*. This condition corresponds to $\sigma < 0$ in the $s$ plane.

It is perhaps obvious, but if the difference equation does not include any delayed output terms, the characteristic equation reduces to unity. If the transfer function is

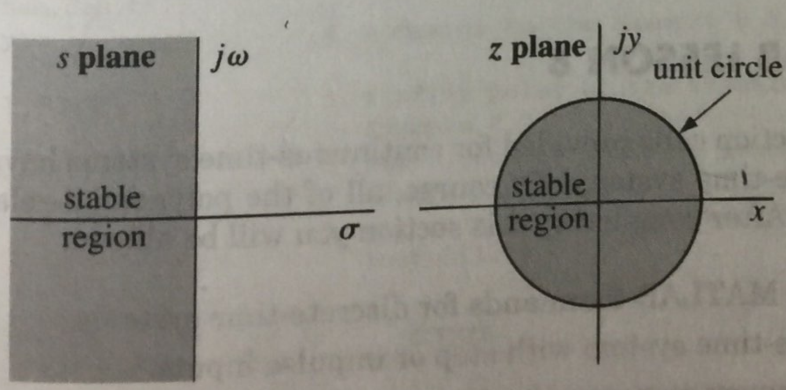



**Figure 8.4** For stability, roots of the characteristic equations for continuous-time systems and discrete-time systems must lie in the shaded regions of the $s$ and $z$ planes, respectively.

expressed in terms of polynomials in positive powers of $z$, all the poles are at $z = 0$, and there is no possibility of an unstable system. This is characteristic of FIR systems.

### EXAMPLE 8.7

Determine the natural response of a system whose difference equation is

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

***Solution***

We obtain the transfer function from the difference equation in the usual way.

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

$$\vec{Y}\left(1 + z^{-1} - \frac{3}{4}z^{-2}\right) = \vec{X}(1 - 2z^{-2})$$

$$\frac{\vec{Y}}{\vec{X}} = \frac{1 - 2z^{-2}}{1 + z^{-1} - \frac{3}{4}z^{-2}} = \frac{z^2 - 2}{z^2 + z - \frac{3}{4}} = \frac{z^2 - 2}{\left(z - \frac{1}{2}\right)\left(z + \frac{3}{2}\right)}$$

The poles indicate that the natural response will be

$$y_n(n) = K_1\left(\frac{1}{2}\right)^n + K_2\left(\frac{-3}{2}\right)^n$$

where $K_1$ and $K_2$ are arbitrary constants. The pole at $z = -3/2$ represents a growing oscillation that makes the system unstable.

## 8.6 MATLAB LESSON 8

Many of the function calls provided for continuous-time systems have their counterparts in discrete-time systems. Of course, all of the polynomial-related commands are still usable. After completing this section you will be able to:

- Identify basic MATLAB commands for discrete-time systems.
- Test a discrete-time system with step or impulse inputs.
- Find the frequency response of a discrete-time system.
- Find the output sample sequence, given the system and its input sequence.

Most commands for discrete-time systems expect the system polynomials to be in negative powers of $z$, such as $1 - z^{-1} + 2z^{-2} + 3z^{-3} + \cdots$, where the left-most entry must be the $z^0$ term. If the system numerator and denominator polynomials are of the same degree, it does not matter whether you interpret the polynomials as positive powers of $z$ or negative powers of $z$. If you seem to be getting strange results, use the help facility to see which way the polynomials are expected. If it still is not clear, add a leading or trailing zero to the polynomial vector. If the leading zero does not change the result, positive powers of $z$ are expected; if a trailing zero does not change the result, negative powers of $z$ are expected.

## MATLAB EXAMPLES

### dimpulse dstep

The simplest way to test the stability of a discrete-time system in the time domain is to use **dimpulse** to generate the iteration table. The discrete step function is another standard test signal. Positive powers of $z$ are expected.

```
>num=[1 0 -1];              % from Example 8.5, num = z^2 - 1 ?or? (1 - z^-2)

>den=[1 -.5];               % den = z - .5 ?or? 1 - .5z^-1
>dimpulse(num,den)          % system is noncausal as specified
>den=[1 -.5 0];             % den = z^2 - .5z + 0 ?or? 1 - .5z^-1
                              (no change)
>y=dimpulse(num,den);       % must need positive powers of z!
```

