Chapter 2

Discrete-time signals and systems

Contents

Overview

- terminology, classes of signals and systems, *linearity*, *time-invariance*. impulse response, convolution, difference equations, correlation, analysis ...

Much of this chapter parallels 306 for CT signals.

Goal: eventually DSP system design; must first learn to *analyze*!

2.1

Discrete-time signals

Our focus: single-channel, continuous-valued signals, namely 1D discrete-time signals $x[n]$.
In mathematical notation we write $x : \mathbb{Z} \to \mathbb{R}$ or $x : \mathbb{Z} \to \mathbb{C}$

- $x[n]$ can be represented graphically by "stem" plot.
- $x[n]$ is *not defined* for noninteger n. (It is not "zero" despite appearance of stem plot.)
- We call $x[n]$ the n**th sample** of the signal.

We will also consider 2D discrete-space images $x[n, m]$.

2.1.1

Some elementary discrete-time signals (important examples)

- **unit sample sequence** or **unit impulse** or **Kronecker delta function** (much simpler than the **Dirac impulse**)

$$\text{Centered: } \delta[n] = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases} \quad \text{Shifted: } \delta[n-k] = \begin{cases} 1, & n = k \\ 0, & n \neq k \end{cases} \quad \textbf{\textit{Picture}}$$

- **unit step signal**

$$u[n] = \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases} = \{\ldots, 0, 0, \underline{1}, 1, \ldots\}$$

  Useful relationship: $\delta[n] = u[n] - u[n-1]$. This is the discrete-time analog of the continuous-time property of Dirac impulses: $\delta(t) = \frac{\mathrm{d}}{\mathrm{d}t} u(t)$.

- **exponential signal** or **geometric progression** (discrete-time analog of continuous-time e^{at})

$$x[n] = a^n \quad \textbf{\textit{plot for }} 0 < a < 1 \textbf{\textit{ real. See text for other cases.}}$$

The **2D Kronecker impulse**:

$$\delta_{\mathrm{2D}}[n, m] = \delta[n]\,\delta[m] = \begin{cases} 1, & n = 0, m = 0 \\ 0, & \text{otherwise.} \end{cases}$$

Signal notation

There are several ways to represent discrete-time signals. One way is graphically. Here are five (!) others.

$$x[n] = \{\ldots, 0, 0, \underline{2}, 1, 1, \ldots\} = u[n] + \delta[n] = 2\,\delta[n] + \delta[n-1] + \delta[n-2] + \cdots = 2\,\delta[n] + \sum_{k=1}^{\infty} \delta[n-k] = \begin{cases} 2, & n = 0, \\ 1, & n \geq 1, \\ 0, & n < 0. \end{cases}$$

For a 4-periodic signal we may write $\{\underline{1}, 0, 7, 5\}_4$ to denote the signal $\{\ldots, \underline{1}, 0, 7, 5, 1, 0, 7, 5, 1, 0, \ldots\}$.

Skill: ***Convert between different discrete-time signal representations.***
Skill: ***Choose representation most appropriate for a given problem.*** (There are perhaps more viable options than for CT signals.)
Example:

$$x[n] = \{\underline{1}, 0, 0, 1/2, 0, 0, 1/4, 0, \ldots\} = \delta[n-0] + \frac{1}{2}\,\delta[n-3] + \frac{1}{4}\,\delta[n-6] + \cdots = \sum_{k=0}^{\infty} \left(\frac{1}{2}\right)^k \delta[n-3k]\,.$$

In MATLAB you have two basic choices.

- Enumeration: `xn = [0 0 1 0 3];` which typically means $x[n] = \delta[n-2] + 3\,\delta[n-5]$

- Signal synthesis: `n = [-5:4]; x = cos(n);` which means $x[n] = \cos(n)$ for $-5 \leq n \leq 4$ (and $x[n]$ is unspecified outside that range).

The `inline` function is also useful, *e.g.*, the unit impulse is: `imp = inline('n == 0', 'n');`
Skill: *Efficiently synthesize simple signals in* MATLAB.

Signal support characteristics

These are signal characteristics related to the *time* axis.

Support Interval
Roughly speaking the **support interval** of a signal is the set of times such that the signal is not zero. We often abbreviate and say simply **support** or **interval** instead of support interval.

- More precisely the support interval of a continuous-time signal $x_{\mathrm{a}}(t)$ is the smallest time interval[1] $[t_1, t_2]$ such that the signal is zero outside this interval.
- For a discrete-time signal $x[n]$, the support interval is a set of consecutive integers: $\{n_1, n_1 + 1, n_1 + 2, \ldots, n_2\}$. Specifically, n_1 is the largest integer such that $x[n] = 0$ for all $n < n_1$, and n_2 is the smallest integer such that $x[n] = 0$ for all $n > n_2$.

Duration
The **duration** or **length** of a signal is the length of its support interval.

- For continuous-time signals, duration = $t_2 - t_1$.
- What is the duration of a discrete-time signal? duration = $n_2 - n_1 + 1$.

Some signals have finite duration and others have infinite duration.

Example. The signal $x[n] = u[n-3] - u[n-7] + \delta[n-5] + \delta[n-9]$ has support $\{3, 4, \ldots, 9\}$ and duration 7.

[1]Intervals can be **open** as in (a, b), **closed** as in $[a, b]$, or half-open, half-closed as in $(a, b]$ and $[a, b)$. For continuous-time signals, in almost all cases of practical interest, it is not necessary to distinguish the support interval as being of one type or the other.

2.1.2

Classification of discrete-time signals

The **energy** of a discrete-time signal is defined as

$$E_x \triangleq \sum_{n=-\infty}^{\infty} |x[n]|^2 .$$

The **average power** of a signal is defined as

$$P_x \triangleq \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} |x[n]|^2 .$$

- If E is finite ($E < \infty$) then $x[n]$ is called an **energy signal** and $P = 0$.
- If E is infinite, then P can be either finite or infinite. If P is finite and nonzero, then $x[n]$ is called a **power signal**.

Example. Consider $x[n] = 5$ (a constant signal). Then

$$P = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} 5^2 = \lim_{N\to\infty} 5^2 = 25.$$

So $x[n]$ is a power signal.
What is E and is $x[n]$ an energy signal? Since P is nonzero, E is infinite.

More classifications
- $x[n]$ is **periodic** with **period** $N \in \mathbb{N}$ iff $x[n+N] = x[n] \;\forall n$
- Otherwise $x[n]$ is **aperiodic**

Fact: N-periodic signals are power signals with $P = \frac{1}{N} \sum_{n=0}^{N-1} |x[n]|^2$.

