Discrete-Time Signals and Systems (Recap)

Richard Heusdens

EE2S31

TU Delft

Discrete-Time Signals

A discrete-time signal x is a function of an integer variable. The signal is *not defined* at instants between two successive samples.

1. Functional representation, such as

$$x(n) = \begin{cases} 1, & \text{for } n = 1, 3 \\ 4, & \text{for } n = 2 \\ 0, & \text{otherwise} \end{cases}$$

2. Sequence representation, such as

$$x(n) = \{\cdots 0, 0, \underset{\uparrow}{0}, 1, 4, 1, 0, 0, \cdots\},$$

where the symbol $\uparrow$ indicates the time origin.

Discrete-Time Signals

Some elementary discrete-time signals:

1. The *unit sample sequence* or *unit impulse* is denoted by $\delta(n)$ and is defined as

$$\delta(n) = \begin{cases} 1, & \text{for } n = 0 \\ 0, & \text{for } n \neq 0 \end{cases}$$

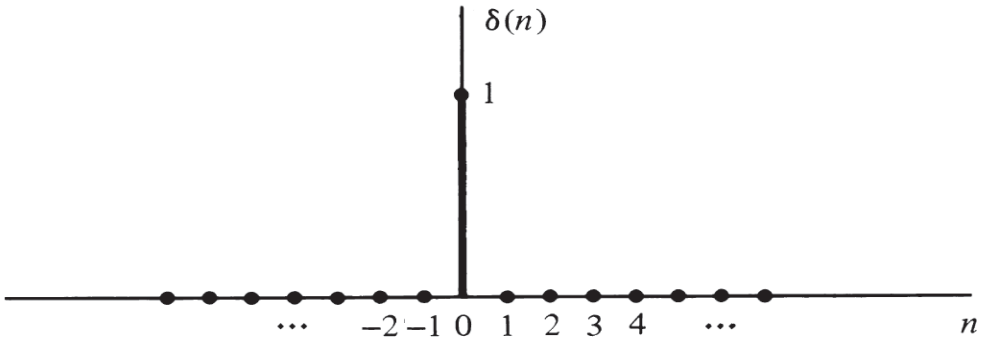


Figure 2.1.2 Graphical representation of the unit sample signal.

Discrete-Time Signals

2. The *unit step function* is denoted by $u(n)$ and is defined as

$$u(n) = \begin{cases} 1, & \text{for } n \geq 0 \\ 0, & \text{for } n < 0 \end{cases}$$

3. The *exponential signal* is a sequence of the form

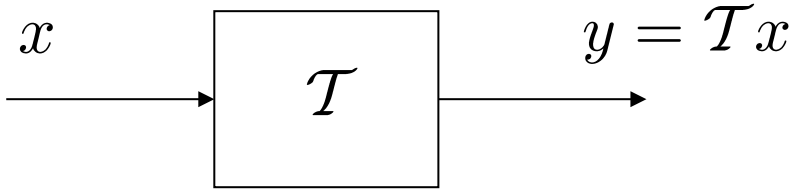
$$x(n) = a^n \quad \text{for all } n$$

When a is complex, it can be expressed as $a = re^{j\theta}$, $r, \theta \in \mathbb{R}$.

4. .....

Discrete-Time Systems

A *discrete-time system* is a device or algorithm that operates on a discrete-time signal, called the *input* or *excitation*, according to some well-defined rule, to produce another discrete-time signal called the *output* or *response* of the system.



Example:

$$y(n) = x(n) + x(n-1) + x(n-2)$$

$$y(n) = 2y(n-1) - 3x(n) + 4x(n-1)$$

$$y(n) = x^2(n)$$

Discrete-Time Systems

Classifiction of discrete-time systems:

- Time-invariant versus time-variant systems
- Linear versus non-linear systems

Let D denote the time-delay operator, so that $\left(D^k x\right)(n) = x(n-k)$.

Definition: A system $\mathcal{T}$ is *time invariant* or *shift invariant* if and only if

$$y = \mathcal{T} x$$

implies

$$\mathcal{T} D^k x = D^k \mathcal{T} x = D^k y$$

Discrete-Time Systems

Definition: A system is linear if and only if

$$\mathcal{T}\left(a_1 x_1 + a_2 x_2\right) = a_1 \mathcal{T} x_1 + a_2 \mathcal{T} x_2$$

for any arbitrary input sequence x_1 and x_2, and arbitrary constants a_1 and a_2.