When MATLAB solves a difference equation, it delays the application of the input by a number of samples equal to the difference in the degrees of the denominator and numerator polynomials. Usually, the point of application of the impulse or step is obvious from the output results.

```
>de=[1 -.5 0 0 0 0 1];
>y=dimpulse(num,de,8)       % ask for the first eight outputs

>tt=0:5;
>y=dstep(num,den,tt);
>stem(tt,y)                 % compares to the Example 8.3 iteration table

>den2=[1 1 -3/4];           % finding poles of the transfer function from
                              Example 8.7
>r=roots(den2);             % roots expects positive powers of z, as it
                              always has
>abs(r)                     % any pole with a magnitude over 1 indicates
                              instability
```

### freqz

**freqz** expects polynomials in negative powers of $z$. Since it evaluates $H(z)$ along the unit circle ($z = e^{j\pi v}$), interpreting the polynomials incorrectly amounts to multiplying $H$ by some power of $z$, which affects the phase but not the magnitude of the result.

```
>freqz(num,den)                % plots frequency response curves in the z
                                 domain
>h=freqz(num,den);             % the default freqz uses a 512-point FFT
>v=(0:511)/512;                % allows other plotting options
>plot(v,abs(h))
>plot(v,180*angle(h)/pi)
>w=linspace(0,pi);             % calculation frequencies given in the
                                 normalized Ω variable
>h=freqz(num,den,w);
>plot(w,abs(h))
>help freqz                    % other options
```

### filter

The **filter** command performs the difference equation on a specified input number sequence. The output is the (file) filtered input. The **dimpulse** and **dstep** functions are often-used special cases. Polynomials in negative powers of $z$ are expected.

```
>x=[0 ones(1,10) zeros(1,30)];   % prepare a 10-sample-long square input
                                   pulse
>num=1;                          % make up a simple H(z)
>den=[1 -.2 .2];
>freqz(num,den)                  % see what the frequency response looks
                                   like
>y=filter(num,den,x);
>stem(y)                         % see what the output looks like in the n
                                   domain
>grid
```

## EXAMPLE 8.8

Plot the output $y(n)$ given that $x(n) = \sin(0.05\pi n)$ is passed through a filter with a transfer function of

$$H(z) = \frac{z^2 - 1}{z(z - 0.5)}$$

### Solution

Using 100 samples gives 2.5 cycles of the input, which should be enough to show steady-state conditions. The transfer function is the same one plotted in Figure 8.3b.

```
>n=0:99;               % prepare a 100-sample input signal
>x=sin(.05*pi*n);      % it will be a sinewave with v = 0.05
>num=[1 0 -1];         % define H(z)
```

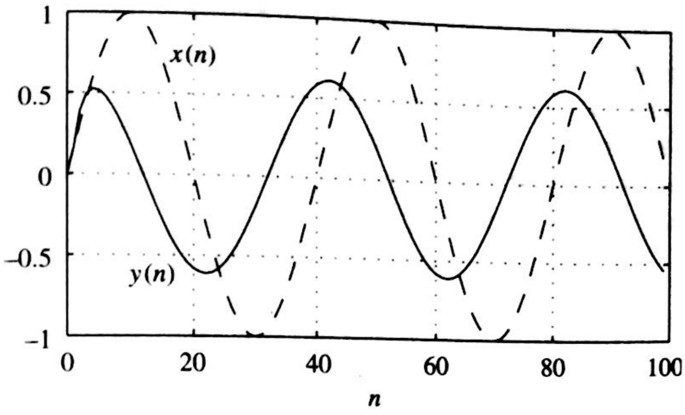



**Figure 8.5** Input and output samples for Example 8.8. A start-up transient is evident in the first 10 samples.

```
>den=[1 -.5 0];                    % H ≈ 0.6 ∠+1.3 rad @ v = 0.05 (Fig. 8.3b)
>y=filter(num,den,x);
>plot(n,y);
>grid
>hold
>plot(n,x,'--')
```

The result (Figure 8.5) shows that the output is about 0.6 times the amplitude of the input and that it leads the input by about eight samples. Since there are 40 samples per cycle, this represents a phase shift of about 1.26 radians.