Symmetry
- $x[n]$ is **symmetric** or **even** iff $x[-n] = x[n]$
- $x[n]$ is **antisymmetric** or **odd** iff $x[-n] = -x[n]$

We can decompose any signal into even and odd components:

$$\begin{aligned} x[n] &= x_e[n] + x_o[n] \\ x_e[n] &\triangleq \frac{1}{2}(x[n] + x[-n]) \text{ Verify that this is even!} \\ x_o[n] &\triangleq \frac{1}{2}(x[n] - x[-n]) \text{ Verify that this is odd!} \end{aligned}$$

Example. $2\,u[n] = \frac{1}{2}(2\,u[n] + 2\,u[-n]) + \frac{1}{2}(2\,u[n] - 2\,u[-n]) = (1 + \delta[n]) + (u[n-1] - u[1-n])$

$$\{\ldots,0,0,\underline{2},2,2,\ldots\} = \{\ldots,1,1,\underline{2},1,1,\ldots\} + \{\ldots,-1,-1,\underline{0},1,1,\ldots\} .$$ ***Picture***

2.1.3

Simple manipulations of discrete-time signals

- Amplitude modifications
  - amplitude scaling $y[n] = a\,x[n]$, amplitude shift $y[n] = x[n] + b$
  - sum of two signals $y[n] = x_1[n] + x_2[n]$
  - product of two signals $y[n] = x_1[n]\,x_2[n]$
- Time modifications
  - Time shifting $y[n] = x[n-k]$. k can be positive (delayed signal) or negative (advanced signal) if signal stored in a computer
  - Folding or reflection or time-reversal $y[n] = x[-n]$
  - Time-scaling or **down-sampling** $y[n] = x[2n]$. (discard every other sample) (*cf.* continuous $f(t) = g(2t)$)
    Why? *e.g.*, to reduce CPU time in a preliminary data analysis, or to reduce memory.

2.6
Correlation of discrete-time signals

2.6.1
Cross-correlation sequences

The **cross correlation** of signals $x[n]$ and $y[n]$ is

$$\boxed{r_{xy}[l] \overset{\triangle}{=} \sum_{n=-\infty}^{\infty} x[n]\, y^*[n-l] = \sum_{n=-\infty}^{\infty} x[n+l]\, y^*[n], \qquad l = 0, \pm 1, \pm 2, \ldots,}$$

where l is called the **lag**. Recipe is almost the same as for convolution: shift, multiply, sum. No folding!

Example applications: time-delay estimation, frequency estimation.
(A 1999 Mercedes Benz has cruise-control that tracks car in front.) ***pictures***

2.6.2
Properties of cross correlation

- $r_{xy}[l] = r_{yx}^*[-l]$ (called **conjugate symmetry** or **Hermitian symmetry**)
- $r_{xy}[l] = x[l] * y^*[-l]$
- $|r_{xy}[l]| \leq \sqrt{E_x E_y}$ where $E_x = \sum_{n=-\infty}^{\infty} |x[n]|^2$ is the signal energy. This is called the **Cauchy-Schwarz inequality**.
- If $w[n] = \alpha\, x[n]$ then $r_{wy}[l] = \alpha\, r_{xy}[l]$

skip Proof for real signals:

$$\begin{aligned} 0 &\leq \sum_{n=-\infty}^{\infty} \left| \frac{x[n]}{\sqrt{E_x}} \pm \frac{y[n-l]}{\sqrt{E_y}} \right|^2 \\ &= \frac{1}{E_x} \sum_{n=-\infty}^{\infty} |x[n]|^2 + \frac{1}{E_y} \sum_{n=-\infty}^{\infty} |y[n-l]|^2 \pm \frac{2}{\sqrt{E_x E_y}} \left[\sum_{n=-\infty}^{\infty} x[n]\, y[n-l] \right] \\ &= 2 \pm 2 \frac{r_{xy}[l]}{\sqrt{E_x E_y}}. \end{aligned}$$

Thus $-\sqrt{E_x E_y} \leq r_{xy}[l] \leq \sqrt{E_x E_y}$.

Autocorrelation

The autocorrelation of a signal is the cross correlation of the signal with itself:

$$\boxed{r_{xx}[l] \overset{\triangle}{=} \sum_{n=-\infty}^{\infty} x[n]\, x^*[n-l] = \sum_{n=-\infty}^{\infty} x[n+l]\, x^*[n], \; l = 0, \pm 1, \pm 2, \ldots.}$$

It inherits the properties of cross correlation. In addition:
- $|r_{xx}[l]| \leq r_{xx}[0]$
- $r_{xx}[0] = E_x$

One application: determining period of sinusoidal signal.

2.6.3

2.6.4

2.6.5

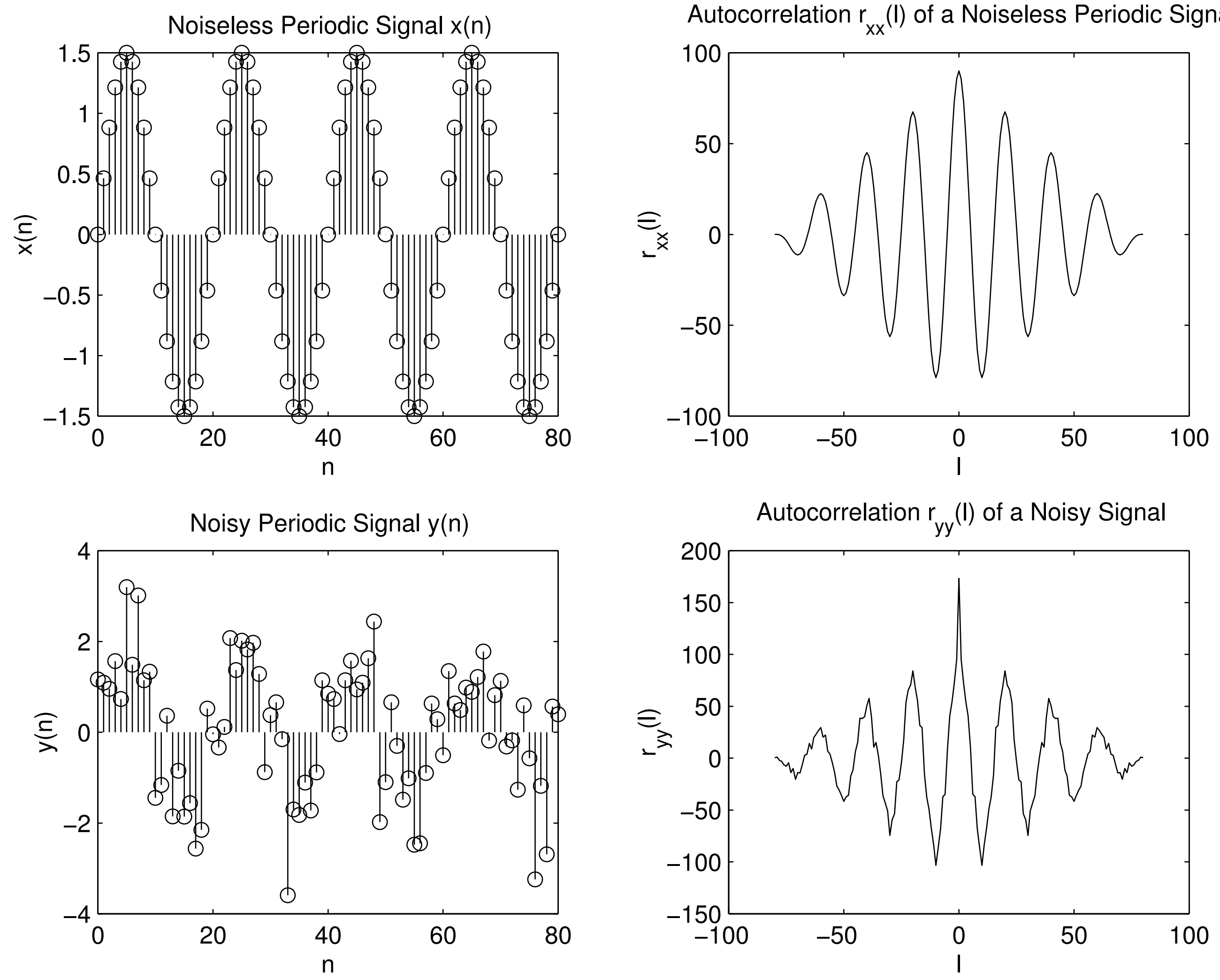


Figure 2.1: Example of autocorrelation of a periodic signal with a noisy signal having the same dominant frequency component.