A linear system is one that satisfies the *superposition principle*.

Discrete-Time Systems

Definition: (LTI) A system is said to be *linear time-invariant (LTI)* if it is linear and time/shift-invariant

Other classifications of systems:

- Static (or memoryless) versus dynamic systems
- Causal versus non-causal systems
- Stable versus non-stable systems

Impulse Response

Impulse response:

$$h \stackrel{\text{def}}{=} \mathcal{T}\delta$$

Express x as a linear combination of weighted (time-delayed) impulses, that is

$$x = \sum_{k=-\infty}^{\infty} x(k) D^k \delta$$

or, equivalently,

$$x(n) = \sum_{k=-\infty}^{\infty} x(k)\delta(n-k)$$

Convolution Sum

If the system is linear and time invariant (LTI), we have

$$y = \mathcal{T}\left(\sum_{k=-\infty}^{\infty} x(k) D^k \delta\right)$$

$$\overset{\text{(linear)}}{=} \sum_{k=-\infty}^{\infty} x(k) \mathcal{T} D^k \delta)$$

$$\overset{\text{(time-invariant)}}{=} \sum_{k=-\infty}^{\infty} x(k) D^k h$$

so that $\boxed{y(n) = \sum_{k=-\infty}^{\infty} x(k) h(n-k)}$

convolution sum

$$y(n) = (x * h)(n)$$

Causal LTI Systems

A system is called *causal* if the output at time $n = n_0$ depends only on present and past input samples, that is, on values of $x(n)$ for $n \leq n_0$.

From

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

we conclude that an LTI system is causal if and only if its impulse response is zero for negative values of n,

$$h(n) = 0 \text{ for } n < 0$$

Stability of LTI Systems

Definition: An arbitrary relaxed system is said to be *bounded input-bounded output* (BIBO) stable if and only if every bounded input produces a bounded output. That is, there exist constants M_x, M_y such that

$$|x(n)| \leq M_x < \infty \quad \Rightarrow \quad |y(n)| \leq M_y < \infty,$$

for all n. If, for some bounded input sequence, the output is unbounded (infinite), the system is classified as unstable.

Stability of LTI Systems

An LTI system is BIBO stable if for every bounded input ($|x(n)| \leq M_x < \infty$) we have

$$|y(n)| = \left| \sum_{k=-\infty}^{\infty} h(k)x(n-k) \right|$$

$$\leq M_x \sum_{k=-\infty}^{\infty} |h(k)| < \infty$$

from which we conclude that $\sum_k |h(k)| < \infty$.

That is, we have BIBO stability if h is absolutely summable. (and only if)

Finite Impulse Response Systems

If the impulse response of a system has finite support (is of finite duration), the response is said to be a *finite impulse response* (FIR). Without loss of generality, we focus on causal FIR systems, so that $h(n) = 0$ for $n < 0$ and $n > M$.

FIR filters are necessarily stable! $\sum_{n=-\infty}^{\infty} |h(n)| = \sum_{n=0}^{M} |h(n)| < \infty$

