Imperial College London

Lecture 13

Impulse Response & Filters

Peter Cheung
Dyson School of Design Engineering

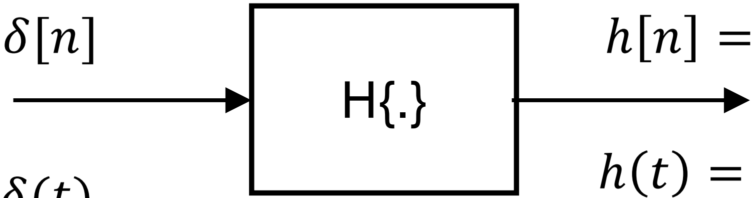
URL: www.ee.ic.ac.uk/pcheung/teaching/DE2_EE/
E-mail: p.cheung@imperial.ac.uk

Impulse Response of a Discrete System

- The response of a discrete time system to a discrete impulse at the input is known as the system's **impulse response**

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

$x(t) = \delta(t)$



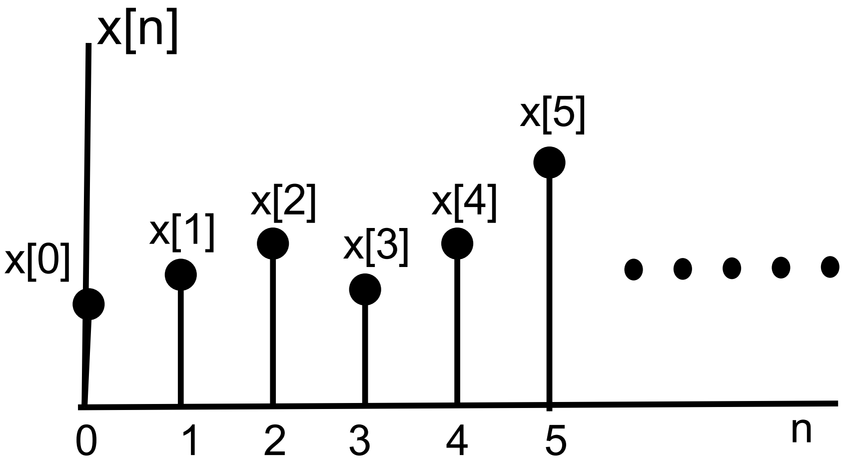
$h[n] = \text{discrete impulse response}$

$h(t) = \text{continuous impulse response}$

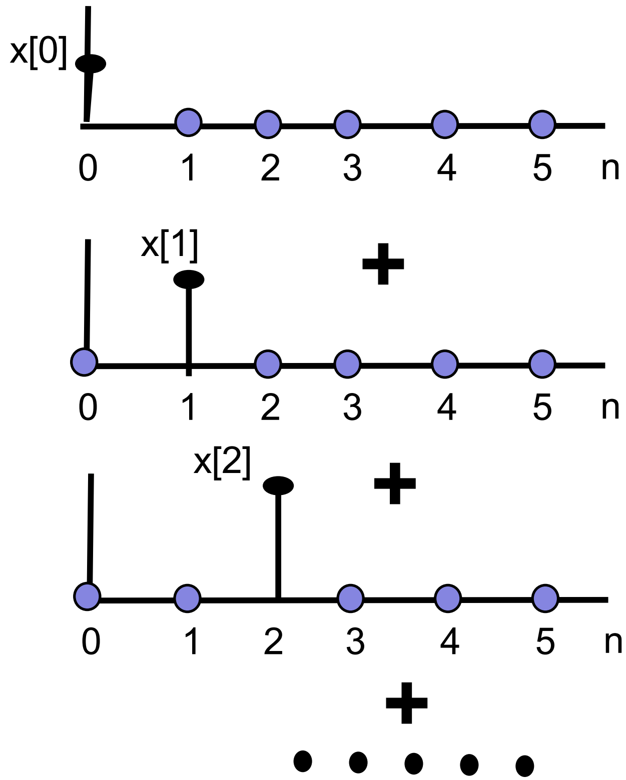
- The impulse response of a **linear** system completely define and characterise the system – both its transient behaviour and its frequency response.
- This applies to both continuous time and discrete time linear systems.
- An impulse signal contains ALL frequency components (L3, S7). Therefore, applying an impulse to input stimulates the systems at ALL frequencies.
- Since integrating a unit impulse = a unit step function, we can obtain the step response of the system by integrating the impulse response.