2.2 Discrete-time systems

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

Mathematically speaking, a system is also a **function**. The input signal $x[n]$ is **transformed** by the system into a signal $y[n]$, which we express mathematically as

$$y[\cdot] = \mathcal{T}\{x[\cdot]\} \quad \text{or} \quad y[n] = \mathcal{T}\{x[\cdot]\}[n] \quad \text{or} \quad x[\cdot] \xrightarrow{\mathcal{T}} y[\cdot]\,.$$

The notation $y[n] = \mathcal{T}\{x[n]\}$ is mathematically vague. The reader must understand that in general $y[n]$ is a function *of the entire sequence* $\{x[n]\}$, not just the single time point $x[n]$.

$$\begin{matrix} \text{Input Signal} \\ x[n] \end{matrix} \rightarrow \boxed{\text{Discrete-time system}} \rightarrow \begin{matrix} \text{Output Signal} \\ y[n] \end{matrix}$$

2.2.1 Input-output description of systems

A discrete-time system can be described in many ways. One way is by its **input-output relationship**, which is a formula expressing the output signal in terms of the input signal.

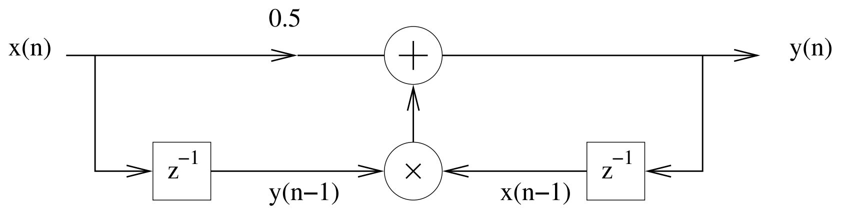
Example. The **accumulator** system.
$y[n] = \sum_{k=-\infty}^{n} x[k] = x[n] + x[n-1] + x[n-2] + \cdots$
$x[n] = u[n] - u[n-3] = \{\ldots, 0, 0, \underline{1}, 1, 1, 0, 0, \ldots\}$
$y[n] = \{\ldots, 0, 0, \underline{1}, 2, 3, 3, \ldots\}$. *Example 2.2.1(f) on p.58 has error.*
Alternative expression: $y[n] = \sum_{k=-\infty}^{n} x[k] = \sum_{k=-\infty}^{n-1} x[k] + x[n] = y[n-1] + x[n]$

Example. Interest-bearing **checking account** with monthly fee: $y[n] = 1.01\, y[n-1] + x[n] - 2\, u[n]$. Why the $u[n]$?
Notice the relevant "physical" quantities (interest rate, fee, balance) are captured in this mathematical expression.
The field of "Systems" deals with mathematical modeling of (more or less) "physical" things.

read about initial condition and initially relaxed

2.2.2 Block diagram representation of discrete-time systems

Example.



y(n) = y(n−1) x(n−1) + 0.5 x(n)

- adder
- constant multiplier
- signal multiplier
- **unit delay** element (why z^{-1} clear later)
  In real-time hardware, each delay element requires 1 signal sample worth of memory (to store each sample until the next arrives).
  How are delays implemented? With buffers / latches / flip-flops.

Other representations to be discussed later (mostly for LTI systems).
- Difference equation
- Impulse response
- System function
- pole-zero plot
- Frequency response

2.2.3

Classification of discrete-time systems
Skill: ***Determining classifications of a given DT system***

Two general aspects to categorize: time properties and amplitude properties.

Time properties (causality, memory, time invariance)

Causality

For a **causal** system, the output $y[n]$ at any time n depends *only* on the "present" and "past" inputs *i.e.*,

$$y[n] = F\{x[n], x[n-1], x[n-2], \ldots\}$$

where $F\{\cdot\}$ is any function.

Causality is necessary for real-time implementation, but many DSP problems involved stored data, *e.g.*, image processing (OCR) or restoration of analog audio recordings.

Otherwise **noncausal** system.

Memory

- For a **static system** or **memoryless** system, the output $y[n]$ depends only on the current input $x[n]$, not on previous or future inputs. Example: $y[n] = e^{x[n]}/\sqrt{n-2}$.
- Otherwise it is a **dynamic system** and must have memory.

Dynamic systems are the interesting ones and will be our focus. (This time we take the more complicated choice!)

Is a memoryless system necessarily causal? Yes. But dynamic systems can be causal or noncausal.

Time invariance

Systems whose input-output behavior does not change with time are called **time-invariant** and will be our focus
Why? "Easier" to analyze. Time-invariance is a desired property of many systems.

A relaxed system $\mathcal{T}$ is called **time invariant** or **shift invariant** iff

$$x[n] \xrightarrow{\mathcal{T}} y[n] \quad \text{implies that} \quad x[n-k] \xrightarrow{\mathcal{T}} y[n-k]$$

for *every* input signal $x[n]$ and *integer* time shift k.

Otherwise the system is called **time variant** or **shift variant**.

Graphically:

$$x[n] \quad \rightarrow \quad \boxed{\text{delay } z^{-k}} \rightarrow \boxed{\text{system } \mathcal{T}} \rightarrow y[n]$$

$$x[n] \quad \rightarrow \quad \boxed{\text{system } \mathcal{T}} \rightarrow \boxed{\text{delay } z^{-k}} \rightarrow y[n]$$

Recipe for showing time-invariance. (This method avoids potentially confusing $y(n,k)$ notation.)

- Determine output $y_1[n]$ due to a generic input $x_1[n]$.
- Determine output $y_2[n]$ due to input $x_2[n] = x_1[n-k]$.
- If $y_2[n] = y_1[n-k]$, then system is time-invariant.

Example: 3-point **moving average** $y[n] = \frac{1}{3}(x[n-1] + x[n] + x[n+1])$. Time invariant? yes.

- Output due to $x_1[n]$ is $y_1[n] = \frac{1}{3}(x_1[n-1] + x_1[n] + x_1[n+1])$.
- Output due to shifted input $x_2[n] = x_1[n-k]$ is
  $y_2[n] = \frac{1}{3}(x_2[n-1] + x_2[n] + x_2[n+1]) = \frac{1}{3}(x_1[n-k-1] + x_1[n-k] + x_1[n-k+1])$
- Since $y_1[n-k] = \frac{1}{3}(x_1[n-k-1] + x_1[n-k] + x_1[n-k+1]) = y_2[n]$, the system is time-invariant.

Example: down-sampler $y[n] = x[2n]$. Time invariant? no. How do we show lack of a property? Find counter-example. If $x[n] = \delta[n]$ then $y[n] = \delta[n]$. If $x[n] = \delta[n-1]$ then $y[n] = 0 \neq \delta[n-1]$. Simple counterexample all that is needed.

We will focus on time-invariant systems.

skip Here is another mathematical way of expressing time invariance. If

$$y[n] = F\{n, x[m_1(n)], x[m_2(n)], \ldots\}$$

for some functions $m_j : \mathbb{Z} \to \mathbb{Z}$, then the system is time-invariant iff for any signal $x[n]$:

$$F\{n-k, x[m_1(n-k)], x[m_2(n-k)], \ldots\} = F\{n, x[m_1(n)-k], x[m_2(n)-k], \ldots\} \; \forall n, k \in \mathbb{Z}.$$

The left side is "shifting the output" whereas the right side is "shifting the input."