The convolution sum for such a system reduces to

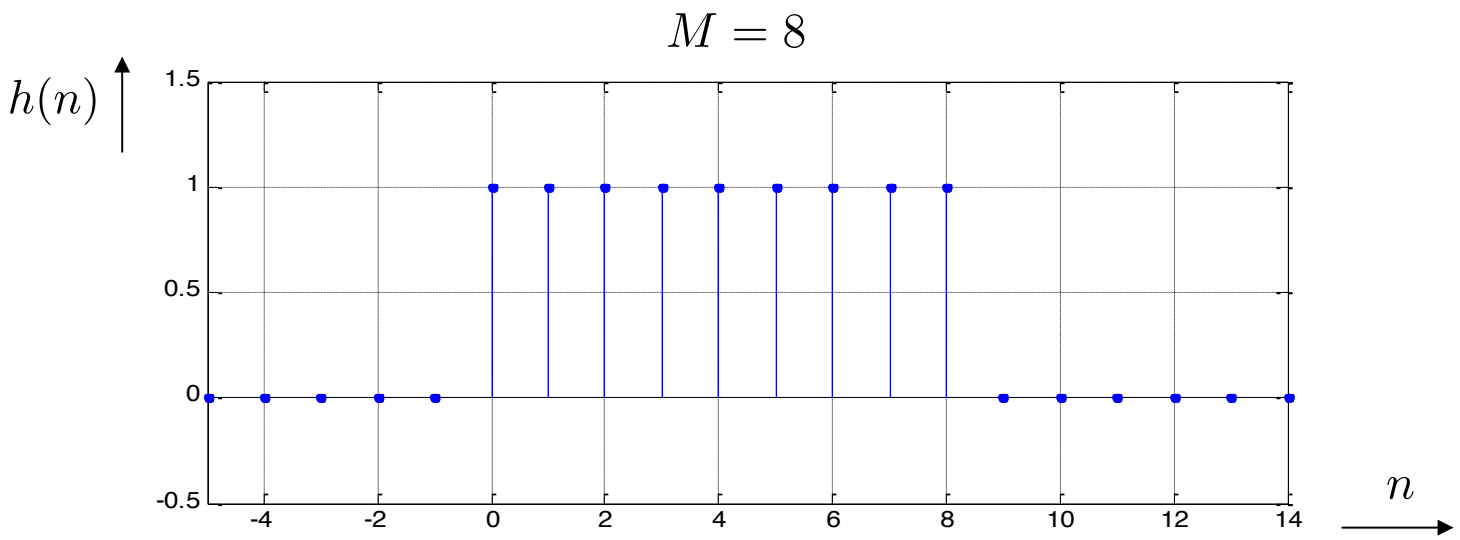
$$y(n) = \sum_{k=0}^{M} h(k)x(n-k)$$

The output at any time n is simply a weighted linear combination of the past input samples $x(n), \ldots, x(n-M)$.

Finite Impulse Response Systems

Example:

$$h(n) = \begin{cases} 1, & 0 \leq n \leq M \\ 0, & \text{otherwise} \end{cases}$$

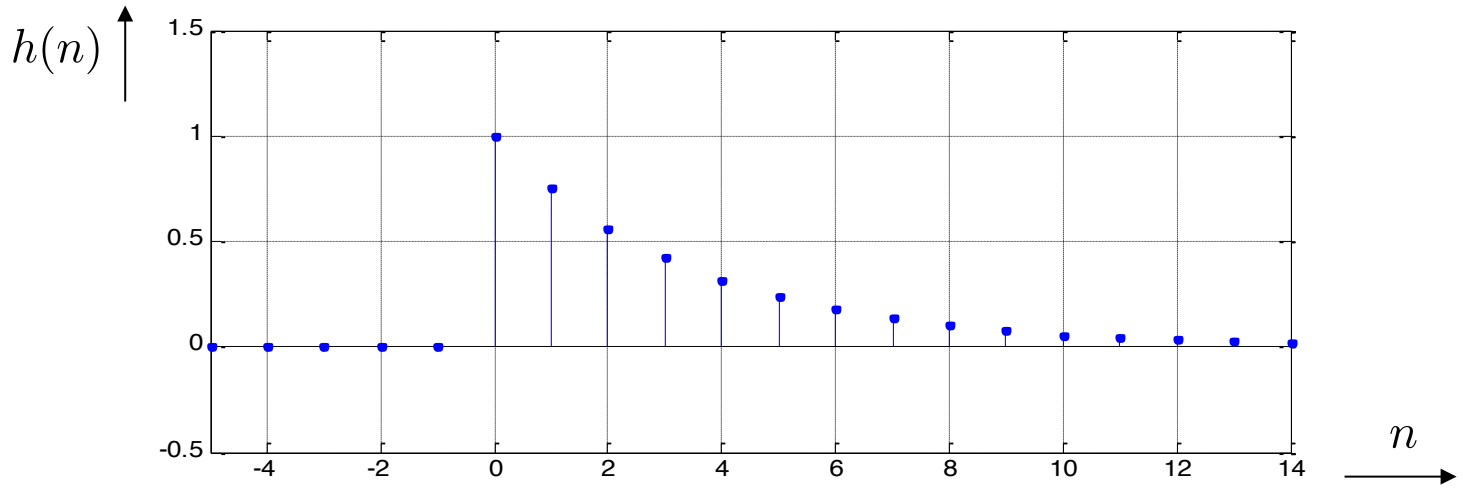


Infinite Impulse Response Systems

If the impulse response of a system has infinite support (is of infinite duration), the response is said to be an *infinite impulse response* (IIR).