## CHAPTER SUMMARY

The behavior of a discrete-time system is described in the form of a linear difference equation. This equation is also the program a digital processor executes. It describes how the current output is calculated from current and past inputs and possibly past outputs. If past outputs are included, the system has an infinite impulse response similar to continuous-time systems but may be unstable. If no past outputs are included, the system has a finite impulse response and is always stable. In principle, the output of a linear difference equation can be determined from the input using an iteration table.

The exponential signal takes the form $z^n$ in discrete-time systems. Assuming signals of this form and applying the delay theorem, we are able to convert a linear difference equation into a transfer function in the $z$ domain. We can represent the system with a pole-zero diagram, identify the poles with the system's natural response, and observe that any pole outside the unit circle represents an unstable natural response.

Evaluating the transfer function along the unit circle gives the frequency response of the system. If a system is found that has a desirable frequency response, we can transform back to the sample domain to find the difference equation that gives that response.

The development of discrete-time systems has closely paralleled that of continuous-time systems. In most respects, discrete-time systems are simpler, but phasor and frequency response calculations for them are more involved. In the absence of simple frequency response estimation techniques, we are even more reliant on computer tools for evaluating discrete-time systems. The computer, of course, is the main reason we are interested in discrete-time systems anyway.

## PROBLEMS

### Section 8.1

1. Express each of the following signals by their phasor representation $\vec{X}z^n$, where $\vec{X}$ is the Euler phasor. Identify $z$ in each case.

$$Ae^{\sigma T_s n}\cos(\omega T_s n + \theta) \leftrightarrow \vec{X} z^n$$

a. $2e^{0.1n}$ b. $4\cos(\pi n - 45°)$
c. $12e^{-n}\cos(0.2n + 14°)$ d. $6(1.2)^n \sin(3n - 25°)$

2. Find the signal whose phasor representation is given. Assume the phasor transformation is:

$$Ae^{\sigma T_s n}\cos(\omega T_s n + \theta) \leftrightarrow [Ae^{j\theta}]\, z^n$$

a. $12(-j)^n$ b. $2(0.2)^n$
c. $6\angle 30°(1 + j)^n$ d. $-j(2\angle 30°)^n$

### Section 8.2

3. A sampled signal is being nulled out by a difference equation implementing the equivalent of a notch filter. The signal is at a frequency of 2.5 kHz and is being sampled at 12 kHz. What signal frequency would be nulled if the sampling rate were reduced to 9 kHz?
4. A digital filter is designed for cutoff at $\upsilon = 0.15$. What is the frequency of the signal where cutoff occurs if the sampling rate is 8 kHz?

### Section 8.3

5. Use iteration to find the first five terms of the difference equation if $x(n)$ is the unit step function.

a. $y(n) - \dfrac{1}{2}y(n - 1) = x(n) - \dfrac{1}{2}x(n - 1)$

b. $y(n) + \dfrac{1}{2}y(n - 2) = x(n) - \dfrac{1}{2}x(n - 1)$

6. Use an iteration table to find the first five output samples from the system whose difference equation is $y(n) = x(n) - 0.2y(n-1) + 0.1y(n-2)$ and $x(n)$ is a unit impulse.

## Section 8.4

7. Find $H(z) = \vec{Y}/\vec{X}$ for the following difference equations.
   a. $y(n) = x(n) - 0.2y(n-1) + 0.1y(n-2)$
   b. $y(n) = x(n) - 0.5x(n-2) + 0.5y(n-2) - 0.5y(n-4)$
8. Find the transfer function represented by the following difference equations.
   a. $y(n) = x(n) + x(n-1) - 0.75y(n-1) + 0.5y(n-3)$
   b. $y(n) = 0.25x(n) - 0.5x(n-1) + x(n-2) - 0.5x(n-3) + 0.25y(n-4)$
9. Given that the input to the system of Problem 7a is $x(n) = 12\cos 0.2\pi n$, find the forced response for $y(n)$.
10. The input to a system with $H(z) = 1 - 0.5z^{-1} + 0.5z^{-2}$ is $x(n) = 2\cos 0.2\pi n$. Determine the forced response of the output.
11. The input to a system with $H(z) = 1 - z^{-4}$ is $x(n) = \cos 0.3\pi n$. Determine the forced response of the system.
12. Find the gain at d-c, $v = 0.5$, and $v = 1$ for the following transfer functions.
   a. $H(z) = \dfrac{z^2 + 1}{z^2 + 0.2z + 0.8}$
   b. $H(z) = 1 - z^{-1} + z^{-2}$
13. Determine the difference equation that implements each of the following transfer functions.
   a. $H(z) = \dfrac{z^2 - z + 2}{z^3 - 0.25z}$
   b. $H(z) = \dfrac{z^3 - 1}{z^3 - 0.2z^2 + 0.2z - 1}$
14. Determine the difference equation for each of the following systems.
   a. $H(z) = \dfrac{z^4 + 4z^3 - 2z^2 + z - 1}{z^4}$
   b. $H(z) = \dfrac{z^4 - 1}{z^4 + 0.5}$
15. What problems, if any, can you see in implementing the following transfer function?