"Amplitude" properties (stability, invertibility, linearity)

Invertibility

A system $\mathcal{T}$ is **invertible** if every output signal corresponds to a unique input signal. If so, then there exists an **inverse system** $\mathcal{T}^{-1}$ that can recover the input signal:

$$x[n] \to \boxed{\mathcal{T}} \to y[n] \to \boxed{\mathcal{T}^{-1}} \to x[n]\,.$$

We will derive tests for invertibility later.

Stability

A system is **bounded-input bounded-output (BIBO) stable** iff every bounded input produces a bounded output.

$$\text{If } \exists M_x \text{ s.t. } |x[n]| \leq M_x < \infty \;\forall n, \text{ then there must exist an } M_y \text{ s.t. } |y[n]| \leq M_y < \infty \;\forall n.$$

Usually M_y will depend on M_x.

Example: accumulator $y[n] = y[n-1] + x[n]$. Consider input signal $x[n] = u[n]$, which is bounded by $M_x = 1$. But $y[n] = n + 1 + y[-1]$ blows up, so the accumulator is an **unstable** system.

We will derive a simple test for BIBO stability shortly.

Linearity

We will also focus on **linear systems**.

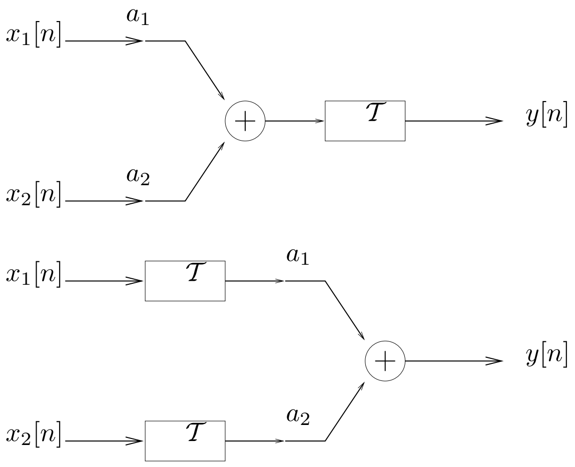
Why linearity?
- The class of **linear systems** is easier to analyze.
- Often linearity is desirable - avoids distortions.
- Many nonlinear systems are approximately linear, so first-order analysis is linear case.

A system $\mathcal{T}$ is **linear** iff

$$\mathcal{T}\{a_1\, x_1[n] + a_2\, x_2[n]\} = a_1\, \mathcal{T}\{x_1[n]\} + a_2\, \mathcal{T}\{x_2[n]\}, \; \textit{i.e.}, \; a_1\, x_1[n] + a_2\, x_2[n] \xrightarrow{\mathcal{T}} a_1\, y_1[n] + a_2\, y_2[n],$$

for any signals $x_1[n]$, $x_2[n]$, and constants a_1 and a_2.
Otherwise the system is called **nonlinear**.

Block diagram illustration of linearity test.



Two important special cases of linearity property.

- **scaling property**: $\boxed{\mathcal{T}\{a\,x[n]\} = aT[x[n]]}$
  Note that from $a = 0$ we see that zero input signal implies zero output signal for a linear system.
- **additivity property**: $\boxed{\mathcal{T}\{x_1[n] + x_2[n]\} = \mathcal{T}\{x_1[n]\} + \mathcal{T}\{x_2[n]\}}$
  Using proof-by-induction, one can easily extend this property to the general **superposition property**:

$$\boxed{\mathcal{T}\left[\sum_{k=1}^{K} x_k[n]\right] = \sum_{k=1}^{K} \mathcal{T}\{x_k[n]\}\,.}$$

In words: the response of a linear system to the sum of several signals is the sum of the response to each of the signals.
In general superposition need not hold for infinite sums; additional continuity assumptions are required.
We assume the superposition summation holds even for **infinite sums** without further comment in this course.

Example: Proof that the accumulator is a linear system, where $y[n] = \sum_{k=-\infty}^{n} x[k]$.
Method:

- Find output signal $y_1[n]$ for a general input signal $x_1[n]$.
- "Repeat" for input $x_2[n]$ and $y_2[n]$.
- Find output signal $y[n]$ when input signal is $x[n] = a_1\,x_1[n] + a_2\,x_2[n]$.
- If $y[n] = a_1\,y_1[n] + a_2\,y_2[n]$, $\forall n$, then the system is linear.

For the accumulator, $y_1[n] = \sum_{k=-\infty}^{n} x_1[n]$ and $y_2[n] = \sum_{k=-\infty}^{n} x_2[n]$. If the input is $x[n] = a_1\,x_1[n] + a_2\,x_2[n]$, then the output is

$$y[n] = \sum_{k=-\infty}^{n} x[n] = \sum_{k=-\infty}^{n} (a_1\,x_1[n] + a_2\,x_2[n]) = a_1 \sum_{k=-\infty}^{n} x_1[n] + a_2 \sum_{k=-\infty}^{n} x_2[n] = a_1\,y_1[n] + a_2\,y_2[n]\,.$$

Since this holds for all n, for all input signals $x_1[n]$ and $x_2[n]$, and for any constants a_1 and a_2, the accumulator is linear.

Example: To show that $y[n] = \sqrt{x[n]}$ is nonlinear, all that is needed is a counter-example to the above properties. The scaling property will usually suffice. Let $x_1[n] = 2$, a constant signal. Then $y_1[n] = \sqrt{2}$. Now suppose the input is $x[n] = 3\,x_1[n] = 6$, then the output is $y[n] = \sqrt{6} \neq 3\,y_1[n]$, so the system is nonlinear.

2.2.4

Interconnection of discrete-time systems

skip essentially repeated in 2.3.4

2.3 Analysis of discrete-time linear time-invariant systems (Our focus is LTI hereafter)

Why analysis? So far we only have input-output relationships. For a given system can compute $y[n]$ for selected $x[n]$'s, but very difficult to design filters etc. by such trial-and-error.

Overview: $x[n] \rightarrow \boxed{\text{LTI } h[n]} \rightarrow y[n] = x[n] * h[n]$.

Linearity leads to the above superposition property, which simplifies the analysis. Time-invariance then further simplifies.

2.3.1 Techniques for the analysis of linear systems

General strategy:
- Decompose input signal $x[n]$ into a *weighted sum of elementary functions* $x_k[n]$, *i.e.*, $x[n] = \sum_k c_k\, x_k[n]$
- Determine response of system to each elementary function (this should be easy from input-output relationship):

$$x_k[n] \xrightarrow{\mathcal{T}} y_k[n]$$

- Apply superposition property:

$$x[n] = \sum_k c_k\, x_k[n] \xrightarrow{\mathcal{T}} y[n] = \sum_k c_k\, y_k[n]\,.$$

Two particularly good choices for the elementary functions $x_k[n]$:
- impulse functions $\delta[n-k]$
- complex exponentials $e^{\jmath\omega_k n}$ (later).

2.3.2 Resolution of discrete-time signal into impulses

It follows directly from the definition of $\delta[n]$ that

$$\boxed{x[n] = \sum_{k=-\infty}^{\infty} x[k]\, \delta[n-k]\,.}$$

This is the **sifting property** of the unit impulse function.