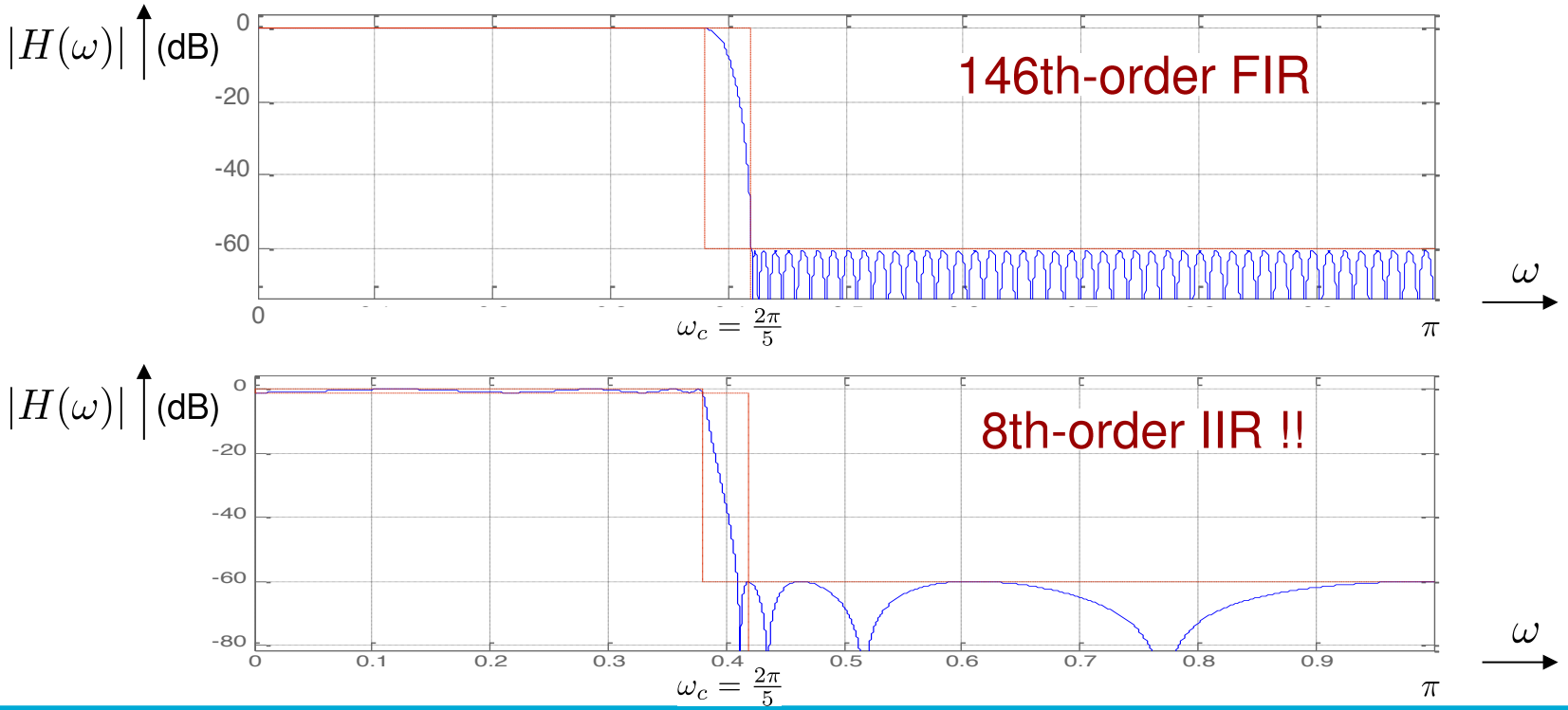
Example:

$$h(n) = \left(\frac{3}{4}\right)^n u(n)$$



FIR vs. IIR Filters

Example: filter design ($\omega_c = \frac{2\pi}{5}$, transition width 0.04π, stopband attentuation 60 dB)



