Humor with Signals

A conversation between Pepito and Arturita

Arturita

Did you understand the Fourier series?

Pepito

I haven't watched it yet...Is it available on streaming?

Arturita

Thank you :)

*Internet meme, source unknown

Lecture 4: FIR filters

Juan Diego Cardenas-Cartagena, M.Sc.
(j.d.cardenas.cartagena@rug.nl)

Signals and Systems
1B - 2024/2025

Preliminaries

- The first lab assignment is available now. And its deadline is on Friday, December 6, at 17:30.
- The answers for Tutorials 1 and 2 are available in Brightspace. We will upload the answers for Tutorial 3 on Friday. However, **it is strongly recommended that you attend the tutorials**, as the TAs will provide individual support and discuss applications of selected topics in AI.
- The Fourier and its transforms cheat sheets is now available in the Useful Resources tab.
- Lab assignment 2 is available now. And its deadline is December 20, 2024 at 17:30.

Overview

Table of Contents

Recap

Shannon sampling theorem:

- A continuous-time signal $x(t)$ with frequencies no higher than f_{max} can be reconstructed exactly from its samples $x[n] = x(nT_s)$ if the samples are taken at a rate $f_s > 2f_{max}$.
- Only works for **bandlimited** signal that has a maximum frequency f_{max}.
- The minimum sampling rate $2f_{max}$ is called the **Nyquist rate**.

Question

Where is the course with respect to the Machine Learning Lifecycle?

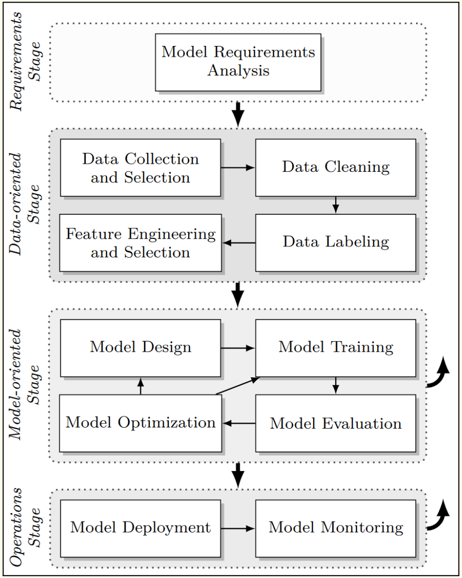
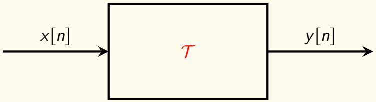


Figure: The Machine Learning Lifecycle, M. Schlegel and K.-U. Sattler, "Management of machine learning lifecycle artifacts: A survey," ACM SIGMOD Record, 2023.

Table of Contents

Filters



A system is an operator $\mathcal{T}$ that transforms signals

$$x[n] \overset{\mathcal{T}}{\longmapsto} y[n] \tag{1}$$

- Filters are specific types of systems.
- **Finite impulse response (FIR)** filters are based on a finite number of values in a discrete input signal.

Note

Terms ‘filter’ and ‘system’ are used interchangeably

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{2}$$

- The **difference equation** defines the filter by showing how y is generated from x.
- Each entry $y[n]$ is the average of three values of $x[n]$.
- Example: Let $x[n] = \{2, 4, 6, 4, 2\}, n = 0, 1, \ldots, 4$. Compute $y[n]$:

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	0	2	4	6	4	2	0	0
$y[n]$									

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{3}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	**0**	**0**	**2**	4	6	4	2	0	0
$y[n]$									

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{4}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{2}$	4	6	4	2	0	0
$y[n]$	$\frac{2}{3}$								

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{5}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	**0**	**2**	**4**	6	4	2	0	0
$y[n]$	$\frac{2}{3}$								

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{6}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	**0**	**2**	**4**	6	4	2	0	0
$y[n]$	$\frac{2}{3}$	2							

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{7}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	0	**2**	**4**	**6**	4	2	0	0
$y[n]$	$\frac{2}{3}$	2							

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{8}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	0	**2**	**4**	**6**	4	2	0	0
$y[n]$	$\frac{2}{3}$	2	4						

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{9}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	0	2	**4**	**6**	**4**	2	0	0
$y[n]$	$\frac{2}{3}$	2	4						

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{10}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	0	2	**4**	**6**	**4**	2	0	0
$y[n]$	$\frac{2}{3}$	2	4	$\frac{14}{3}$					

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{11}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	> 5
$x[n]$	0	0	2	4	**6**	**4**	**2**	0	0
$y[n]$	$\frac{2}{3}$	2	4	$\frac{14}{3}$					

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{12}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	> 5
$x[n]$	0	0	2	4	**6**	**4**	**2**	0	0
$y[n]$	$\frac{2}{3}$	2	4	$\frac{14}{3}$	4				

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{13}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	0	2	4	6	**4**	**2**	**0**	0
$y[n]$	$\frac{2}{3}$	2	4	$\frac{14}{3}$	4				

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{14}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	> 5
$x[n]$	0	0	2	4	6	**4**	**2**	**0**	0
$y[n]$	$\frac{2}{3}$	2	4	$\frac{14}{3}$	4	2			

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{15}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	0	2	4	6	4	**2**	**0**	**0**
$y[n]$	$\frac{2}{3}$	2	4	$\frac{14}{3}$	4	2			

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{16}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	0	2	4	6	4	**2**	**0**	**0**
$y[n]$	$\frac{2}{3}$	2	4	$\frac{14}{3}$	4	2	$\frac{2}{3}$		

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{17}$$

- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	0	2	4	6	4	2	**0**	**0**
$y[n]$	$\frac{2}{3}$	2	4	$\frac{14}{3}$	4	2	$\frac{2}{3}$		

Running average (moving average)

The three-point running average is an example of a FIR filter

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2]) \tag{18}$$

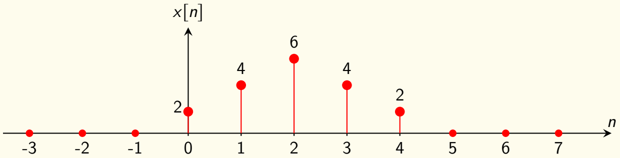
- The **difference equation** defines the filter by showing how y is generated from x
- Each entry $y[n]$ is the average of three values of $x[n]$, i.e.,

n	-2	-1	0	1	2	3	4	5	>5
$x[n]$	0	0	2	4	6	4	2	**0**	**0**
$y[n]$	$\frac{2}{3}$	2	4	$\frac{14}{3}$	4	2	$\frac{2}{3}$	0	0

Question

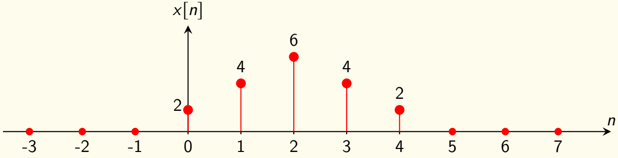
Suppose that $\mathcal{T}$ is a transformation that you want to implement in a low-latency (or real-time) application. And let n be the current timestep, i.e., the present. What are the implications of using $x[n+1], x[n+2], \ldots$?

