BASIC SYSTEM PROPERTIES

What: Input \( x[n] \rightarrow \text{SYSTEM} \rightarrow y[n] \) output.

Why: Design the system to filter input \( x[n] \).

DEF: A system is **LINEAR** if these two properties hold:

1. **Scaling**: If \( x[n] \rightarrow \text{SYS} \rightarrow y[n] \), then \( ax[n] \rightarrow \text{SYS} \rightarrow ay[n] \)
   for: any constant \( a \). NOT true if \( a \) varies with time (i.e., \( a[n] \)).

2. **Superposition**: If \( x_1[n] \rightarrow \text{SYS} \rightarrow y_1[n] \) and \( x_2[n] \rightarrow \text{SYS} \rightarrow y_2[n] \),
   Then: \( (ax_1[n] + bx_2[n]) \rightarrow \text{SYS} \rightarrow (ay_1[n] + by_2[n]) \)
   for: any constants \( a, b \). NOT true if \( a \) or \( b \) vary with time (i.e., \( a[n], b[n] \)).

EX: \( y[n] = 3x[n-2] \); \( y[n] = x[n+1] - nx[n+2] + 2x[n-1] \); \( y[n] = \sin(n)x[n] \).

NOT: \( y[n] = x^2[n] \); \( y[n] = \sin(x[n]) \); \( y[n] = |x[n]| \); \( y[n] = x[n]/x[n-1] \).

HOW: If any nonlinear function of \( x[n] \), not linear. Nonlinear of just \( n \) OK.

DEF: A system is **TIME-ININVARIANT** if this property holds:

If \( x[n] \rightarrow \text{SYS} \rightarrow y[n] \), then \( x[n - N] \rightarrow \text{SYS} \rightarrow y[n - N] \)
for: any integer time delay \( N \). NOT true if \( N \) varies with time (e.g., \( N(n) \)).


HOW: If \( n \) appears anywhere other than in \( x[n] \), not time-invariant. Else OK.

DEF: A system is **CAUSAL** if it has this form for some function \( F(\cdot) \):

\[ y[n] = F(x[n], x[n-1], x[n-2], \ldots) \text{ (present and past input only).} \]

Note: Physical systems must be causal. But DSP filters need not be causal!

DEF: A system is **MEMORYLESS** if \( y[n] = F(x[n]) \) (present input only).

DEF: A system is **(BIBO) STABLE** iff: Let \( x[n] \rightarrow \text{SYSTEM} \rightarrow y[n] \).

If \( |x[n]| < M \) for some constant \( M \), then \( |y[n]| < N \) for some \( N \).

i.e.: “Every bounded input (BI) produces a bounded output (BO).”

HOW: BIBO stable \( \iff \sum_{n=-\infty}^{\infty} |h[n]| < L \) for some constant \( L \)

where: Impulse \( \delta[n] \rightarrow \text{SYS} \rightarrow h[n] = \text{impulse response.} \)

EX: A time-invariant system is observed to have these two responses:
\( \{0, 0, 3\} \rightarrow \text{SYS} \rightarrow \{0, 1, 0, 2\} \) and \( \{0, 0, 0, 1\} \rightarrow \text{SYS} \rightarrow \{1, 2, 1\} \).

Prove: The system is nonlinear.

Proof: By contradiction. Suppose the system is linear. But then:
\( \{0, 0, 0, 1\} \rightarrow \text{SYS} \rightarrow \{1, 2, 1\} \) implies \( \{0, 0, 3\} \rightarrow \text{SYS} \rightarrow \{3, 6, 3\} \)
since we know it is time-invariant. Then \( \{0, 0, 3\} \) produces two outputs!
CONVOLUTION AND IMPULSE RESPONSE

\[ x[n] = \{3, 1, 4, 6\} \leftrightarrow x[n] = 3\delta[n+1] + 1\delta[n] + 4\delta[n-1] + 6\delta[n-2]. \]

**Note:** \( x[n] = \sum_i x[i]\delta[n-i] = x[n] * \delta[n] \) (sifting property of impulse).

**Delay:** \( x[n-D] \) is \( x[n] \) shifted right (later) if \( D > 0 \); left (earlier) if \( D < 0 \).

**Fold:** \( x[-n] \) is \( x[n] \) flipped/folded/reversed around \( n = 0 \).

**Both:** \( x[N-n] \) is \( x[-n] \) shifted right if \( N > 0 \) (since \( x[0] \) is now at \( n = N \)).

**FOR LINEAR TIME-INVARIANT (LTI) SYSTEMS:**

1. \( \delta[n] \rightarrow \boxed{\text{LTI}} \rightarrow h[n] \) Definition of **Impulse response** \( h[n] \).
2. \( \delta[n-i] \rightarrow \boxed{\text{LTI}} \rightarrow h[n-i] \) **Time invariant:** delay by \( i \).
3. \( x[i]\delta[n-i] \rightarrow \boxed{\text{LTI}} \rightarrow x[i]h[n-i] \) **Linear:** scale by \( x[i] \).
4. \( \sum_i x[i]\delta[n-i] \rightarrow \boxed{\text{LTI}} \rightarrow \sum_i x[i]h[n-i] \) **Linear:** superposition.
5. \( x[n] \rightarrow \boxed{\text{LTI}} \rightarrow y[n] = \sum_i x[i]h[n-i] = h[n] * x[n] \) **Convolution.**

Input \( x[n] \) into LTI system with no initial stored energy→output \( y[n] \).