$$H(z) = \frac{z^4 - 1}{z^3 + 0.5z - 0.25}$$

## Section 8.5

16. Test the stability of the system in Problem 7b using an iteration table. Follow the output until the result is clearly established.

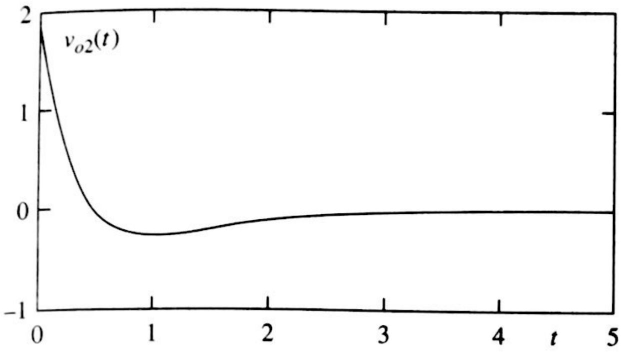
17. Test the stability of the system in Problem 8b using an iteration table.
18. Test the stability of each system in Problem 14 by finding the location of its poles in the $z$ plane.
19. Test the indicated difference equations for stability by locating the poles of each transfer function.
    a. $y(n) = 0.14x(n) + 0.14x(n - 1) + 1.02y(n - 1)$
    b. $y(n) = 0.5x(n) - 0.3x(n - 2) - 2y(n - 1) - y(n - 2)$

## *Section 8.6*

20. Use the **dstep** command to find the first 12 outputs from the systems of Problem 5.
21. Plot the magnitude of the gain versus frequency for the difference equations indicated. Use a linear scale for $|H|$.
    a. $y(n) = 0.1373x(n) + 0.1373x(n - 1) + 0.7254y(n - 1)$
    b. $y(n) = 0.0301x(n) - 0.0301x(n - 2) + 1.8439y(n - 1) - 0.9398y(n - 2)$
    c. $y(n) = 0.9288x(n) - 1.7682x(n - 1) + 0.9288x(n - 2) + 1.7682y(n - 1) - 0.8576y(n - 2)$
    d. $y(n) = 0.8627x(n) - 0.8627x(n - 1) + 0.7254y(n - 1)$
22. Test the stability of the systems in Problem 19 using the **dimpulse** command.
23. An input signal $x(t) = \sin 250\pi t + \cos 1000\pi t + \cos 3000\pi t$ is sampled at $t = n/6000$. It is filtered with a "comb filter" whose difference equation is $2y(n) = x(n) + x(n - 6)$. Use about 100 samples.
    a. Plot $x(n)$ vs. $n$.
    b. Plot the magnitude of the filter transfer function for $0 \leq v < 1$.
    c. Plot $y(n)$ vs. $n$.
    d. Explain the results of (c) based on the transfer function and the known input.
24. Repeat Problem 23 for $x(t) = \cos 1000\pi t - \sin 3000\pi t + \cos 11500\pi t$. Explain the results.
25. Create a 300-sample signal consisting of $x = x1 + x2$, where $x1$ is a low-frequency square wave and $x2$ represents high-frequency noise.