Example. $x[n] = \{\underline{8}, \pi, 0, \sqrt{7}\} \Longrightarrow x[n] = 8\,\delta[n-0] + \pi\,\delta[n-1] + \sqrt{7}\,\delta[n-3]\,.$

2.3.3 Response of LTI systems to arbitrary inputs: the convolution sum

Define the special symbol $h_k[n]$ to denote the system output when the input is $\delta[n-k]$, *i.e.*,

$$\delta[n-k] \xrightarrow{\mathcal{T}} h_k[n]\,.$$

In words: $h_k[n]$ is the response to an impulse at the kth sample.

Now use superposition to determine the response to a general input signal $x[n]$:

$$x[n] = \sum_k x[k]\, \delta[n-k] \xrightarrow{\mathcal{T}} y[n] = \sum_k x[k]\, h_k[n]\,.$$

Thus we have proven the **superposition summation** for any **linear** system $\mathcal{T}$:

$$\boxed{x[n] \underbrace{\xrightarrow{\mathcal{T}}}_{\text{Linear}} y[n] = \sum_{k=-\infty}^{\infty} x[k]\, h_k[n]\,.}$$

In words: overall output is weighted sum of response due to kth sample.
Every input sample that comes into the system causes the system to respond. For a linear system, the overall response is the sum of the contributions due to each input sample.

We have not yet used time-invariance. If the system is **time-invariant**, then the response to a delayed impulse $\delta[n-k]$ is just a delayed version of the response to an impulse at time 0: $\delta[n]$. Thus, for a LTI system: $\boxed{h_k[n] = h_0[n-k]}$

Thus for time-invariant linear systems, the superposition summation becomes

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]\, h_0[n-k]\,.$$

The "0" subscript is redundant, so we drop it and just write $h[n-k]$ instead. We call $h[n]$ the **impulse response** of the system.

In summary, for an LTI system with impulse response $h[n]$, the response to an arbitrary input is given by the **convolution sum**:

$$\boxed{\underbrace{x[n] \xrightarrow{\mathcal{T}} y[n]}_{\text{LTI}} = \sum_{k=-\infty}^{\infty} x[k]\, h[n-k] = \sum_{k=-\infty}^{\infty} x[n-k]\, h[k] \overset{\triangle}{=} x[n] * h[n] = (x * h)[n].}$$

The second equality, which implies that convolution is **commutative**, it obtained from the first by letting $m = n-k$ so that we have $y[n] = \sum_m x[n-m]\, h[m]$. Now just replace m with k since it is just a dummy summation argument and any letter (other than n...) will do.

The convolution sum shows that with the impulse response $h[n]$, you can compute the output $y[n]$ for *any* input signal $x[n]$. Thus

$$\boxed{\text{A LTI system is characterized completely by its impulse response } h[n].}$$

Example. If $y[n] = a\, y[n-1] + x[n]$, then (later we show that) $h[n] = a^n\, u[n]$.
For the input $x[n] = u[n]$, find the output $y[n]$, which is called the **unit-step response** of the system. Assume $a \neq 1$.

First a useful fact. $\boxed{\sum_{k=0}^{n} a^k = \begin{cases} \dfrac{1-a^{n+1}}{1-a}, & a \neq 1 \\ n+1, & a = 1. \end{cases}}$ Now using the second convolution formula above:

$$y[n] = \sum_{k=-\infty}^{\infty} x[n-k]\, h[k] = \sum_{k=-\infty}^{\infty} u[n-k]\, a^k\, u[k] = \begin{cases} \sum_{k=0}^{n} a^k, & n \geq 0 \\ 0, & \text{otherwise} \end{cases} = \frac{1-a^{n+1}}{1-a}\, u[n]\,.$$

Picture . Inner terms nonzero only where $k \leq n$ and $k \geq 0$, *i.e.*, $0 \leq k \leq n$. Thus nonzero only if $n \geq 0$.

Sometimes the summation limits are less obvious, and one can use the following recipe for convolution. **Skill: *convolving***
- Fold: fold $h[k]$ about $k = 0$ to get $h[-k]$
- Shift: shift $h[-k]$ by n to get $h[n-k]$
- Multiply: $x[k]$ by $h[n-k]$ for every k
- Sum: $y[n] = \sum_{k=-\infty}^{\infty} x[k]\, h[n-k]$

Repeat for all possible n; generally breaks in to a few intervals.

skip Graphically for above example

Fold: $h[-k] = a^{-k}\, u[-k]$
Shift: $h[n-k] = a^{n-k}\, u[n-k]$
Multiply: $x[k]\, h[n-k] = u[k]\, a^{n-k}\, u[n-k]$, nonzero only for $0 \leq k \leq n$ and hence $n \geq 0$.
Sum: (using $m = n-k$)

$$\sum_{k=0}^{n} a^{n-k} = \sum_{m=0}^{n} a^m = \begin{cases} \dfrac{1-a^{n+1}}{1-a}, & a \neq 1 \\ n+1, & a = 1. \end{cases} \; u[n]$$

as before.

Example: $h[n] = \delta[n-n_0]$, showing that $y[n] = x[n-n_0]$.
What is the system with this impulse response called? A **delay** or **shifter**. ***Picture***

2.3.4

Properties of convolution and the interconnection of LTI systems

Skill: ***Use properties to simplify LTI systems.***
Awareness of these properties is necessary for efficient designs.

Support
If $x[n]$ has support $n = N_1, \ldots, N_1 + L_1 - 1$ (length L_1)
and $h[n]$ has support $n = N_2, \ldots, N_2 + L_2 - 1$ (length L_2)
then $y[n] = x[n] * h[n]$ has support $n = N_1 + N_2, \ldots, N_1 + L_1 - 1 + N_2 + L_2 - 1$ (length L_2)
What is the duration of $y[n]$? $L = L_1 + L_2 - 1$

Time-shift

$$x[n] * h[n] = y[n] \quad \Longrightarrow \quad \begin{array}{c} x[n-n_0] * h[n] = y[n-n_0] \\ x[n-n_1] * h[n-n_2] = y[n-n_1-n_2] \end{array}$$

Commutative law

$$\boxed{x[n] * h[n] = h[n] * x[n]}$$

Proof:

$$x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k]\, h[n-k] = \sum_{k'=-\infty}^{\infty} x[n-k']\, h[k'] = h[n] * x[n],$$

where $k' = n - k$.

Associative law

$$\boxed{(x[n] * h_1[n]) * h_2[n] = x[n] * (h_1[n] * h_2[n])}$$

Proof: let $y_1[n] = (x[n] * h_1[n]) * h_2[n]$ and $y_2[n] = x[n] * (h_1[n] * h_2[n])$.
We must show $y_1[n] = y_2[n]$.

$$\begin{aligned} y_1[n] &= \sum_{k=-\infty}^{\infty} (x * h_1)[n-k]\; h_2[k] = \sum_k \left(\sum_l x[l]\, h_1[n-k-l] \right) h_2[k] = \sum_l x[l] \left(\sum_k h_1[n-l-k]\, h_2[k] \right) \\ &= \sum_l x[l] (h_1 * h_2)[n-l] = (x * [h_1 * h_2])[n] = y_2[n]\,. \end{aligned}$$

The above laws hold in general for any number of systems connected in **series**. So the following notation is acceptable:

$$h[n] = h_1[n] * h_2[n] * \cdots * h_k[n]\,.$$

In particular:

$$\begin{aligned} (x * h_1) * h_2 &= x * (h_1 * h_2) \\ &= x * (h_2 * h_1) \\ &= (x * h_2) * h_1, \end{aligned}$$

so the order of serial connection of LTI systems with impulse response h_1 and h_2 does not affect output signal. See picture.