Running average (moving average)

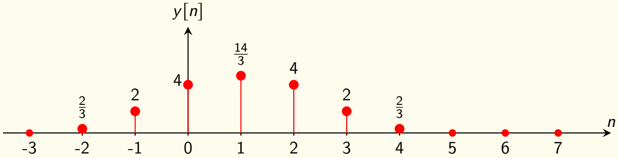


$x[n]$ is a **finite-length** signal with **support** $n \in \{0, 4\}$

Running average (moving average)



$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2])$$



- Output $y[n]$ usually has more non-zero entries than $x[n]$
- Note that y starts before x

FIR filters

In general n is interpreted as time. For value $y[n], x[n]$ is the **present** value of the input.

- Values $x[m]$, $m < n$ are in the past
- Values $x[m]$, $m > n$ are in the future

FIR filters

In general n is interpreted as time. For value $y[n], x[n]$ is the **present** value of the input.

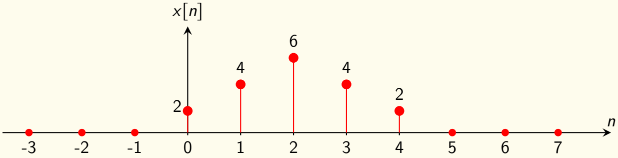
- Values $x[m]$, $m < n$ are in the past
- Values $x[m]$, $m > n$ are in the future

Filter that use future values are called **noncausal**. Otherwise, they are **causal**.

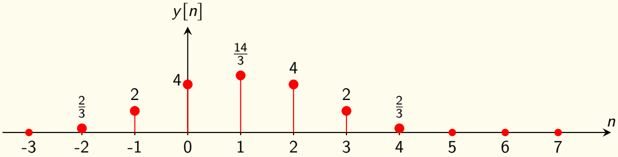
- Causal filters can be implemented in real-time systems.
- Consider the running average filter as an example of a causal filter:

$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n-1] + x[n-2]).$$

Running average (moving average)

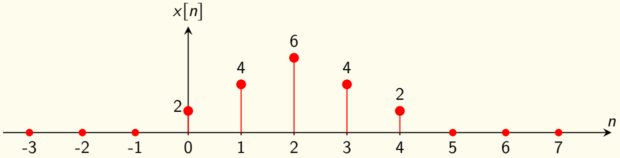


$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n+1] + x[n+2])$$

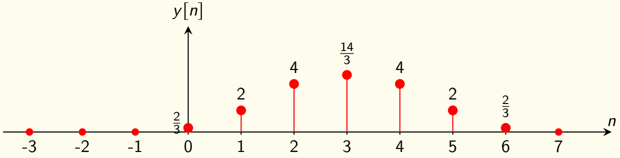


Noncausal filters cannot be implemented in real-time systems

Running average (moving average)



$$y[n] = \mathcal{T}\{x[n]\} = \frac{1}{3}(x[n] + x[n-1] + x[n-2])$$



Causal filters can be implemented in real-time systems

General causal FIR filter

The general form of causal FIR filters is

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] = b_0 x[n] + b_1 x[n-1] + \cdots + b_M x[n-M] \tag{19}$$

General causal FIR filter

The general form of causal FIR filters is

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] = b_0 x[n] + b_1 x[n-1] + \cdots + b_M x[n-M] \tag{19}$$

- Also known as a (weighted) running average filter
  - The 3-point running average has $M = 2$ and $b_k = \frac{1}{3}$
- The filter has **order** M
- The **length** L counts the number of (non-zero) coefficients b_k
  - Typically, $L = M + 1$

Example - 3-point running average

Suppose we have the following input signal

$$x[n] = (1.02)^n + \frac{1}{2}\cos(2\pi n/8 + \pi/4),\ 0 \leqslant n \leqslant 40.$$

And the filter output is

$$y_3[n] = \sum_{k=0}^{2} \frac{1}{3} x[n-k].$$

Example - 7-point running average

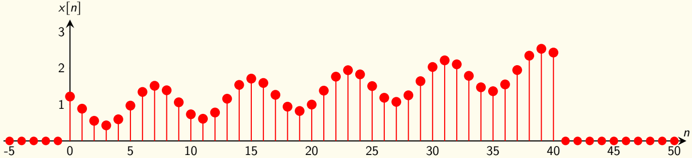


Figure: Input signal $x[n]$

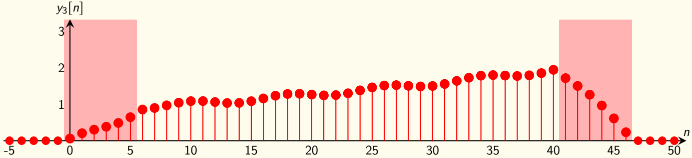


Figure: Output signal $y_3[n]$

Table of Contents

Unit impulse

The Kronecker delta function describes a **unit impulse**

$$\delta[n] \begin{cases} 1, & n = 0, \\ 0, & n \neq 0. \end{cases} \tag{20}$$

Unit impulse

The Kronecker delta function describes a **unit impulse**

$$\delta[n] \begin{cases} 1, & n = 0, \\ 0, & n \neq 0. \end{cases} \tag{20}$$

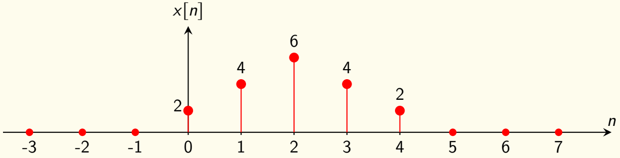
Value 1 at exactly one instant

n	$\dots$	-2	-1	0	1	2	3	4	5	$\dots$
$\delta[n]$	0	0	0	**1**	0	0	0	0	0	0
$\delta[n-2]$	0	0	0	0	0	**1**	0	0	0	0

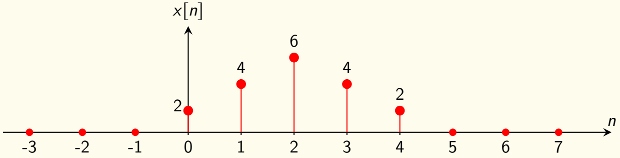
Unit impulse

A discrete-time signal $x[n]$ can be written as a **linear combination of unit impulses**.