**PROPERTIES OF DISCRETE CONVOLUTION**

1. \( y[n] = h[n] * x[n] = x[n] * h[n] = \sum_i h[i]x[n-i] = \sum h[n-i]x[i] \).
2. \( h[n], x[n] \) both causal \( (h[n] = 0 \text{ for } n < 0 \text{ and } x[n] = 0 \text{ for } n < 0) \)
   \( \rightarrow y[n] = \sum_{i=0}^n h[i]x[n-i] = \sum_{i=0}^n h[n-i]x[i] \) also causal.
3. Suppose \( h[n] \neq 0 \) only for \( 0 \leq n \leq L \) \((h[n] \text{ has length } L + 1)\).

   Suppose \( x[n] \neq 0 \) only for \( 0 \leq n \leq M \) \((x[n] \text{ has length } M + 1)\).

   Then \( y[n] \neq 0 \) only for \( 0 \leq n \leq L + M \) \((y[n] \text{ has length } L + M + 1)\).

**Note:** \( \text{Length}[y[n]]=\text{Length}[h[n]]+\text{Length}[x[n]]-1 \).

**Note:** \( y[0] = h[0]x[0]; \ y[L+M] = h[L]x[M]; \ x[n] * \delta[n-D] = x[n-D] \).

4. \( x[n] \rightarrow \boxed{h_1[n]} \rightarrow \boxed{h_2[n]} \rightarrow y[n] \) (cascade connection)
   
   Equivalent to: \( x[n] \rightarrow \boxed{h_1[n] * h_2[n]} \rightarrow y[n] \).

5. \( x[n] \rightarrow \boxed{\neg \neg h_1[n]} \rightarrow \boxed{\neg \neg h_2[n]} \rightarrow \bigoplus \rightarrow y[n] \) (parallel connection)
   
   Equivalent to: \( x[n] \rightarrow \boxed{h_1[n] + h_2[n]} \rightarrow y[n] \).

**MA:** \( y[n] = b_0x[n] + b_1x[n-1] + \ldots + b_qx[n-q] \) (Moving Average)

**Huh?** Present output=weighted average of \( q \) most recent\ inputs.

**Note:** Equivalent to \( y[n] = b[n] * x[n] \) where \( b[k] = b_k, 0 \leq k \leq q \).

**FIR:** Finite Impulse Response \( \Leftrightarrow h[n] \text{ has finite duration.} \)

**EX:** Any MA system is also an FIR system, and vice-versa.

**IIR:** Infinite Impulse Response \( \Leftrightarrow h[n] \text{ not finite duration.} \)

**EX:** \( h[n] = a^n u[n] = a^n \) for \( n \geq 0 \) and \( |a| < 1 \) is stable and IIR.
RECURSIVE COMPUTATION OF IMPULSE RESPONSE

Goal: Compute impulse response \( h[n] \) of system \( y[n] - \frac{1}{2}y[n-1] = 3x[n] \).

Sol’n: Compute recursively \( h[n] - \frac{1}{2}h[n-1] = 3\delta[n] = 0 \) if \( n > 0 \).

\( n=0: \) \( h[0] - \frac{1}{2}h[-1] = 3\delta[0] \rightarrow h[0] - \frac{1}{2}(0) = 3(1) \rightarrow h[0] = 3. \)

\( n=1: \) \( h[1] - \frac{1}{2}h[0] = 3\delta[1] \rightarrow h[1] - \frac{1}{2}(3) = 3(0) \rightarrow h[1] = \frac{3}{2}. \)

\( n=2: \) \( h[2] - \frac{1}{2}h[1] = 3\delta[2] \rightarrow h[2] - \frac{1}{2}(\frac{3}{2}) = 3(0) \rightarrow h[2] = \frac{3}{4}. \)

\( n=3: \) \( h[3] - \frac{1}{2}h[2] = 3\delta[3] \rightarrow h[3] - \frac{1}{2}(\frac{3}{4}) = 3(0) \rightarrow h[3] = \frac{3}{8}. \)

\( h[n] = 3(\frac{1}{2})^n u[n] = 3(\frac{1}{2})^n \) for \( n \geq 0 \). Geometric signal.

BIBO (BOUNDED INPUT → BOUNDED OUTPUT) STABILITY

Goal: Determine whether an LTI system is BIBO stable from its \( h[n] \).