Distributive law

$$\boxed{x[n] * (h_1[n] + h_2[n]) = (x[n] * h_1[n]) + (x[n] * h_2[n])}$$

Proof:

$$\begin{aligned} x[n] * (h_1[n] + h_2[n]) &= \sum_{k=-\infty}^{\infty} x[n-k]\, (h_1[k] + h_2[k]) \\ &= \sum_{k=-\infty}^{\infty} x[n-k]\, h_1[k] + \sum_{k=-\infty}^{\infty} x[n-k]\, h_2[k] \\ &= x[n] * h_1[n] + x[n] * h_2[n]\,. \end{aligned}$$

All of the above follow from simple properties of addition and multiplication due to LTI assumption.

Here are the above properties illustrated with block diagrams.

$$x[n] \to \boxed{h[n]} \to y[n]$$

Commutative yields same output!

$$h[n] \to \boxed{x[n]} \to y[n]$$

The order of interconnection of systems in **series** or **cascade** does not affect the output[2].

$$x[n] \to \boxed{h_1[n]} \to \boxed{h_2[n]} \to y[n]$$

Associative

$$x[n] \to \boxed{h_1[n] * h_2[n]} \to y[n]$$

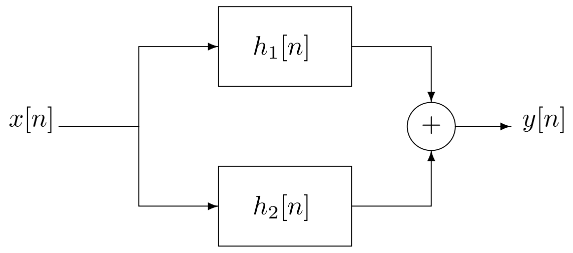
Commutative

$$x[n] \to \boxed{h_2[n] * h_1[n]} \to y[n]$$

Associative

$$x[n] \to \boxed{h_2[n]} \to \boxed{h_1[n]} \to y[n]$$

Parallel connection:



Distributive: $x[n] \to \boxed{h[n] = h_1[n] + h_2[n]} \to y[n]$

Example.

$$x[n] \to \boxed{h_1[n] = \delta[n] - \delta[n-1]} \to \boxed{h_2[n] = u[n]} \to y[n]$$

Overall impulse response:

$$h[n] = h_1[n] * h_2[n] = (\delta[n] - \delta[n-1]) * u[n] = u[n] - u[n-1] = \delta[n]\,.$$

Convolution with impulses

Time shift / delay:

$$x[n] * \delta[n - n_0] = x[n - n_0]$$

Identity:

$$x[n] * \delta[n] = x[n]$$

Cascade of time shifts:

$$\delta[n - n_1] * \delta[n - n_2] = \delta[n - n_1 - n_2]$$

[2]This claim only holds for ideal LTI systems based on real numbers. In digital systems where the signal values and filter coefficients are quantized, the order of interconnection can matter in some cases.

Properties of LTI systems in terms of the impulse response

Since an LTI system is characterized completely by its impulse response, we should be able to express all of the other four properties (**causality**, **invertibility**, **memory**, **stability**) in terms of $h[n]$.

2.3.5 Causal LTI systems

Recall system is causal iff output $y[n]$ depends only on present and past values of input.

For an LTI system with impulse response $h[n]$:

$$y[n] = \sum_{k=-\infty}^{\infty} h[k]\, x[n-k] = \sum_{k=0}^{\infty} h[k]\, x[n-k] + \sum_{k=-\infty}^{-1} h[k]\, x[n-k]\,.$$

The first sum depends on present and past input samples $x[n], x[n-1], \ldots$, whereas the second sum depends on future input samples $x[n+1], x[n+1], \ldots$.
Thus the system is causal iff the impulse response terms corresponding to the second sum is zero.
These terms are $h[-1], h[-2], \ldots$.

$$\boxed{\text{An LTI system is } \textbf{causal} \text{ iff its impulse response } h[n] = 0 \text{ for all } n < 0.}$$

In the causal case the convolution summation simplifies slightly since we can drop the right sum above:

$$y[n] = \sum_{k=0}^{\infty} h[k]\, x[n-k] = \sum_{k=-\infty}^{n} x[k]\, h[n-k]\,.$$

Example. Is the LTI system with $h[n] = u[n - n_0 - 5]$ causal? Only if $n_0 + 5 \geq 0$.

A **causal sequence** is a sequence $x[n]$ which is zero for all $n < 0$.

If the input to a **causal** LTI system is a **causal sequence**, then the output is simply

$$y[n] = \begin{cases} 0, & n < 0 \\ \sum_{k=0}^{n} h[k]\, x[n-k] = \sum_{k=0}^{n} x[k]\, h[n-k], & n \geq 0. \end{cases}$$

The above sum is precisely what is computed by MATLAB's `conv` function, for finite-length $x[n]$ and $h[n]$.

Invertibility of LTI systems

Fact: if an LTI system $\mathcal{T}$ is invertible, then its corresponding inverse system is also LTI.
(Proof left as an exercise.)

Thus the inverse system has an impulse response, say, $h_{\text{inv}}[n]$ such that

$$x[n] \to \boxed{h[n]} \to \boxed{h_{\text{inv}}[n]} \to x[n]\,.$$

In other words: $\boxed{h[n] * h_{\text{inv}}[n] = \delta[n]\,.}$

2.3.6 Stability of LTI systems

Recall $y[n] = \sum_{k=-\infty}^{\infty} h[k]\, x[n-k]$ so by the **triangle inequality**[3]

$$|y[n]| = \left| \sum_{k=-\infty}^{\infty} h[k]\, x[n-k] \right| \leq \sum_{k=-\infty}^{\infty} |h[k]\, x[n-k]| = \sum_{k=-\infty}^{\infty} |h[k]|\, |x[n-k]| \leq M_x \sum_{k=-\infty}^{\infty} |h[k]|$$

if $|x[n]| \leq M_x \; \forall n$.

Thus, for an LTI system to be BIBO stable, it is sufficient that its impulse response be **absolutely summable**, *i.e.*,

$$\sum_{n=-\infty}^{\infty} |h[n]| < \infty,$$

since in that case, if $x[n]$ is bounded, so will be $y[n]$.

It is also *necessary* for the above summation to be finite! To prove necessity, we construct a counter-example. We show that if $h[n]$ is *not* absolutely summable, then we can construct some bounded input signal for which the output is unbounded. A suitable choice is

$$x[n] = \begin{cases} h^*[-n] \,/\, |h[-n]|\,, & h[-n] \neq 0 \\ 0, & h[-n] = 0, \end{cases}$$

which is bounded by $M_x = 1$. But

$$y[0] = \sum_{k=-\infty}^{\infty} h[k]\, x[-k] = \sum_{k=-\infty}^{\infty} |h[k]| = \infty$$

if the impulse response is not absolutely summable, so the output would not be bounded for the specified bounded input signal.

