Section 2.2 DT System properties

$$x(n) \longrightarrow \boxed{H} \longrightarrow y(n) = H[x(n)]$$

input — system — output

1) **Linear**: if $x_1 \to y_1$, $x_2 \to y_2$

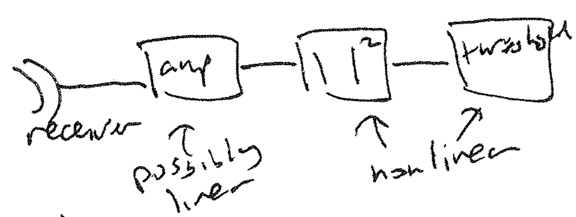
then $x_3 = a x_1 + b x_2 \Rightarrow y_3 = a y_1 + b y_2$

scale/add input $\to$ scaled/added outputs

ex nonlinear: threshold, squaring

2) **shift-invariant or time-invariant**

signal $\longrightarrow$



delay input = delay output

if $x(n) \to y(n)$, then $x(n-k) \to y(n-k)$

Causal: output depends on past & current inputs

$$y(n) = F(x(n), x(n-1), \dots)$$

BIBO stable: bounded input → bounded output

i.e. if $|x(n)| \le M_x < \infty \quad \forall n$

then $|y(n)| < M_y < \infty \quad \forall n$

concept: impulse response $h[n]$. Response to impulse $\delta[n]$

★ thought experiment: hit a bell w/ a hammer.

which of these properties does it exhibit

Convolution $\qquad x \to \boxed{h} \to y$

Key result for LTI systems
Linear
Time-invariant

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

"shorthand"

What this says: Knowing the impulse response, and knowing the system is LTI, we can calculate response to any input.

kind of remarkable.

We'll look at this ~~two~~ 3 ways:
- graphical "derivation" for insight
- calculation method
- math derivation

First, note the sum above is $\sum_{-\infty}^{\infty}$

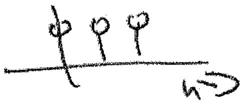
→ if $h(n)$ really non-zero for ∞ time we can't store on computer (IIR)

→ if $h(n)$ is finite length (FIR), we can do calculations

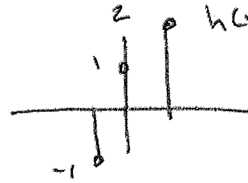
(2)

Graphical "derivation" of convolution – not ~~good for~~ a good practical calculation method, but intuitive.

say, we know (1) $x(n)$



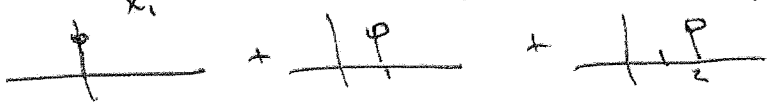
(2) $h(n)$



(3) system is LTI

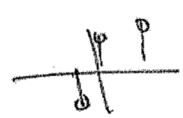
First, write our input as sum of impulses (by linearity)

$x =$ x_1 + x_2 + x_3



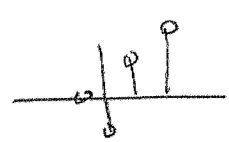
then
→ $h(n)$ is response to $\delta(n)$ (first signal
→ by time-invariance, x_2 and x_3 ⇒ delayed versions of h
→ by linearity, we can scale each term by the amplitude of (x_1, x_2, x_3)
→ by linearity, we can add results

$x_1 \longrightarrow y_1 = h(n)$



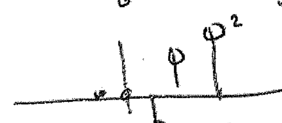
$x(0)\, h(n)$

$x_2 \longrightarrow y_2$



$x(1)\, h(n-1)$

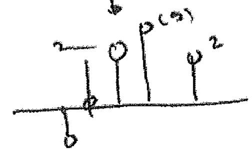
$x_3 \longrightarrow y_3$



$x(2)\, h(n-2)$

add

$y(n) =$



$\sum_{k=0}^{2} x(k)\, h(n-k)$

The above is not a great practical way to compute $x * h$

Q. Why not? A: requires computing each term + storing them all simultaneously

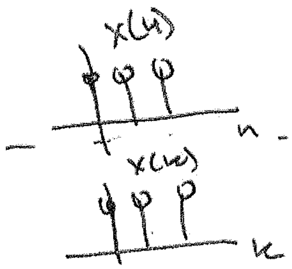
<u>Better/practical</u>: trust the equation