Discrete signal as sum of weighted impulses

- Remember from L10, S7, we can represent a causal discrete signal *x[n]* in terms of sum of weighted delayed impulses:

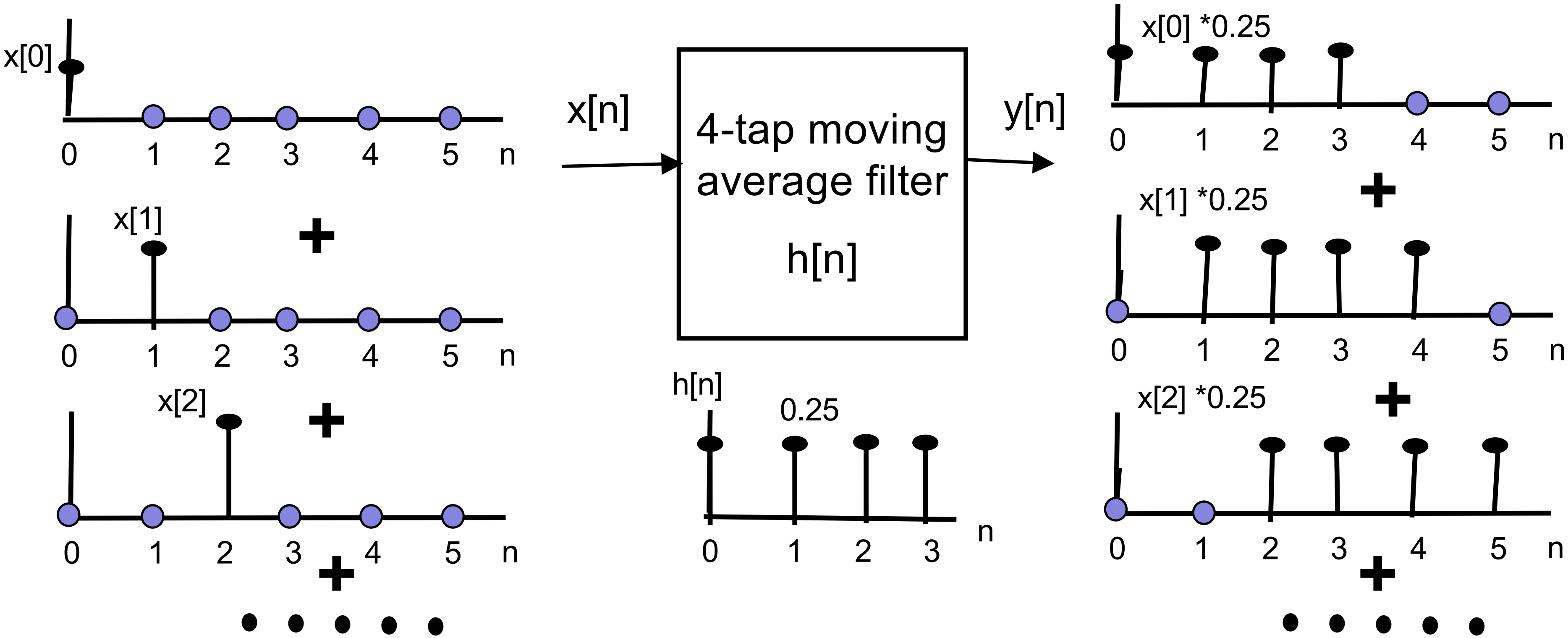


=



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

Impulse Response and Convolution

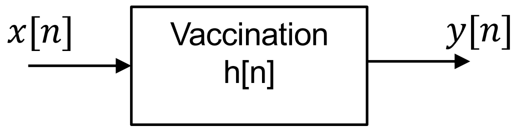


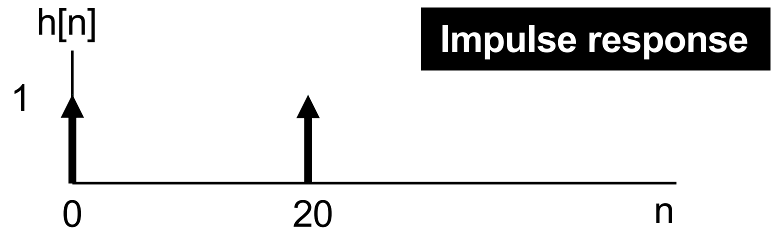
- We can therefore derive the output of a discrete linear system, by adding together the system's response to EACH input sample separately.
- This operation is known as **convolution**:

$$y[n] = x[n] * h[n] = \sum_{m=0}^{\infty} x[m]h[n-m]$$

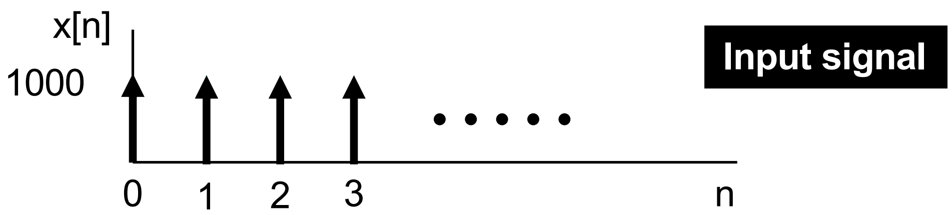
Convolution example - COVID vaccination

- Covid vaccine require two doses, 3 weeks apart. Impulse response h[n] for this vaccination system is:



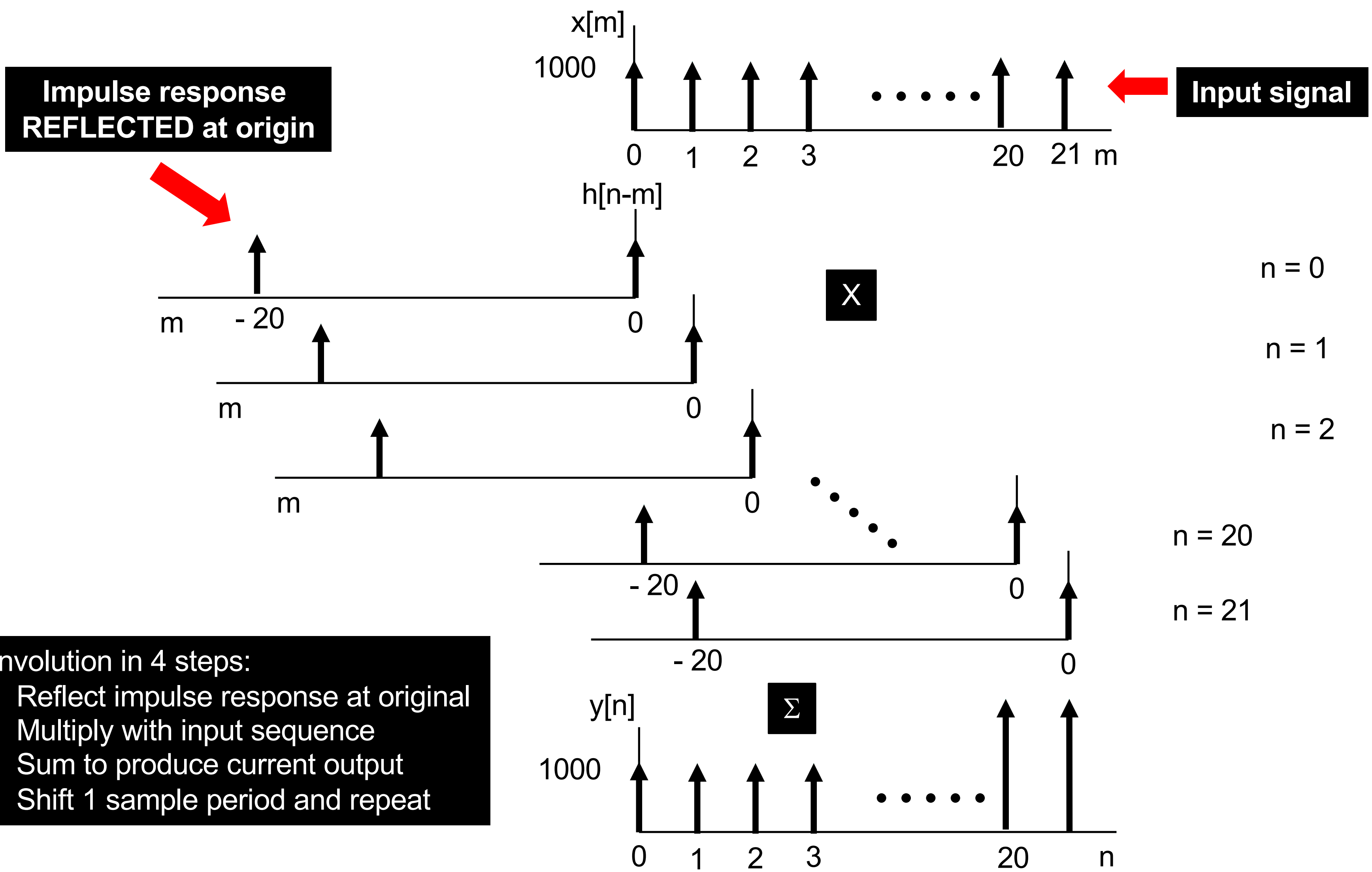


- UK started vaccination of its elderly population with a plan of vaccinating 1000 people per day after the first day. The input x[n] is:



- How many doses would NHS needs to provide from day 0? (i.e. y[n])

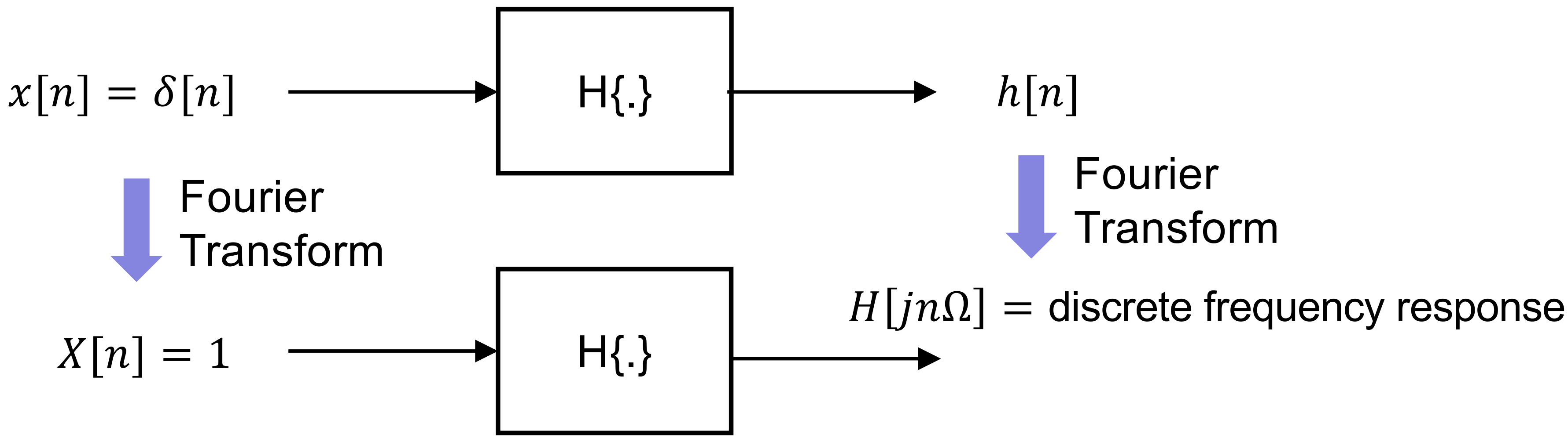
Graphical representation of Convolution



Convolution in 4 steps:
1. Reflect impulse response at original
2. Multiply with input sequence
3. Sum to produce current output
4. Shift 1 sample period and repeat