Thus we have shown the following important property.

$$\boxed{\text{A LTI system is BIBO stable iff its impulse response is absolutely summable, } i.e., \sum_{n=-\infty}^{\infty} |h[n]| < \infty.}$$

Example. Accumulator: $y[n] = y[n-1] + x[n]$.
What is impulse response? Let $x[n] = \delta[n]$, then $y[n] = u[n]$. So $h[n] = u[n]$.
Stable? No: $\sum_{n=-\infty}^{\infty} |h[n]| = \infty$, so unstable.

One can also show the following for a BIBO stable system (see text).
- The impulse response $h[n]$ goes to zero as $n \to \infty$.
- If the input $x[n]$ has finite duration, then the output $y[n]$ will decay to zero as $n \to \infty$.

2.3.7 Duration of impulse response (the last classification for now)

Two classes of LTI systems.
- **finite impulse response** or **FIR**: only a finite number of $h[n]$ are nonzero.
- **infinite impulse response** or **IIR**: an infinite number of $h[n]$ are nonzero.

A LTI system is **memoryless** or **static** iff its impulse response has the form $h[n] = c\,\delta[n]$.

[3] $|a+b| \leq |a| + |b|$ or more generally $\left|\sum_k a_k\right| \leq \sum_k |a_k|$.

2.4

Discrete-time systems described by difference equations

The convolution summation $y[n] = \sum_{k=-\infty}^{\infty} h[k]\, x[n-k]$ looks fine on paper, but if the impulse response is IIR, it cannot be implemented directly with that formula since there would be an infinite number of adds and multiplies!
(And there are many cases where IIR is desirable, *e.g.*, telephone echo cancellation.)

For an arbitrary impulse response $h[n]$, there may not exist an implementation with a finite number of flops. Fortunately however, there is a broad class of interesting and useful IIR systems that one *can* implement using **difference equations**.

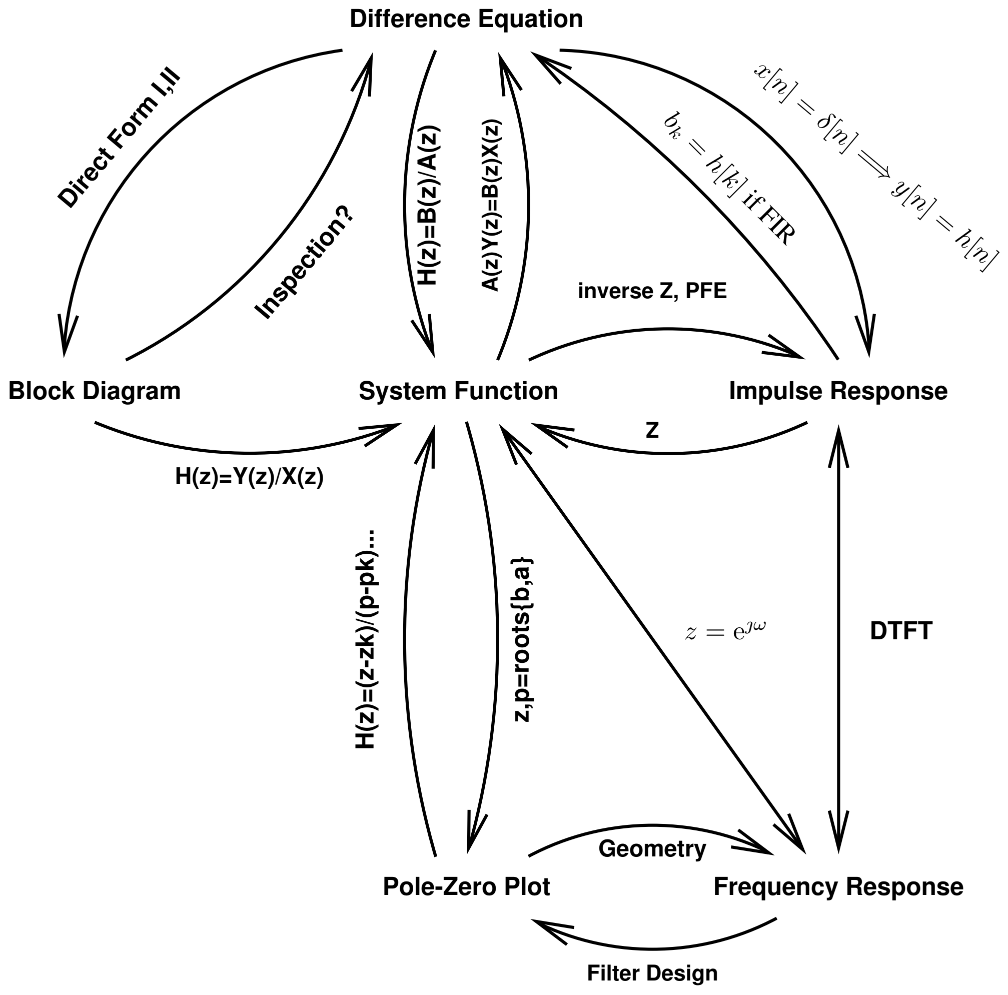
Example: Consider LTI system with impulse response $h[n] = a^n\, u[n]$. FIR or IIR? IIR.
Naive approach to implementation would just truncate the infinite sum to m terms (may need large m):

$$y[n] = \sum_{k=-\infty}^{\infty} h[k]\, x[n-k] = \sum_{k=0}^{\infty} a^k\, x[n-k] \approx x[n] + a\, x[n-1] + a^2\, x[n-2] + \cdots + a^m\, x[n-m]\,.$$

Efficient approach uses mathematical properties:

$$\begin{aligned} y[n] &= \sum_{k=0}^{\infty} a^k\, x[n-k] = x[n] + \sum_{k=1}^{\infty} a^k\, x[n-k] = x[n] + \sum_{l=0}^{\infty} a^{l+1}\, x[n-(l+1)] = x[n] + a \sum_{l=0}^{\infty} a^l\, x[(n-1)-l] \\ &= x[n] + a\, y[n-1], \end{aligned}$$

where $l = k-1$. This simple manipulation led to a **recursive** expression for $y[n]$, and one that is easily implemented using just one delay, one multiply, and one add. And, it is *exact*, unlike truncation! ***block diagram***



Skill: ***Understand role of each representation in describing system characteristics.***
Skill: ***Convert between various representations of LTI systems described by difference equations***

2.4.1

Recursive and nonrecursive discrete-time systems

2.4.2

LTI systems via constant-coefficient difference equations

A realizable system must only require a finite number of operations. A very general form is those systems that can be described by **linear constant-coefficient difference equations**:

$$\boxed{y[n] = -\sum_{k=1}^{N} a_k \, y[n-k] + \sum_{k=0}^{M} b_k \, x[n-k] \, .}$$

Such systems are the discrete-time analog of **linear constant-coefficient differential equations** for continuous-time systems.

Such systems are **linear** and **time-invariant** and **causal**.

See MATLAB's `y = filter(b,a,x)` command.

Example. The accumulator system $y[n] = y[n-1] + x[n]$, where $N = 1$, $a_1 = -1$, $M = 0$, $b_0 = 1$.

Categories:

If $N \geq 1$:
- the current output $y[n]$ depends on past outputs $y[n-1], \ldots, y[n-N]$,
- the system is called **recursive**
- the impulse response is IIR. (Usually we use z-transform to find $h[n]$.)