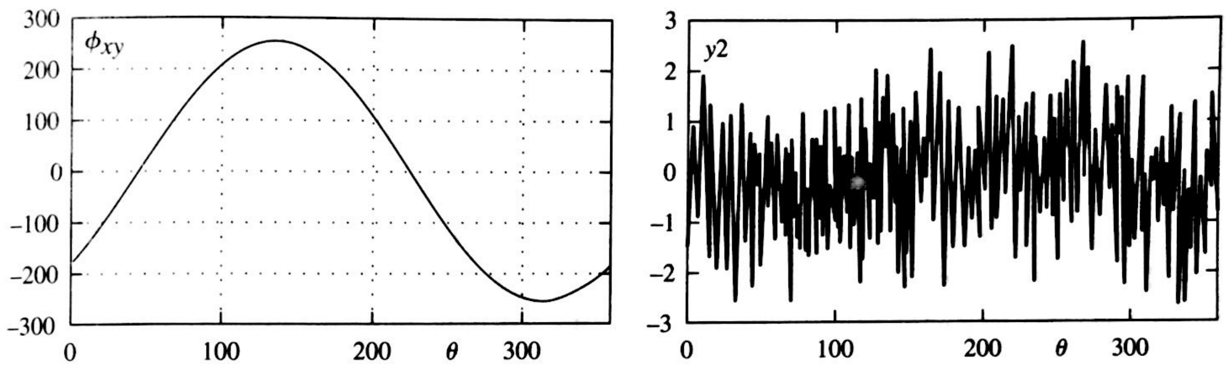
$$x1 = [\text{ones}(1,75)\ \ -1*\text{ones}(1,75)\ \ \text{ones}(1,75)\ \ -1*\text{ones}(1,75)]$$
$$x2 = 2*\cos(0.7*\pi*n) + 3*\cos(0.9*\pi*n)$$

    a. Plot $x$ vs. $n$.
    b. Filter the signal with the low-pass filter difference equation of Problem 21a and plot the output vs. $n$.
    c. Put the output of the first pass through the filter again. Plot the new output waveform. (A sinusoid is a poor test signal for a filter, because the foregoing process can be repeated over and over and the sinusoid will eventually emerge free of noise. Any other waveform, such as the square wave of this problem, is degraded each time it passes through the filter.)




b) Can't use *impulse* when the denominator is same order as numerator. What sampling frequency will avoid aliasing in a high-pass filter? Instead, do one step of synthetic division on the transfer function.
$v_{o2}(t) = v_i(t) - v_{o1}(t)$

7.33 a)




b)
```
> n=0:511;x=sin(2*pi*n/512); y1=sin(2*pi*n/512-3*pi/4); ang=n*360/512;
> X=fft(x);
> y2=0.3*y1+randn(1,512); plot(ang,y2)      % no way can phase be measured
> Y2=fft(y2);G=Y2.*conj(X); cor=ifft(G); plot(ang, real(cor));
> [a,b]=max(cor)
a = 86.0297    0.0000i
b = 195
> phase=195*360/512 = 137.1094 % off by 2.1°
```

This result will vary since *randn* gives different values each time it is called. Repeating the programming sequence many times and averaging the results will give a best estimate.

7.35 $\sum_{m=0}^{N-1} e^{j2\pi nm/N} = \dfrac{1-e^{j2\pi q}}{1-e^{2\pi q/N}}$ where $0 \le q \le N-1$

$\dfrac{1-e^{j2\pi q}}{1-e^{j2\pi q/N}} = \dfrac{1-1}{1-e^{j2\pi q/N}} = 0$ if $q \neq 0$ and $\sum_{m=0}^{N-1} e^{j2\pi qm/N} = \sum_{m=0}^{N-1} 1^m = N$ if $q = 0$

## Chapter 8

8.1 a) $\tilde{X} = 1; z = e^{0.1} = 1.1052$
b) $\tilde{X} = 2\angle -45°; z = e^{j\pi} = -1$
c) $\tilde{X} = 6\angle 14°; z = e^{-1}e^{j0.2}$
d) $\tilde{X} = 3\angle -115°; z = 1.2e^{j3}$

8.3 $v = 2f/f_s = 5/12 = 2f'/9$ kHz; $f' = 1.875$ kHz is the new notch frequency.

| $n$ | $x(n)$ | $-x(n-1)/2$ | $y(n-1)/2$ | $y(n)$ |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | −1/2 | 1/2 | 1 |
| 2 | 1 | −1/2 | 1/2 | 1 |
| 3 | 1 | −1/2 | 1/2 | 1 |
| 4 | 1 | −1/2 | 1/2 | 1 |
| 5 | 1 | −1/2 | 1/2 | 1 |

| $n$ | $x(n)$ | $-x(n-1)/2$ | $-y(n-2)/2$ | $y(n)$ |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | −1/2 | 0 | 1/2 |
| 2 | 1 | −1/2 | −1/2 | 0 |
| 3 | 1 | −1/2 | −1/4 | 1/4 |
| 4 | 1 | −1/2 | 0 | 1/2 |
| 5 | 1 | −1/2 | −1/8 | 3/8 |