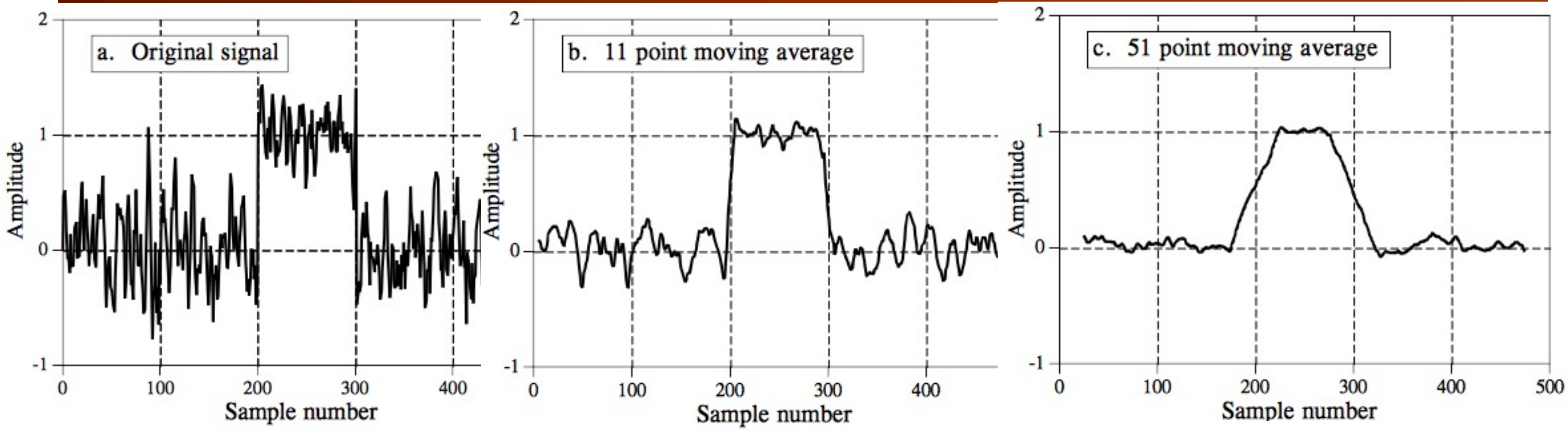
Impulse Response & Frequency Response

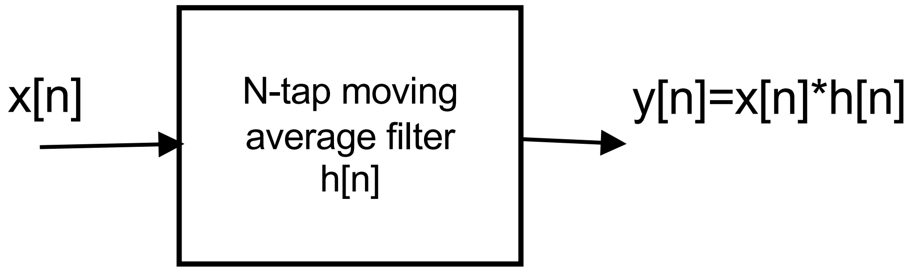
- Since a unit impulse contains all frequency, and its Fourier transform is a constant at 1 (see L3, S7), the Fourier transform of the impulse response h[n] or h(t) give us the systems' frequency response:



- This applies to both continuous time and discrete time linear systems.
- An impulse signal contains ALL frequency components (L3, S7). Therefore, applying an impulse to input stimulates the systems at ALL frequencies.

Moving Average Filter = FIR lowpass filter



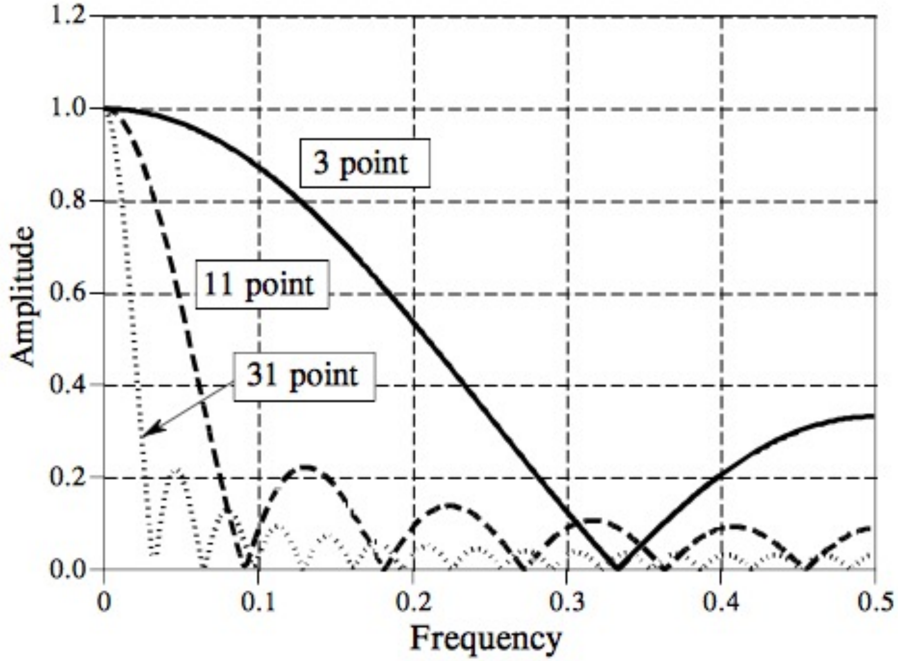


- N-tap (or N point) moving after filter – high N, lower the cut off frequency
- For a N-tap moving average filter, it impulse response has N impulses.
- If input x[n] has M non-zero samples (i.e. finite length), output y[n] is also finite in length, and has M+N non-zero samples. Hence the name Finite Impulse Response (FIR) filter.