Unit impulse - Example

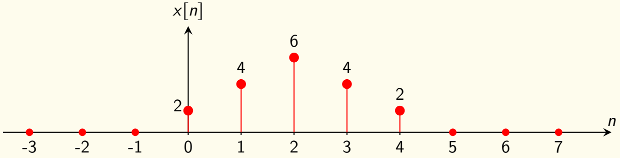


Unit impulse - Example



$$x[n] = 2\delta[n] + 4\delta[n-1] + 6\delta[n-2] + 4\delta[n-3] + 2\delta[n-4] \tag{21}$$

Unit impulse - Example



$$x[n] = 2\delta[n] + 4\delta[n-1] + 6\delta[n-2] + 4\delta[n-3] + 2\delta[n-4] \tag{21}$$

n	$\ldots$	-2	-1	0	1	2	3	4	5	$\ldots$
$2\delta[n]$	0	0	0	**2**	0	0	0	0	0	0
$4\delta[n-1]$	0	0	0	0	**4**	0	0	0	0	0
$6\delta[n-2]$	0	0	0	0	0	**6**	0	0	0	0
$4\delta[n-3]$	0	0	0	0	0	0	**4**	0	0	0
$2\delta[n-4]$	0	0	0	0	0	0	0	**2**	0	0
$x[n]$	0	0	0	**2**	**4**	**6**	**4**	**2**	0	0

Unit impulse response

The unit impulse response $h[n]$ of a filter is its response to the unit impulse $\delta[n]$

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] = b_0 x[n] + b_1 x[n-1] + \cdots + b_M x[n-M] \tag{22}$$

$$h[n] = \sum_{k=0}^{M} b_k \delta[n-k] = b_n, \text{ for } n = 0, 1, \ldots, M \tag{23}$$

Unit impulse response

The unit impulse response $h[n]$ of a filter is its response to the unit impulse $\delta[n]$

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] = b_0 x[n] + b_1 x[n-1] + \cdots + b_M x[n-M] \tag{22}$$

$$h[n] = \sum_{k=0}^{M} b_k \delta[n-k] = b_n, \text{ for } n = 0, 1, \ldots, M \tag{23}$$

$$y[n] = \sum_{k=0}^{M} h[n] x[n-k] \tag{24}$$

Unit impulse response

The unit impulse response $h[n]$ of a filter is its response to the unit impulse $\delta[n]$

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] = b_0 x[n] + b_1 x[n-1] + \cdots + b_M x[n-M] \tag{22}$$

$$h[n] = \sum_{k=0}^{M} b_k \delta[n-k] = b_n, \text{ for } n = 0, 1, \ldots, M \tag{23}$$

$$y[n] = \sum_{k=0}^{M} h[n] x[n-k] \tag{24}$$

That is, the output, $y[n]$, is the **convolution sum** of $x[n]$ and $h[n]$,

$$y[n] = h[n] * x[n] \tag{25}$$

Example

Suppose we want to find the response $y[n]$ for filter $h[n] = \{3, -1, 2, 1\}$ of input $x[n] = \{2, 4, 6, 4, 2\}$

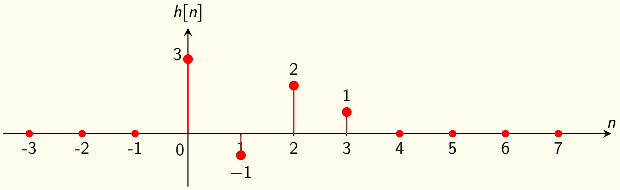


Figure: Filter $h[n]$

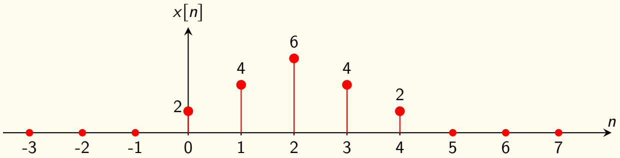


Figure: Input $x[n]$

Example

Suppose we want to find the response $y[n]$ for filter $h[n] = \{3, -1, 2, 1\}$ of input $x[n] = \{2, 4, 6, 4, 2\}$

n	0	1	2	3	4	5	6	7	...
$h[n]$	3	-1	2	1					
$x[n]$	2	4	6	4	2	0	0	0	0
$h[0]x[n-0]$	6	12	18	12	6	0	0	0	0
$h[1]x[n-1]$		-2	-4	-6	-4	-2	0	0	0
$h[2]x[n-2]$			4	8	12	8	4	0	0
$h[3]x[n-3]$				2	4	6	6	2	0
$y[n]$	6	10	18	16	18	12	8	2	0

FIR filters as convolutions

A FIR filter is a convolution with a finite length impulse response.

FIR filters as convolutions

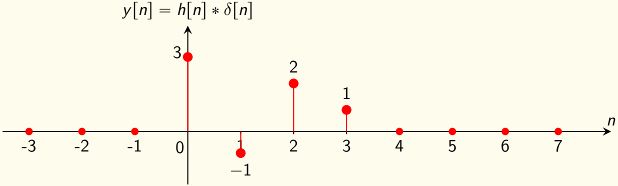
A FIR filter is a convolution with a finite length impulse response.

- The input may still be infinite in length
- E.g. the unit-step signal $u[n]$

$$u[n] = \begin{cases} 1, & \text{if } n \geqslant 0, \\ 0, & \text{if } n < 0 \end{cases} \tag{26}$$

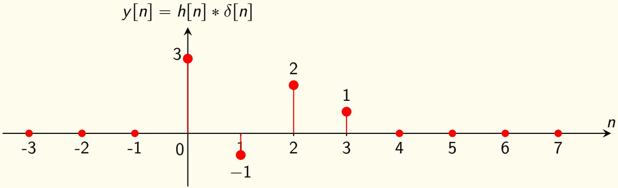
Unit-step signal

- The unit impulse $\delta[n]$ recovers the filter coefficients b_n

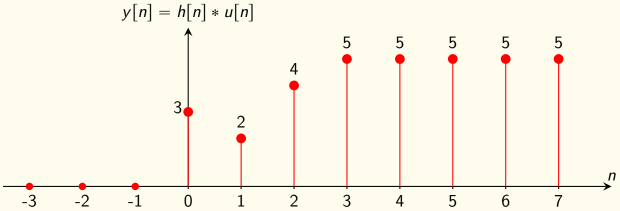


Unit-step signal

- The unit impulse $\delta[n]$ recovers the filter coefficients b_n



- The unit step $u[n]$ returns the sum of the filter coefficients $\sum_{k=0}^{n} b_k$



FIR filters

The general form of a FIR filter is

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] \tag{27}$$

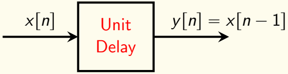
FIR filters

The general form of a FIR filter is

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] \tag{27}$$

This can be represented using three basic building blocks

- **Delaying** the input signal



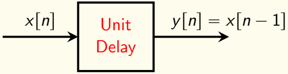
FIR filters

The general form of a FIR filter is

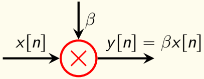
$$y[n] = \sum_{k=0}^{M} b_k x[n-k] \tag{27}$$