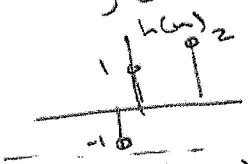
$$y(n) = \sum_{k=-\infty}^{\infty} x(k)\, h(n-k)$$

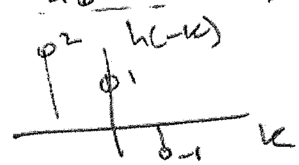
↑ sum is over k ↑ $-k$ says → flip h

<u>algorithm</u>:

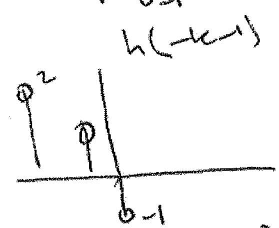
1. put $x(n)$ on k axis
2. <u>flip</u> $h(k)$ to $h(-k)$
3. <u>shift</u> $h(-k)$ by n, for each n
3. multiply $x(k)\, h(n-k)$
4. add: $y(n) = \sum x(k)\, \cancel{h}(n-k)$



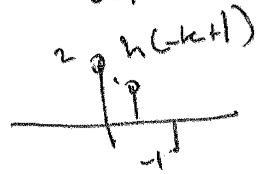




$y(0) = (1)(1) + (1)(-1) = 0$



$y(-1) = (1)(-1) = -1$



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

$y(2) = 1 \cdot 2 + 1 \cdot 1 = 3$

$y(3) = 1 \cdot 2 = 2$

~~Conv~~

More formal derivation: see PPT / PDF / textbook

Convolution properties:

1) $x * h = h * x$ commutative

2) $[x * h_1] * h_2 = x * [h_1 * h_2]$ associative

3) $x * [h_1 + h_2] = x * h_1 + x * h_2$ distributive

CCDE's – Constant Coefficient Difference Equations
(P+M 2.4)

For DT systems, we can do a lot by simply delaying, scaling + summing signals.

Most general form: (for causal)

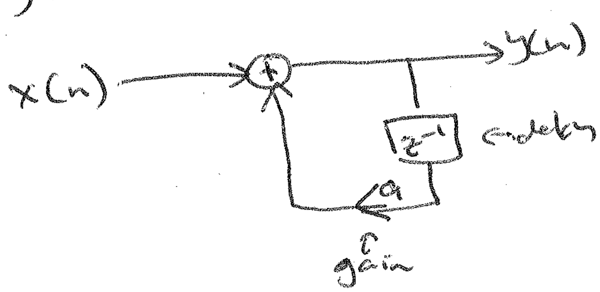
$$\underbrace{y(n)}_{\text{current output}} = \underbrace{-\sum_{k=1}^{N} a_k\, y(n-k)}_{\text{scaled past outputs – feedback}} + \underbrace{\sum_{k=0}^{N} b_k\, x(n-k)}_{\text{scaled current } (k=0) \text{ and past inputs}}$$

This equation leads to linear (no powers), time-invariant (constant a_k, b_k), causal systems.

See PPT + P+M Fig 2.5.2 for most general form – can be rewritten for speed

Simple feedback example

Before, we asked: how can we implement an IIR system on the computer? Answer is: recursion!



$$y(n) = x(n) + a\,y(n-1)$$

Say we put in impulse:

$x(n) = \delta(n)$

$y(n) = 0, \quad n < 0$

$y(0) = 1$

$y(1) = a\,y(0) + 0 = a$

$y(2) = a^2$

$y(n) = a^n$, etc.

note: since $x(n) = \delta(n)$, actually $y(n) = h(n) \rightarrow$ impulse response!
IIR

$a = 1 \rightarrow h(n) = u(n)$ (step)

$a = -1 \rightarrow$ toggles $(1, -1, 1, -1, \ldots)$

$|a| < 1$ decays, $|a| > 1$ grows.

$\triangle$ feedback leads to infinite impulse response
Thus in our general eqn, FIR means $a_k = 0 \ \forall k > 0$
else, it's IIR

Q) how to solve/analyze CCDE's? see PPT slides for discussion

A key tool is the <u>z-transform</u>