If $N = 0$
- the output $y[n]$ depends only on the current input sample and on M past input samples
- the system is called **nonrecursive**
- the impulse response is FIR and is given by

$$h[n] = \left\{\underline{b_0}, b_1, \ldots, b_M\right\} = \sum_{k=0}^{M} b_k \, \delta[n-k] \, .$$

2.4.3

Solution of linear constant-coefficient difference equations

The **homogeneous solution** or **zero-input solution** is of the form of linear combinations of $\left\{\lambda^n, \ n\lambda^n, \ \ldots n^N \lambda^n\right\}$, analogous to $\left\{\mathrm{e}^{\lambda t}, \ t \, \mathrm{e}^{\lambda t}, \ \ldots t^N \, \mathrm{e}^{\lambda t}\right\}$ for differential equations, where the λ's are roots of the **characteristic polynomial** $\sum_{k=0}^{N} a_k \lambda^{n-k}$.

2.4.4

Impulse response of a LTI recursive system

Brute force: let $x[n] = \delta[n]$ and execute recursion to find $y[n] = h[n]$. Easier to study using z-transform. Later...

Summary of difference equations

Main point is recursive implementation with a moderate number of adds and multiplies can yield a broad class of IIR impulse responses, but not all possible IIR cases.

Filter design is partly about how to make efficient approximations to a desired impulse response! (N and M limited by DSP processing rate / sampling rate).

flops required: $M + N + 1$ multiplies, $M + N$ adds, or about $2(M + N)$ flops

40kHz sampling, 1Mflop DSP, can give 1000/40 = 25 flops per sample, so $M + N \approx 12$ is the realizable system order for that chip

TMS 320C67 is 1 Gflop (!)

2.5 Implementation of discrete-time systems

2.5.1 Structures for realization of LTI systems

Example.

$$y[n] = -a_1\, y[n-1] + b_0\, x[n] + b_1\, x[n-1]$$

Approach 1:

$$\begin{aligned} y[n] &= -a_1\, y[n-1] + v[n] \\ v[n] &= b_0\, x[n] + b_1\, x[n-1] \end{aligned}$$

Approach 2:

$$\begin{aligned} w[n] &= -a_1\, w[n-1] + x[n] \\ y[n] &= b_0\, w[n] + b_1\, w[n-1] \end{aligned}$$

See Fig. 2.2.

Generalizes to higher order systems described by difference equations.

$$y[n] = -\sum_{k=1}^{N} a_k\, y[n-k] + \sum_{k=0}^{M} b_k\, x[n-k]$$

Approach 1:

$$\begin{aligned} v[n] &= \sum_{k=0}^{M} b_k\, x[n-k] \\ y[n] &= -\sum_{k=1}^{N} a_k\, y[n-k] + v[n] \end{aligned}$$

The first system $v[n] = \ldots$ is nonrecursive, where as the second system is recursive.

See Fig. 2.3.

Approach 2:

$$\begin{aligned} w[n] &= -\sum_{k=1}^{N} a_k\, w[n-k] + x[n] \\ y[n] &= \sum_{k=0}^{M} b_k\, w[n-k] \end{aligned}$$

See Fig. 2.4.

2.5.2 Recursive and nonrecursive realization of FIR systems

Any FIR system can be implemented nonrecursively (this is obvious using Direct Form I). However, any FIR system can be also be implemented recursively, which can sometimes save flops. Book gives example for a moving average system. We will discuss more later, perhaps.

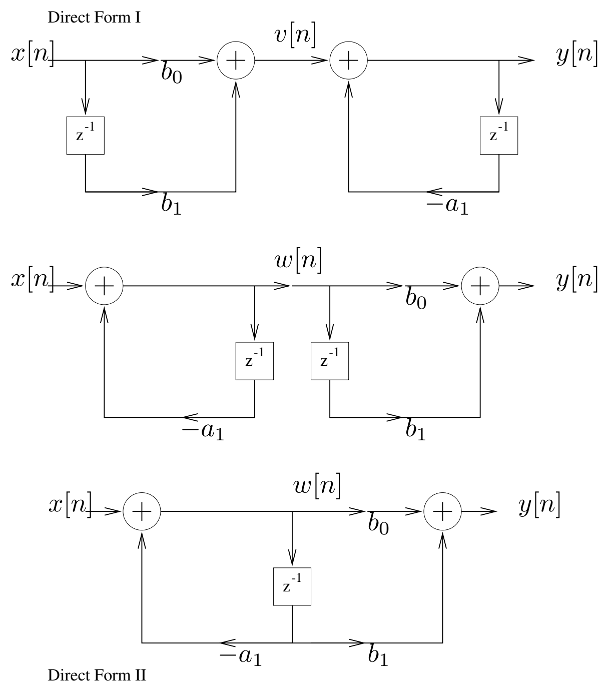


Figure 2.2: Conversion from Form I to Form II.

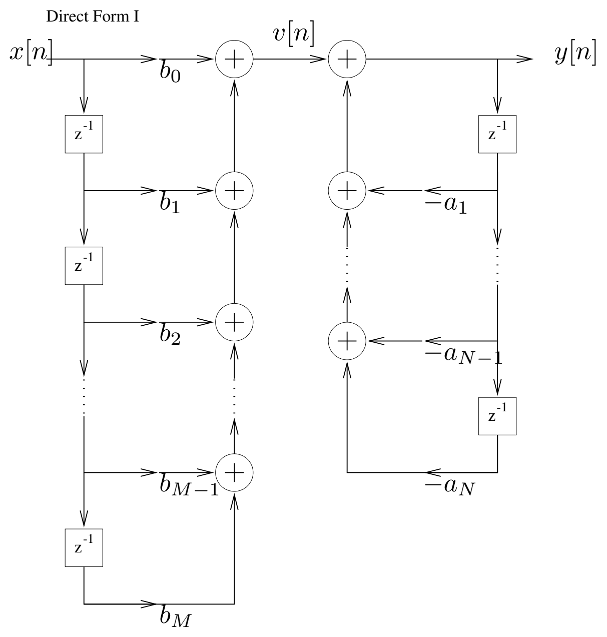


Figure 2.3: General Direct Form I.

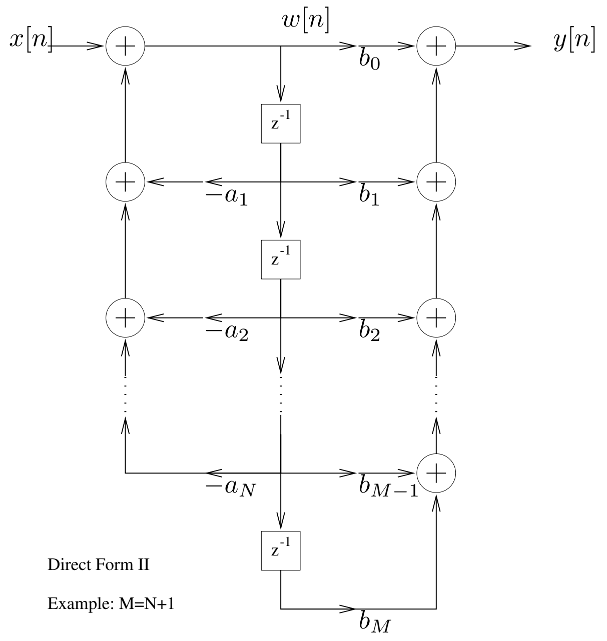


Figure 2.4: General Direct Form II.

2.7 Summary

Classified signals and systems.
Focus on LTI systems, characterized in terms of impulse response.
Output of a LTI system is convolution of input signal with impulse response function.