This can be represented using three basic building blocks

- **Delaying** the input signal



- **Multiplying** the input by filter coefficients



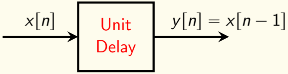
FIR filters

The general form of a FIR filter is

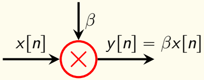
$$y[n] = \sum_{k=0}^{M} b_k x[n-k] \tag{27}$$

This can be represented using three basic building blocks

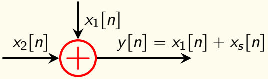
- **Delaying** the input signal



- **Multiplying** the input by filter coefficients

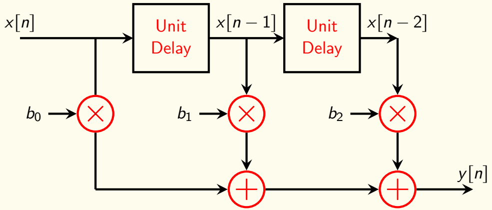


- **Adding** signals together



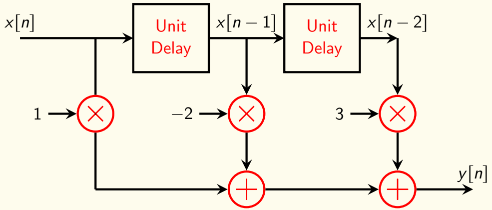
Example - Direct form

$$y[n] = h[n] * x[n] = x[n] - 2x[n-1] + 3x[n-2]$$



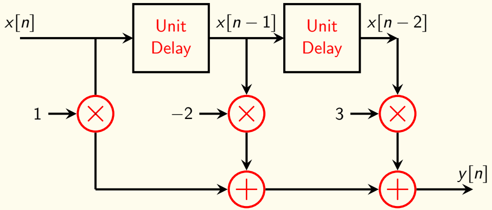
Example - Direct form

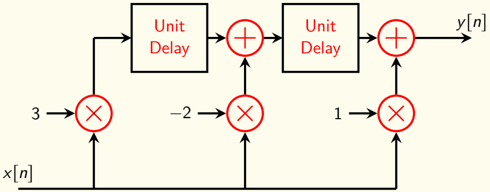
$$y[n] = h[n] * x[n] = x[n] - 2x[n-1] + 3x[n-2]$$



Example - Direct form

$$y[n] = h[n] * x[n] = x[n] - 2x[n-1] + 3x[n-2]$$





Break!

See you at ____________________________

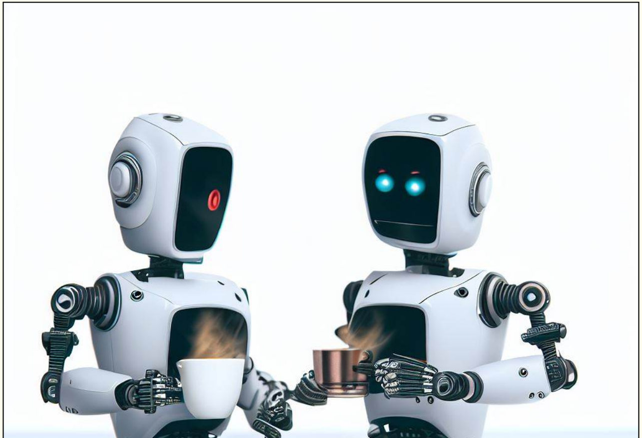
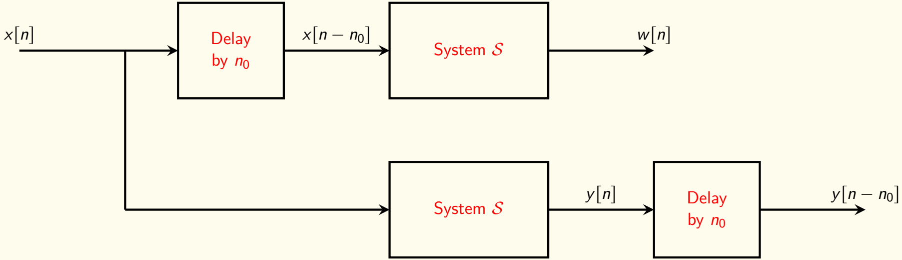


Table of Contents

Time invariance

A discrete-time system is time-invariant if time-delaying the input by n_0 will result in time-delaying the output by the same amount n_0



For time-invariant systems, $w[n] = y[n - n_0]$

Example - Time invariant

Consider the system $\mathcal{S}$ below

$$x[n] \overset{\mathcal{S}}{\mapsto} (x[n])^2$$

Example - Time invariant

Consider the system $\mathcal{S}$ below

$$x[n] \overset{\mathcal{S}}{\mapsto} (x[n])^2$$

This system is time-invariant because

$$x[n] \overset{\mathcal{D}}{\mapsto} x[n - n_0] \overset{\mathcal{S}}{\mapsto} (x[n - n_0])^2$$

Example - Time invariant

Consider the system $\mathcal{S}$ below

$$x[n] \overset{\mathcal{S}}{\mapsto} (x[n])^2$$

This system is time-invariant because

$$x[n] \overset{\mathcal{D}}{\mapsto} x[n-n_0] \overset{\mathcal{S}}{\mapsto} (x[n-n_0])^2$$

$$x[n] \overset{\mathcal{S}}{\mapsto} (x[n])^2 \overset{\mathcal{D}}{\mapsto} (x[n-n_0])^2$$

Example - Not time invariant

Consider the system $\mathcal{F}$ below

$$x[n] \overset{\mathcal{F}}{\mapsto} n \cdot x[n]$$

Example - Not time invariant

Consider the system $\mathcal{F}$ below

$$x[n] \overset{\mathcal{F}}{\mapsto} n \cdot x[n]$$

This system is not time-invariant because

$$x[n] \overset{\mathcal{D}}{\mapsto} x[n - n_0] \overset{\mathcal{F}}{\mapsto} n \cdot x[n - n_0]$$

Example - Not time invariant

Consider the system $\mathcal{F}$ below

$$x[n] \overset{\mathcal{F}}{\mapsto} n \cdot x[n]$$

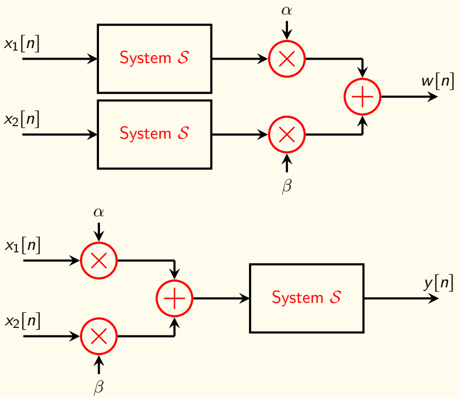
This system is not time-invariant because

$$x[n] \overset{\mathcal{D}}{\mapsto} x[n - n_0] \overset{\mathcal{F}}{\mapsto} n \cdot x[n - n_0]$$

$$x[n] \overset{\mathcal{F}}{\mapsto} n \cdot x[n] \overset{\mathcal{D}}{\mapsto} (\mathbf{n} - \mathbf{n_0}) \cdot x[n - n_0]$$