Infinite Impulse Response Systems

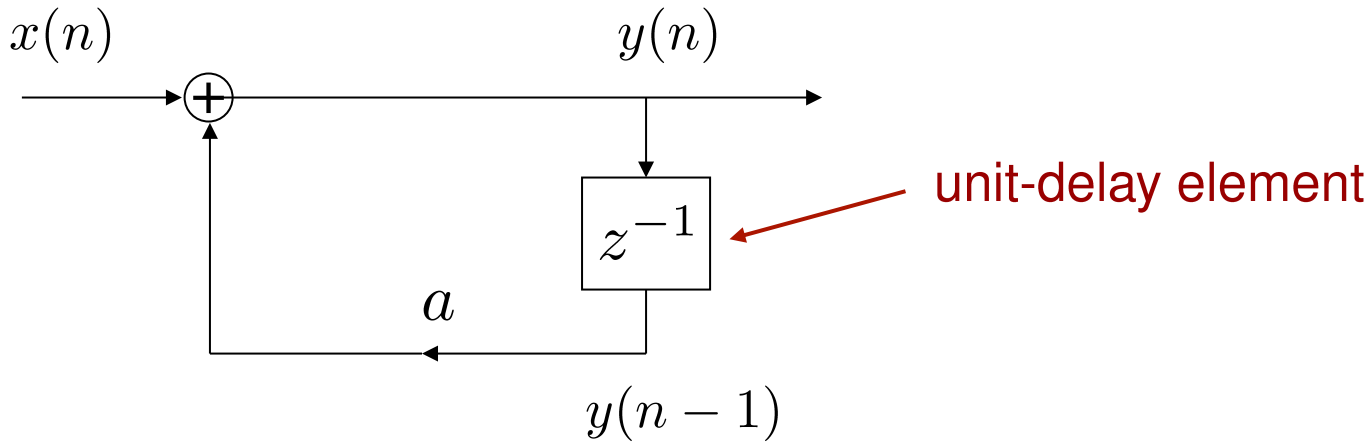
Remarks:

- In the FIR case, the convolution sum suggests a means for the realization of the system, which involves additions, multiplications and a finite number of memory (delay) elements.
- In the IIR case, however, its practical implementation as implied by the convolution sum is clearly impossible.
- However, a subclass of IIR filters can be implemented efficiently other than in the form suggested by the convolution sum. This class is more conveniently described by difference equations.

Difference Equations

Example: first-order recursive system

$$y(n) = ay(n-1) + x(n)$$



Difference Equations

Compute successive values of $y(n)$ for $n \geq 0$:

$$
\begin{aligned}
y(0) &= ay(-1) + x(0) \\
y(1) &= ay(0) + x(1) = a^2 y(-1) + ax(0) + x(1) \\
y(2) &= ay(1) + x(2) = a^3 y(-1) + a^2 x(0) + ax(1) + x(2) \\
&\vdots \\
y(n) &= a^{n+1} y(-1) + \sum_{k=0}^{n} a^k x(n-k)
\end{aligned}
$$

Zero-Input and Zero-State Response

$$y(n) = \underbrace{a^{n+1}y(-1)}_{y_{\mathrm{zi}}(n)} + \underbrace{\sum_{k=0}^{n} a^k x(n-k)}_{y_{\mathrm{zs}}(n)}, \quad n \geq 0$$

Two contributions:

- *Zero-input response* y_{zi} is the output of the system when $x(n) = 0$ for all n.
- *Zero-state response* y_{zs} is the response of the system when it starts with zero initial conditions

A system is said to be *relaxed* when $y_{\mathrm{zi}}(n) = 0$ for all n.