8.7 a) $H(z) = \dfrac{z^2}{z^2 + 0.2z - 0.1}$ b) $H(z) = \dfrac{z^2(z^2 - 0.5)}{z^4 - 0.5z^2 + 0.5}$

8.9 $H(e^{j\pi 0.2}) = 0.8840\angle 0.0198$ $y(n) = 10.61\cos(0.2\pi n + 0.0198)$

8.11 $H(e^{j\pi 0.3}) = 1 - e^{-j1.2\pi} = 1.902\angle - 18.0°$ $y(n) = 1.902\cos(0.3\pi n - 18.0°)$

8.13 a) $y(n) = x(n-1) - x(n-2) + 2x(n-3) + 0.25y(n-2)$

b) $y(n) = x(n) - x(n-3) + 0.2y(n-1) - 0.2y(n-2) + y(n-3)$

8.15 The system is noncausal, since the next input, $x(n+1)$, is needed to calculate the current $y(n)$.

8.17

| $n$ | $0.25\,x(n)$ | $+0.25y(n-4)$ | $y(n)$ |
|---|---|---|---|
| 0 | 0.25 | 0 | 0.25 |
| 1 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 |
| 4 | 0 | $(.25)^2$ | $(.25)^2$ |
| 8 | 0 | $(.25)^3$ | $(.25)^3$ |
| 12 | 0 | $(.25)^4$ | $(.25)^4$ |

Stable: $y(4m) = 0.25(1/4)^m$, $m$ integers, all other, $y = 0$.

8.19 a) Poles at $z = 1.02$, ∴ unstable. b) Repeated poles at $z = -1$, ∴ unstable.

8.21 `>h=freqz(num,den); v=(0:511)/512; plot(v,abs(h))`

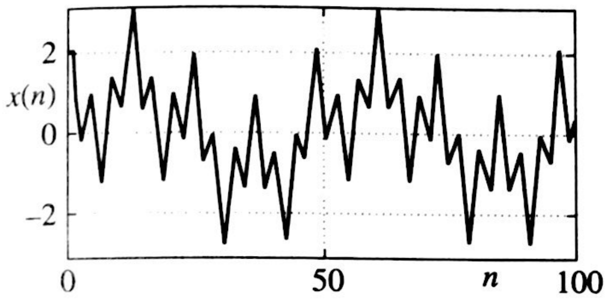
a) num = [0.1373 0.1373]; den = [1 −0.7254];
Low-pass filter with cutoff @ $v = 0.1$

b) num = [0.0301 0 −0.0301]; den = [1 −1.8439 0.9398];
Bandpass filter with $v_c = 0.1$

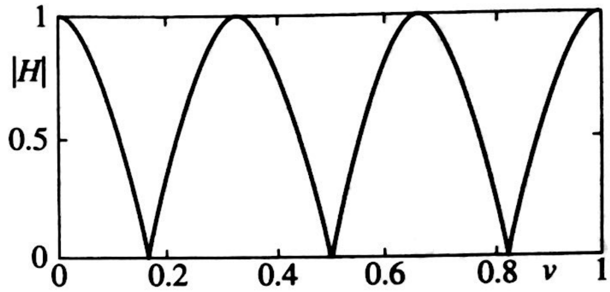
c) num = [0.9288 −1.7682 0.9288]; den = [1 −1.7682 0.8576];
Bandstop filter with $v_c = 0.1$

d) num = [0.8627 $-0.8627$]; den = [1 $-0.7254$];
High-pass filter with cutoff @ $v = 0.1$

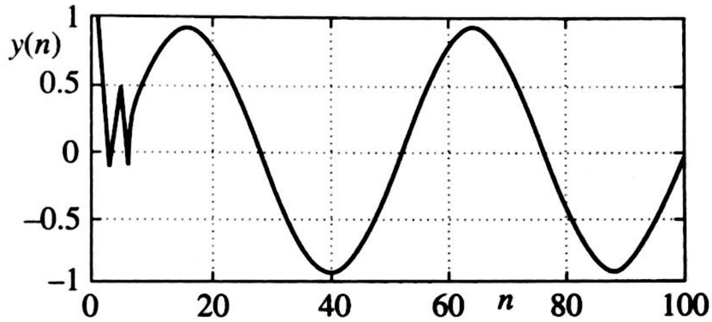
8.23 a)




b)