Linearity

A discrete-time system is **linear** if a linear combination of inputs will result in the same linear combination of outputs



For linear systems, $w[n] = y[n]$

Example - Linear

Consider the system $\mathcal{F}$ below

$$x[n] \stackrel{\mathcal{F}}{\mapsto} n \cdot x[n]$$

Example - Linear

Consider the system $\mathcal{F}$ below

$$x[n] \overset{\mathcal{F}}{\mapsto} n \cdot x[n]$$

This system is linear because

$$\left.\begin{array}{l} x_1[n] \overset{\alpha}{\mapsto} \alpha x_1[n] \\ x_2[n] \overset{\beta}{\mapsto} \beta x_2[n] \end{array}\right\} \overset{+}{\mapsto} \alpha x_1[n] + \beta x_2[n] \overset{\mathcal{F}}{\mapsto} \alpha n x_1[n] + \beta n x_2[n]$$

Example - Linear

Consider the system $\mathcal{F}$ below

$$x[n] \overset{\mathcal{F}}{\mapsto} n \cdot x[n]$$

This system is linear because

$$\left.\begin{array}{l} x_1[n] \overset{\alpha}{\mapsto} \alpha x_1[n] \\ x_2[n] \overset{\beta}{\mapsto} \beta x_2[n] \end{array}\right\} \overset{+}{\mapsto} \alpha x_1[n] + \beta x_2[n] \overset{\mathcal{F}}{\mapsto} \alpha n x_1[n] + \beta n x_2[n]$$

$$\left.\begin{array}{l} x_1[n] \overset{\mathcal{F}}{\mapsto} n x_1[n] \overset{\alpha}{\mapsto} \alpha n x_1[n] \\ x_2[n] \overset{\mathcal{F}}{\mapsto} n x_2[n] \overset{\beta}{\mapsto} \beta n x_2[n] \end{array}\right\} \overset{+}{\mapsto} \alpha n x_1[n] + \beta n x_2[n]$$

Example - Not linear

Consider the system $\mathcal{S}$ below

$$x[n] \overset{\mathcal{S}}{\mapsto} (x[n])^2$$

Example - Not linear

Consider the system $\mathcal{S}$ below

$$x[n] \overset{\mathcal{S}}{\mapsto} (x[n])^2$$

This system is not linear because

$$\alpha x_1[n] + \beta x_2[n] \overset{\mathcal{S}}{\mapsto} \alpha^2 (x_1[n])^2 + 2\alpha\beta x_1[n] x_2[n] + \beta^2 (x_2[n])^2$$

Example - Not linear

Consider the system $\mathcal{S}$ below

$$x[n] \stackrel{\mathcal{S}}{\mapsto} (x[n])^2$$

This system is not linear because

$$\alpha x_1[n] + \beta x_2[n] \stackrel{\mathcal{S}}{\mapsto} \alpha^2 (x_1[n])^2 + 2\alpha\beta x_1[n] x_2[n] + \beta^2 (x_2[n])^2$$

$$\left.\begin{array}{l} x_1[n] \stackrel{\mathcal{S}}{\mapsto} (x_1[n])^2 \stackrel{\alpha}{\mapsto} \alpha n (x_1[n])^2 \\ x_2[n] \stackrel{\mathcal{S}}{\mapsto} (x_2[n])^2 \stackrel{\beta}{\mapsto} \beta n (x_2[n])^2 \end{array}\right\} \stackrel{+}{\mapsto} \alpha (x_1[n])^2 + \beta (x_2[n])^2$$

Linear time-invariant systems

Linear time-invariant (LTI) systems are both linear and time-invariant

- All FIR filters are LTI systems
- Not all LTI systems are FIR filters
- Impulse response $h[n]$ completely describes a LTI system.

An LTI system can always be represented as

$$y[n] = h[n] * x[n] \tag{28}$$

Properties of convolutions

$$y[n] = h[n] * x[n] = \sum_{k=-\infty}^{\infty} h[n]x[n-k] \tag{29}$$

The convolution has a few useful properties

- Commutative

$$x[n] * h[n] = h[n] * x[n] \tag{30}$$

- Identity

$$x[n] * \delta[n] = \delta[n] * x[n] = x[n] \tag{31}$$

- Associative

$$(x[n] * h_1[n]) * h_2[n] = x[n] * (h_1[n] * h_2[n]) \tag{32}$$

- Distributive

$$(x_1[n] + x_2[n]) * h[n] = x_1[n] * h[n] + x_2[n] * h[n] \tag{33}$$

Cascaded LTI systems

Cascades connect multiple systems together

- The output of one system is the input for the next system

For a cascade of LTI systems, the order of the system does not matter

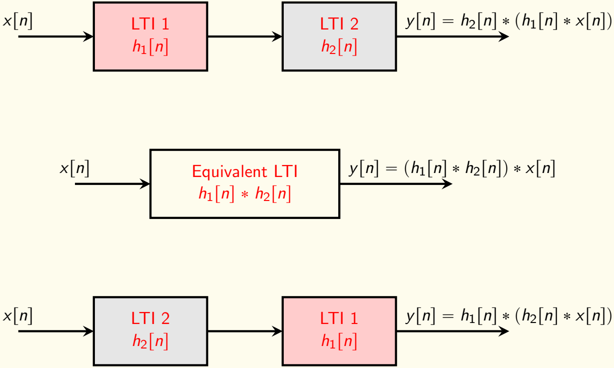


Table of Contents

Convolution in 2D

Filters can also be applied to 2-dimensional signals

- E.g. images
- The impulse reponse h becomes a two-dimensional sequence

2D convolution

$$y[n,m] = \sum_{i=-k}^{k} \sum_{j=-k}^{k} h[i,j]x[n-i,m-j] \tag{34}$$

2D correlator

$$y[n,m] = \sum_{i=-k}^{k} \sum_{j=-k}^{k} h[i,j]x[n+i,m+j] \tag{35}$$

Example

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} 45 & 82 & 61 & 44 & 68 \\ 28 & 3 & 35 & 1 & 12 \\ 98 & 96 & 31 & 81 & 94 \\ 18 & 34 & 12 & 66 & 63 \\ 65 & 76 & 7 & 11 & 86 \end{bmatrix} = \begin{bmatrix} -70 & -219 & -83 & -46 & -216 \\ 34 & 229 & -44 & 168 & 115 \\ -250 & -218 & 100 & -132 & -220 \\ 125 & 66 & 90 & -97 & -6 \\ -166 & -198 & 71 & 115 & -270 \end{bmatrix}$$