Steady-State and Transient Response

Example: $x(n) = u(n)$

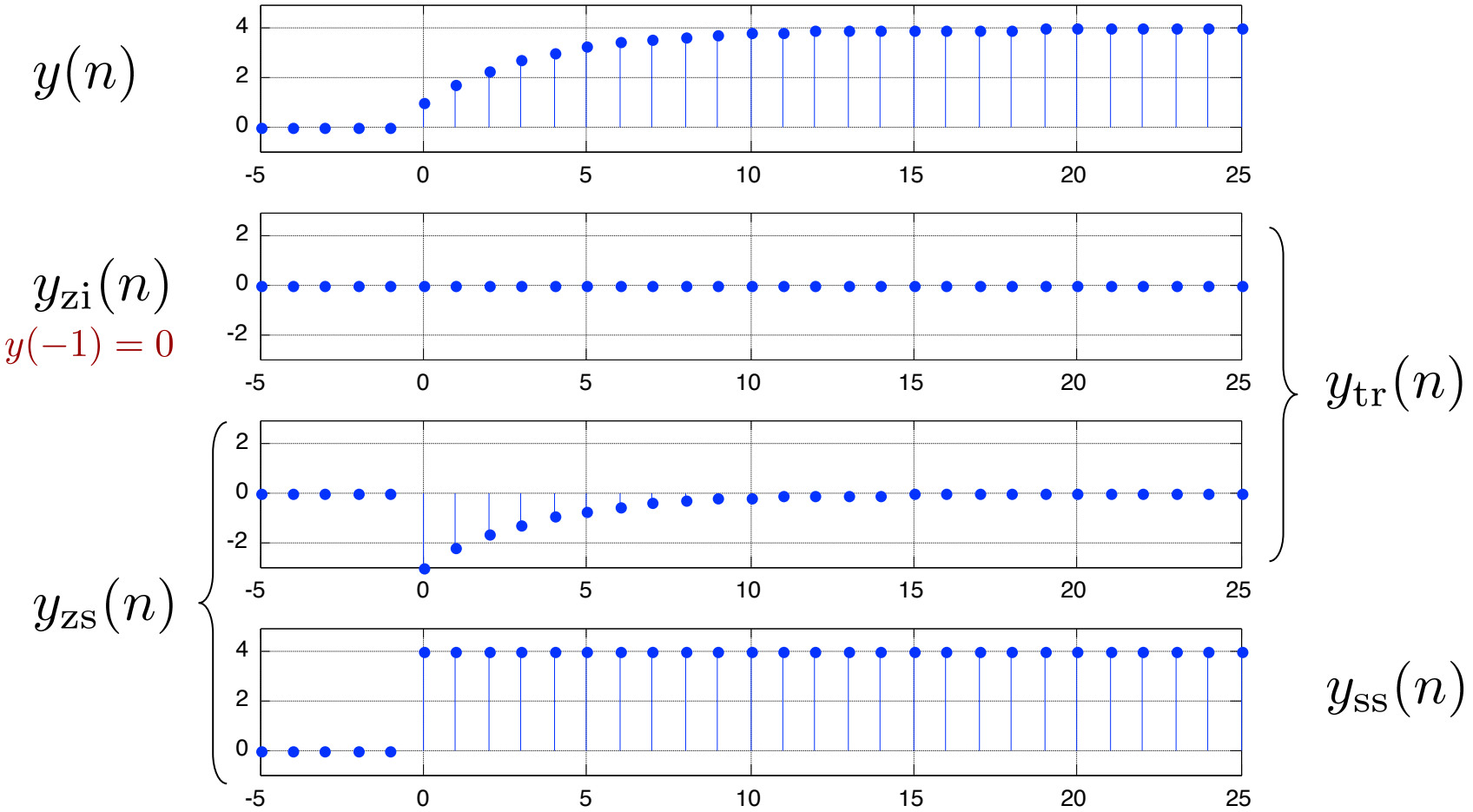
$$y(n) = \underbrace{a^{n+1}y(-1)}_{y_{\rm zi}(n)} + \underbrace{\frac{1-a^{n+1}}{1-a}}_{y_{\rm zs}(n)} = \underbrace{a^{n+1}y(-1) - \frac{a^{n+1}}{1-a}}_{y_{\rm tr}(n)} + \underbrace{\frac{1}{1-a}}_{y_{\rm ss}(n)}$$