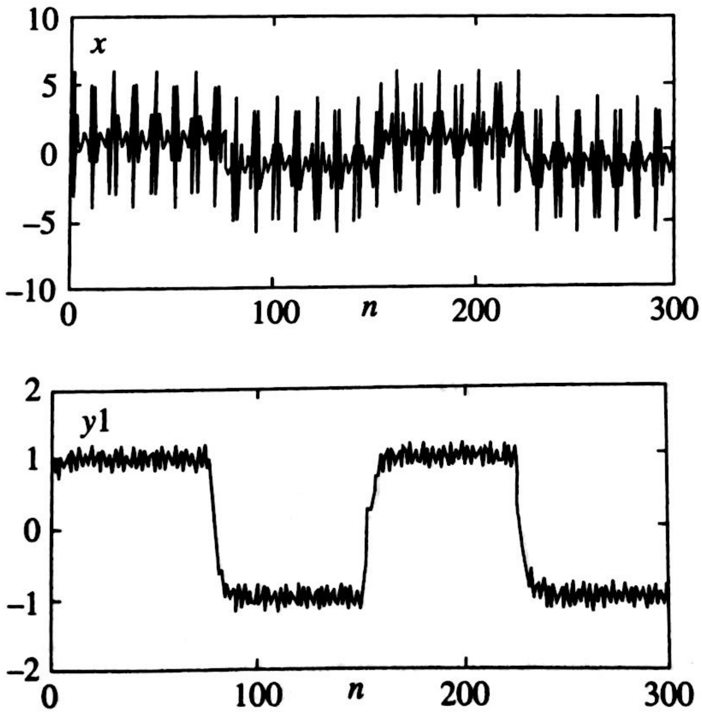


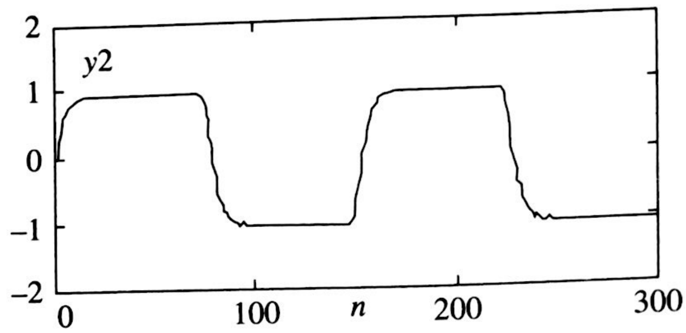

c)




d) Input consists of equal-sized components at $v = 0.042, 0.167$, and $0.50$. The components at 0.167 and 0.5 are not passed by the filter, so only the lowest-frequency component emerges. Output freq. = 0.042(3000) = 126 Hz. (Check: T ≈ 48 samples; $f$ = 6000/48 = 125 Hz)

8.25






## Chapter 9

9.1 $H(z) = \dfrac{1}{\left[C\dfrac{z-1}{z+1}\right]+1} = \dfrac{z+1}{(C+1)z-(C-1)} =$

$$\frac{z+1}{[1+\cot(\pi v_{co}/2)]z+[1-\cot(\pi v_{co}/2)]}$$

9.3 $H_p(s) = \dfrac{1.4314}{s^2+1.4256s+1.5162}$, $C = 1.6319$, $H(z) = \dfrac{0.2200(z^2+2z+1)}{z^2-0.3525z+0.2848}$

9.5 $H_p(p) = \dfrac{1}{p^2+1.4142p+1}$, $H_{HP}(s) = \dfrac{s^2}{s^2+1.4142s+1}$, $C = 1$,

$$H(z) = \frac{0.2929(z^2-2z+1)}{z^2+0.1716}$$

9.7 $C = 1.5158, B = 0.3446C = 0.5224$

9.9 $C = 3.5200, B = 1.2359, H_{BS}(z) = \dfrac{0.8981-1.5279z^{-1}+0.8981z^{-2}}{1-1.5279z^{-1}+0.7961z^{-2}}$

9.11 a)