Example

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} 45 & 82 & 61 & 44 & 68 \\ 28 & 3 & 35 & 1 & 12 \\ 98 & 96 & 31 & 81 & 94 \\ 18 & 34 & 12 & 66 & 63 \\ 65 & 76 & 7 & 11 & 86 \end{bmatrix} = \begin{bmatrix} -70 & -219 & -83 & -46 & -216 \\ 34 & \mathbf{229} & -44 & 168 & 115 \\ -250 & -218 & 100 & -132 & -220 \\ 125 & 66 & 90 & -97 & -6 \\ -166 & -198 & 71 & 115 & -270 \end{bmatrix}$$

Example

$$\begin{bmatrix} \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{1} & \textbf{-4} & \mathbf{1} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} \end{bmatrix} * \begin{bmatrix} \mathbf{45} & \mathbf{82} & \mathbf{61} & 44 & 68 \\ \mathbf{28} & \mathbf{3} & \mathbf{35} & 1 & 12 \\ \mathbf{98} & \mathbf{96} & \mathbf{31} & 81 & 94 \\ 18 & 34 & 12 & 66 & 63 \\ 65 & 76 & 7 & 11 & 86 \end{bmatrix} = \begin{bmatrix} -70 & -219 & -83 & -46 & -216 \\ 34 & \mathbf{229} & -44 & 168 & 115 \\ -250 & -218 & 100 & -132 & -220 \\ 125 & 66 & 90 & -97 & -6 \\ -166 & -198 & 71 & 115 & -270 \end{bmatrix}$$

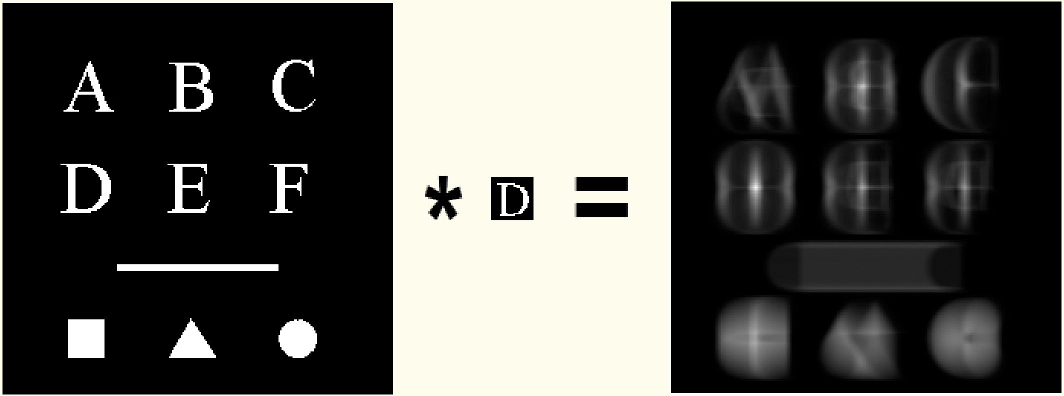
Example

$$\begin{bmatrix} \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{1} & \mathbf{-4} & \mathbf{1} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} \end{bmatrix} * \begin{bmatrix} \mathbf{45} & \mathbf{82} & \mathbf{61} & 44 & 68 \\ \mathbf{28} & \mathbf{3} & \mathbf{35} & 1 & 12 \\ \mathbf{98} & \mathbf{96} & \mathbf{31} & 81 & 94 \\ 18 & 34 & 12 & 66 & 63 \\ 65 & 76 & 7 & 11 & 86 \end{bmatrix} = \begin{bmatrix} -70 & -219 & -83 & -46 & -216 \\ 34 & \mathbf{229} & -44 & 168 & 115 \\ -250 & -218 & 100 & -132 & -220 \\ 125 & 66 & 90 & -97 & -6 \\ -166 & -198 & 71 & 115 & -270 \end{bmatrix}$$

$$\begin{aligned} y[1,1] =& h[0,0]x[0,0] + h[0,1]x[0,1] + h[0,2]x[0,2] + \\ & h[1,0]x[1,0] + h[1,1]x[1,1] + h[1,2]x[1,2] + \\ & h[2,0]x[2,0] + h[2,1]x[2,1] + h[2,2]x[2,2] \\ =& 0 \cdot 45 + 1 \cdot 82 + 0 \cdot 61 + \\ & 1 \cdot 28 - 4 \cdot 3 + 1 \cdot 35 + \\ & 0 \cdot 98 + 1 \cdot 96 + 0 \cdot 31 \\ =& 229 \end{aligned}$$

Note that this filter yields a high value for local minima.

2D correlator



Brighter white indicates higher correlation between input x and input response h

Sobel edge detection

$$h_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \; h_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

For two-dimensional input $x[n, m]$, Sobel edge detection results in

$$y[n, m] = \sqrt{(h_x * x)[n, m]^2 + (h_y * x)[n, m]^2} \tag{36}$$

Table of Contents

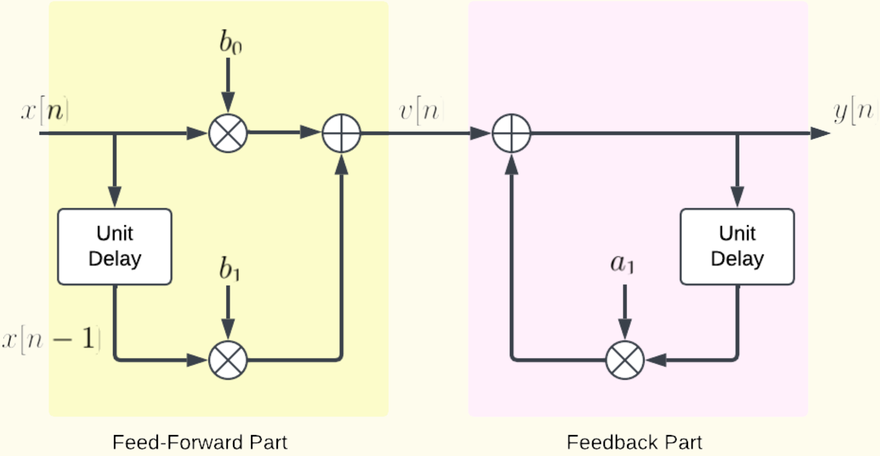
Note that the output of FIR filters only depends on finite samples from the input signal. What if we want to design an LTI filters whose output depends on previous outputs as well?