- The *steady-state response*, denoted by $y_{\rm ss}$, is defined by

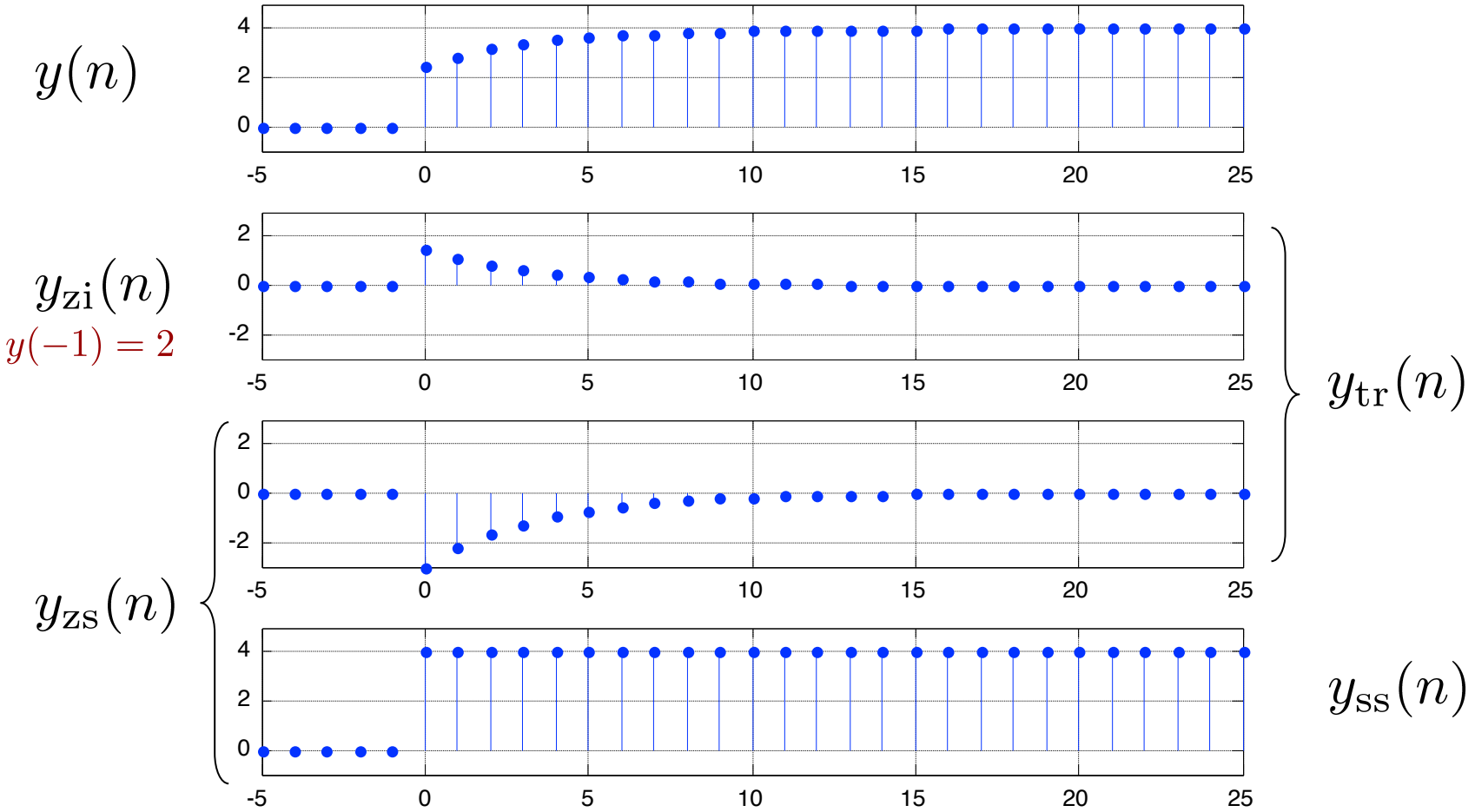
$$y_{\rm ss}(n) = \lim_{n\to\infty} y(n) = \frac{1}{1-a}$$

- The part of the response that dies out as n approaches infinity is called the *transient response*, denoted by $y_{\rm tr}$