EX #1: \( h[n] = \{2, 3, -4\} \rightarrow \sum|h[n]| = |2| + |3| + |-4| = 9 < \infty \rightarrow \text{BIBO stable}. \)

EX #2: \( h[n] = (-\frac{1}{2})^n u[n] \rightarrow \sum|h[n]| = \sum|(-\frac{1}{2})^n| = \frac{1}{1-0.5} < \infty \rightarrow \text{BIBO stable}. \)

EX #3: \( h[n] = (\frac{-1}{n+1})^n u[n] \rightarrow \sum|h[n]| = \sum\frac{1}{n+1}u[n] \rightarrow \infty \rightarrow \text{NOT BIBO stable}. \)

Note: \( \sum\frac{(-1)^n}{n+1}u[n] = \log 2 \) but \( \sum|\frac{-1}{n+1}|u[n] = \sum\frac{1}{n+1}u[n] \rightarrow \infty \)

so absolute summability vs. summability matters for BIBO stability!

CONVOLUTION OF TWO FINITE SIGNALS

Goal: Compute \( \{1, 2, 3\} \ast \{4, 5, 6, 7\} = \{4, 13, 28, 34, 32, 21\} \).

\( y[0] = h[0]x[0] = (1)(4) = 04. \)


Note: Length \( y[n]=\text{Length } h[n]+\text{Length } x[n] - 1 = 3 + 4 - 1 = 6. \)

CONVOLUTION OF FINITE AND INFINITE SIGNALS

Goal: Compute \( \{2, -1, 3\} \ast (\frac{1}{2})^n u[n] = 2\delta[n] + 3(\frac{1}{2})^{n-2}u[n-2] \).

Sol’n: \( \{2, -1, 3\} \ast (\frac{1}{2})^n u[n] = (2\delta[n]-\delta[n-1]^3\delta[n-2]) \ast (\frac{1}{2})^n u[n] = 

2(\frac{1}{2})^n u[n] - 1(\frac{1}{2})^{n-1}u[n-1] + 3(\frac{1}{2})^{n-2}u[n-2] = 2\delta[n] + 3(\frac{1}{2})^{n-2}u[n-2]. \)

Note: \( \{2, -1\} \ast (\frac{1}{2})^n u[n] = 2\delta[n]. \) So \( x[n] \rightarrow |(\frac{1}{2})^n u[n]| \rightarrow |\{2, -1\}| \rightarrow 2x[n]. \)
MA: \( y(n) = b_0 x(n) + b_1 x(n-1) + \ldots + b_q x(n-q) \) (Moving Average)

Huh? Present output = weighted average of \( q \) most recent inputs.

Note: Equivalent to \( y(n) = b(n) \ast x(n) \) where \( b(k) = b_k, 0 \leq k \leq q \).

Why? So why bother? Because now can have initial conditions.

AR: \( y(n) + a_1 y(n-1) + \ldots + a_p y(n-p) = x(n) \) (AutoRegression)

Huh? Present output = weighted sum of \( p \) most recent outputs.

Note: Compute \( y(n) \) recursively from its \( p \) most recent values.

ARMA: \[ \sum_{i=0}^{p} a_i y(n-i) = \sum_{i=0}^{q} b_i x(n-i) \] (Combine AR and MA → ARMA).

PROPERTIES OF DIFFERENCE EQUATIONS

1. Clearly represent LTI system. Now allow initial conditions.
2. MA model ⇔ FIR (Finite Impulse Response) filter:
   For MA model, \( h(n) \) has finite length \( q + 1 \).
3. AR model ⇔ IIR (Infinite Impulse Response) filter:
   For AR model, \( h(n) \) does not have finite length.
   But can implement using finite set of coefficients \( a_i \).
4. Analogous to differential equation in continuous time:
   \( \left( \frac{d^p}{dt^p} + a_1 \frac{d^{p-1}}{dt^{p-1}} + \ldots + a_p \right) y(t) = \left( \frac{d^q}{dt^q} + b_1 \frac{d^{q-1}}{dt^{q-1}} + \ldots + b_q \right) x(t) \).
   Note: Coefficients \( a_i \) and \( b_i \) are not directly analogous here.
   Note: Time-invariant in both cases since coefficients indpt of time.
5. SKIP Section 2.4.3 in text: 1-sided z-transform is much easier.
   READ Section 2.5 in text: signal flow graph implementations.

DIFFERENCE EQUATIONS VS. \( h(n) \)

Advantages of Difference Equations:

1. Analogous to differential equation in continuous time.
   Often can write them down from model of physical system.
2. Can incorporate nonzero initial conditions.
3. AR model may be much more efficient implementation of system.

Disadvantages of Difference Equations:

1. There may not be a difference equation description of system!
   Example: \( h(n) = \frac{1}{n}, n > 0 \) has no difference equation model.
   Recall: \( h(t) = \delta(t-1) \) (delay) has no differential equation model.
2. May only want output \( y(N) \) for single time \( N \) (e.g., end of year).
   Then convolution \( y(N) = \sum h(i) x(N-i) \) more efficient computation.