Infinite Impulse Response (IIR) Filters are LTI filters whose output depends the input signal and previous outputs via a **feedback loop**. These filters are characterized with the difference equation

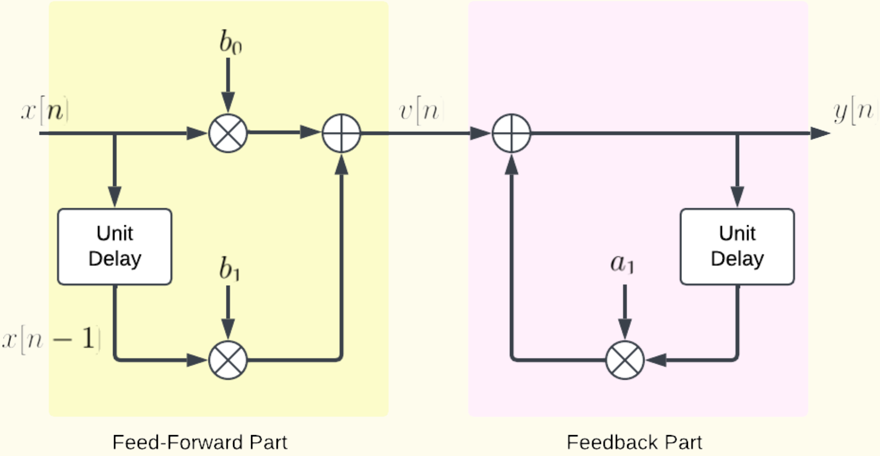
$$y[n] = \sum_{k=0}^{N} b_k x[n-k] + \sum_{l=1}^{M} a_l y[n-l], \tag{37}$$

where a_l and b_k are the filter coefficients. Specifically, a_l represents the coefficients of the **feedback loop**, and b_k represents the coefficients of the **feedforward loop**.

Consider the following IIR filter:



Consider the following IIR filter:



The output of this filter is given by

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

Question

What is the advantage of this feedback system? Also, what could go wrong with IIR filters and their feedback loop?

Question

What is the advantage of this feedback system? Also, what could go wrong with IIR filters and their feedback loop?

Initial Conditions

- IIR filters are sensitive to initial conditions in the input signal. We assume that $x[n] = 0$ for $n < n_0$, where n_0 is the starting time.
- Same for the output signal: $y[n] = 0$ for $n < n_0$.

These conditions guarantee that the IIR filter in (37) is LTI.

Consider the following IIR filter:

$$y[n] = 0.8y[n-1] + 5x[n]. \tag{38}$$

With input signal

$$x[n] = 2\delta[n] - 3\delta[n-1] + 2\delta[n-2]. \tag{39}$$

Consider the following IIR filter:

$$y[n] = 0.8y[n-1] + 5x[n]. \tag{38}$$

With input signal

$$x[n] = 2\delta[n] - 3\delta[n-1] + 2\delta[n-2]. \tag{39}$$

Let us compute the output signal $y[n]$ for $n = 0, 1, 2, \ldots.$

$$\begin{aligned}
y[0] &= 0.8y[-1] + 5x[0] = 0.8(0) + 5(2) = 10 \\
y[1] &= 0.8y[0] + 5x[1] = 0.8(10) + 5(-3) = -7 \\
y[2] &= 0.8y[1] + 5x[2] = 0.8(-7) + 5(0) = -5.6 \\
&\ldots
\end{aligned}$$

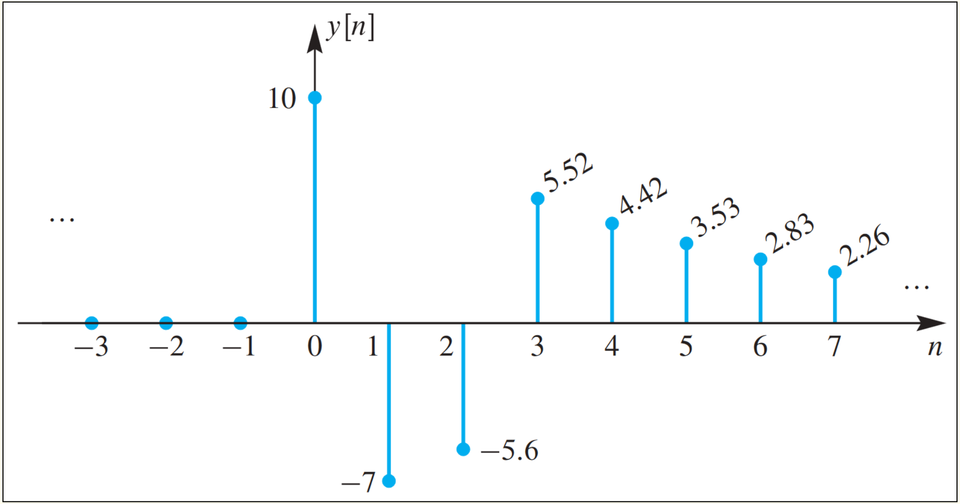


Figure: Output signal $y[n]$ of the IIR filter in (38) with input in (39).

Impulse Response of IIR Filters

Recall

From (28), the output of an LTI system is the convolution of the input signal with the impulse response.

Impulse Response of IIR Filters

Recall

From (28), the output of an LTI system is the convolution of the input signal with the impulse response.

Example: Let us compute the impulse response for the first-order IIR filter given by

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

where $x[n] = \delta[n]$ and $x[n] = 0, y[n] = 0, n < 0$.

Impulse Response of IIR Filters

Recall

From (28), the output of an LTI system is the convolution of the input signal with the impulse response.

Example: Let us compute the impulse response for the first-order IIR filter given by

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

where $x[n] = \delta[n]$ and $x[n] = 0, y[n] = 0, n < 0$. The filter is characterized by the difference equation

$$h[n] = a_1 h[n-1] + b_0 \delta[n].$$

Impulse Response of IIR Filters

Recall

From (28), the output of an LTI system is the convolution of the input signal with the impulse response.

Example: Let us compute the impulse response for the first-order IIR filter given by

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

where $x[n] = \delta[n]$ and $x[n] = 0, y[n] = 0, n < 0$. The filter is characterized by the difference equation

$$h[n] = a_1 h[n-1] + b_0 \delta[n].$$

n	$n < 0$	0	1	2	3	4	...
$\delta[n]$	0	1	0	0	0	0	...
$h[n-1]$	0	0	b_0	$b_0(a_1)$	$b_0(a_1)^2$	$b_0(a_1)^3$	...
$h[n]$	0	b_0	$b_0(a_1)$	$b_0(a_1)^2$	$b_0(a_1)^3$	$b_0(a_1)^4$	...

In other words, the impulse response of the IIR filter in the example is given by

$$h[n] = b_0(a_1)^n u[n].$$

Question

What do you notice about the impulse response of the IIR filters in the example?

- The initial conditions guarantee that the output of the IIR filter begins at $n = 0$.
- Because of the feedback loop, the impulse response of IIR filters can be infinite, even if the input signal is finite.

Unit Response of IIR Filters

Let us now compute the unit response of the first-order IIR filter in the last example,

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

where $x[n] = u[n]$ and $x[n] = 0, y[n] = 0, n < 0$.