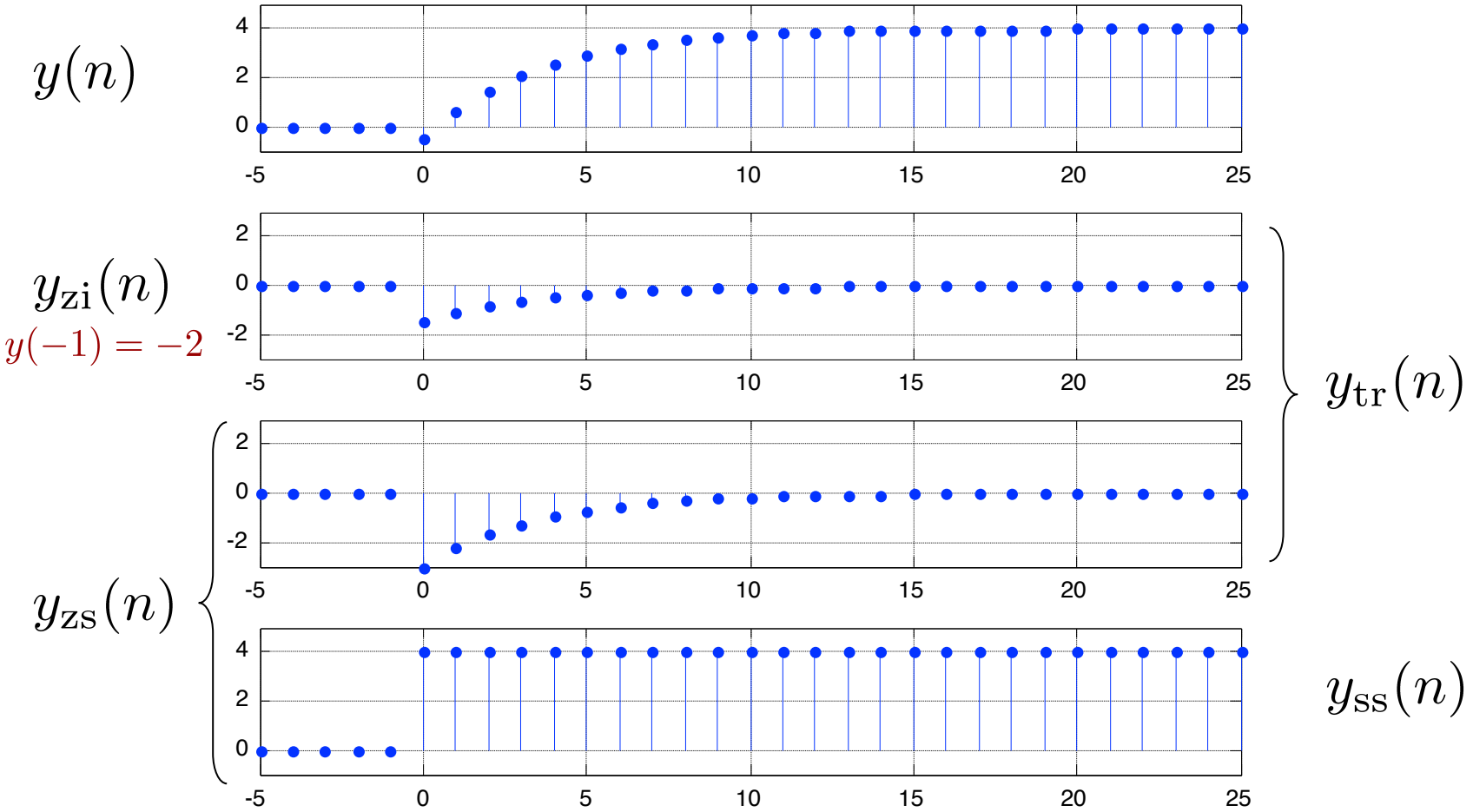
First-Order Recursive System (a=0.75)



First-Order Recursive System (a=0.75)



First-Order Recursive System (a=0.75)



General Nth-order Recursive Systems

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

- In order to determine $y(n)$ for $n \geq 0$, we need the input $x(n)$ for all $n \geq 0$ and the initial conditions $y(-1), \ldots, y(-N)$.
- The initial conditions summarize all that we need to know about the past history of the response to compute the present and future outputs.
- The integer N is called the *order* of the difference equation or the order of the system.
- When the coefficients a_k and b_k are constants (do not depend on n), the recursive system is linear and time-invariant

General Solution

Linear constant-coefficient nonhomogeneous difference equation:

$$\sum_{k=0}^{N} a_k y(n-k) + \sum_{k=0}^{M} b_k x(n-k) = 0, \qquad a_0 = -1$$

Solution is the sum of two parts:

$$y(n) = y_{\mathrm{h}}(n) + y_{\mathrm{p}}(n)$$

where y_{h} is called the *homogeneous* or *complementary* solution, whereas $y_{\mathrm{p}}(n)$ is called a *particular* solution.