Unit Response of IIR Filters

Let us now compute the unit response of the first-order IIR filter in the last example,

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

where $x[n] = u[n]$ and $x[n] = 0, y[n] = 0, n < 0$. The unit response is given by

n	$x[n]$	$y[n]$
$n < 0$	0	0
0	1	b_0
1	1	$a_1(b_0) + b_0$
2	1	$a_1(a_1(b_0) + b_0) + b_0$
3	1	$a_1(a_1^2 + a_1 + 1)(b_0) + b_0$
4	1	$(a_1^4 + a_1^3 + a_1^2 + a_1 + 1)b_0$
$\vdots$	1	$\vdots$

In summary, the unit response of the IIR filter in the example is given by

$$y[n] = b_0(1 + a_1 + a_1^2 + \cdots + a_1^n) = b_0 \sum_{k=0}^{n} a_1^k.$$

In summary, the unit response of the IIR filter in the example is given by

$$y[n] = b_0(1 + a_1 + a_1^2 + \cdots + a_1^n) = b_0 \sum_{k=0}^{n} a_1^k.$$

By using the geometric series formula, we can simplify the unit response as

$$y[n] = b_0 \frac{1 - a_1^{n+1}}{1 - a_1}, \quad a_1 \neq 1, n \geqslant 0.$$

In summary, the unit response of the IIR filter in the example is given by

$$y[n] = b_0(1 + a_1 + a_1^2 + \cdots + a_1^n) = b_0 \sum_{k=0}^{n} a_1^k.$$

By using the geometric series formula, we can simplify the unit response as

$$y[n] = b_0 \frac{1 - a_1^{n+1}}{1 - a_1}, \quad a_1 \neq 1, n \geqslant 0.$$

- When $| a_1 | > 1$, the output $y[n]$ is unbounded as $n \to \infty$. This is called **unstable** condition.
- When $| a_1 | < 1$, the term in the numerator a_1^{n+1} decays to zero as $n \to \infty$. This is called **stable** condition. Furthermore

$$\lim_{n \to \infty} y[n] = \frac{b_0}{1 - a_1}.$$

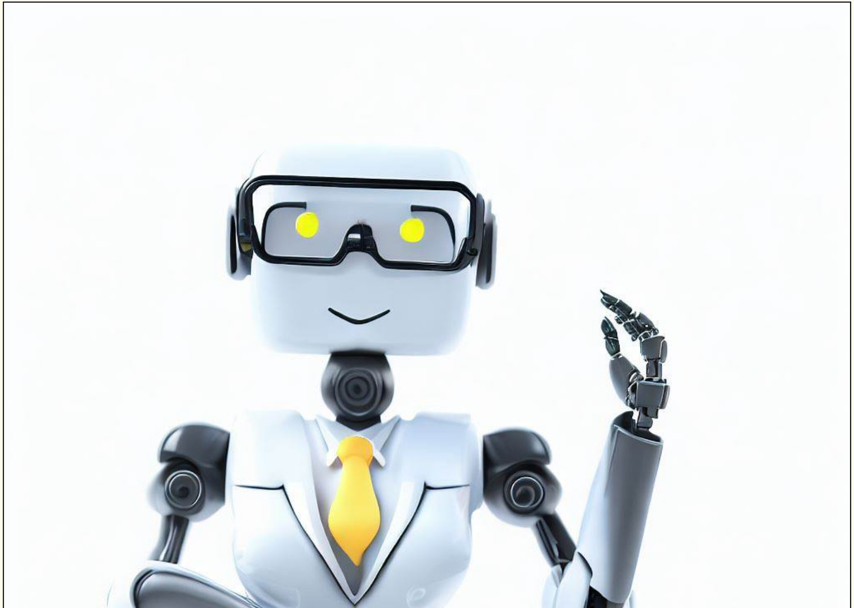
- When $| a_1 | = 1$, we have two cases: 1) $a_1 = 1$, the output is unbounded; and 2) $a_1 = -1$, the output oscillates.

To be continued...

We will come back to the stability of IIR filters discussion in the Laplace and Z transforms lectures.

Table of Contents

Let us wrap up the lecture!



Take-home Messages

- A system or a filter is an operator that transforms signals. **Finite impulse response (FIR)** filters are based on a finite number of values in a discrete input signal.
- A system is **causal** if the output at time n depends only on the input at time n and previous times.
- A discrete-time system is **time-invariant** if time-delaying the input by n_0 will result in time-delaying the output by the same amount n_0.
- A discrete-time system is **linear** if a linear combination of inputs will result in the same linear combination of outputs.

Closing Remarks: Practice Questions

The following questions might appear in the final exam:

- Consider the syste $y[n] = x[n]x[n-1] + \cos(0.3\pi n - \pi/3)$.
  - Is this system linear? Prove it.
  - Is this system time-invariant? Prove it.
  - Is this system causal? Prove it.

Tutorial exercises

During the tutorial, the exercises below will be discussed in class

- Attempt to complete the exercises **before** class starts
- As the weeks progress, more time is needed for explanation

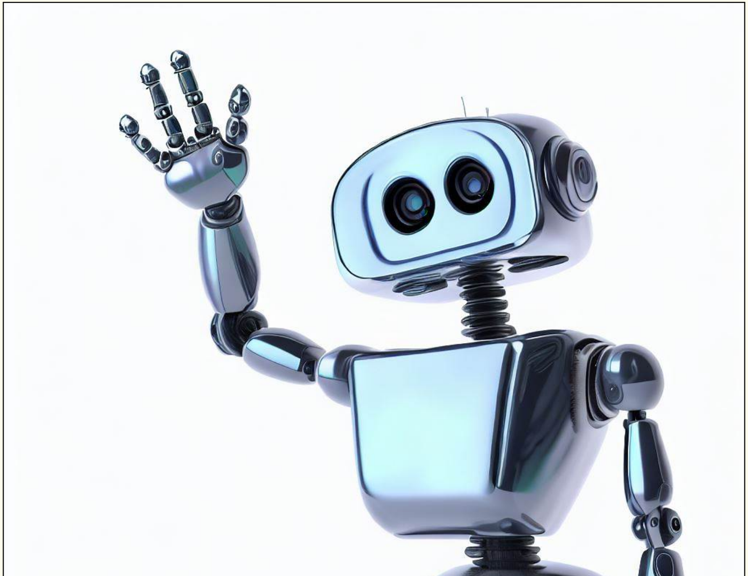
SPF	DSPF
P 5.1 (p. 126)	P 5.1 (p. 207)
P 5.2 (p. 126)	P 5.3 (p. 208)
P 5.3 (p. 126)	P 5.4 (p. 208)
P 5.7 (p. 127)	P 5.8 (p. 209)
P 5.9 (p. 128)	P 5.10 (p. 210)
P 5.12 (p. 128)	P 5.13 (p. 211)
P 5.14 (p. 128)	P 5.15 (p. 211)
P 5.17 (p. 129)	P 5.18 (p. 212)

Next Lecture

Let us analyze FIR filters in the spectrum.

Frequency Response of FIR Filters

Have a nice day!



Acknowledgements

The material for this lecture series was developed by dr. Arnold Meijster and dr. Harmen de Weerd and modified by Juan Diego Cardenas-Cartagena.

Disclaimer

- Grammar was checked with Grammarly and Grammar checker GPT.
- Images without source were created with the assistance of DALL·E.