Frequency Response of N-tap moving average filter

- The impulse response of a moving average filter is a rectangular pulse.
- The Fourier transform of a rectangular pulse is of the form (sin x)/x or sinc(x) function (see Lecture 3 slide 6) in the case of continuous time.
- For discrete time case, the frequency response of a moving average filter with N taps (or points) is:

$$\boxed{H[f] = \frac{\sin(\boldsymbol{f}N)}{N \sin f}}$$



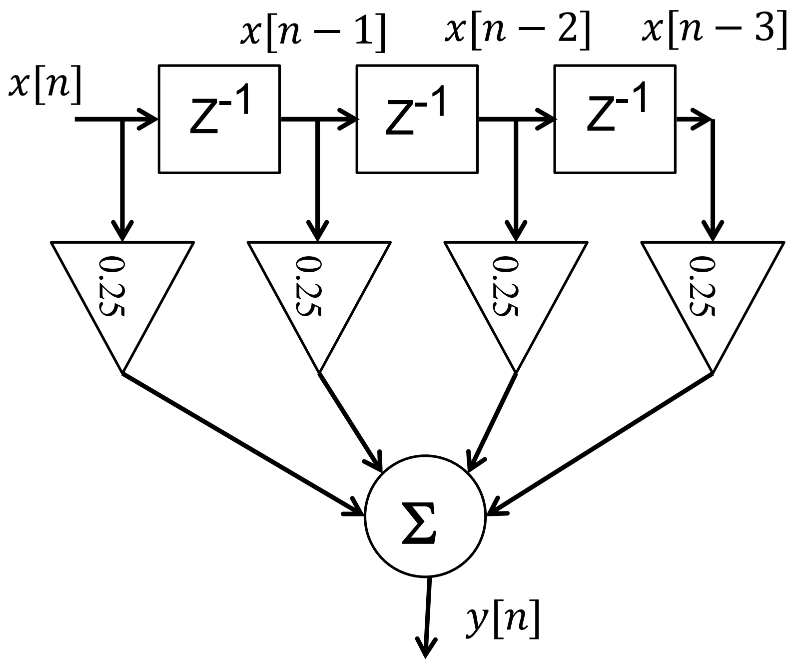
- Here *f* is normalised to 0 -> 0.5 x sampling frequency *fs*.

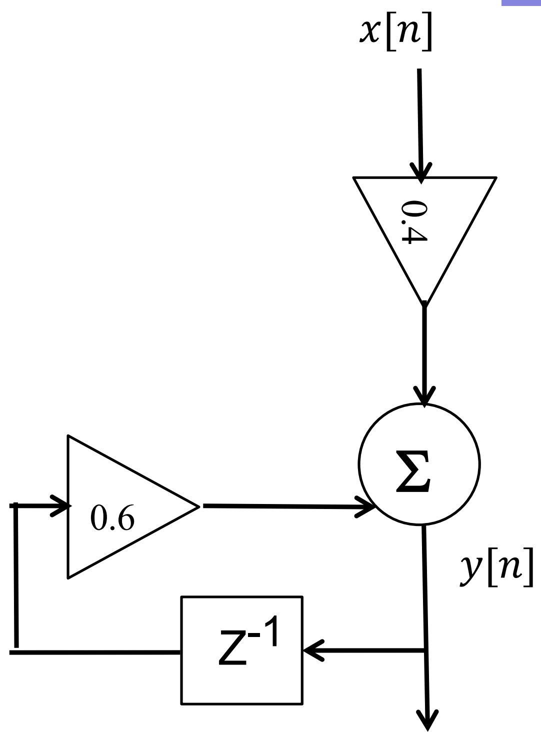
Recursive or Infinite Impulse Response Filter

FIR

x[n] =
{1.0, 1.0, 1.0, 1.0, 1.1, 0.8, 1.2, 0.9, 1.0, 1.2, 0.9, …}

IIR

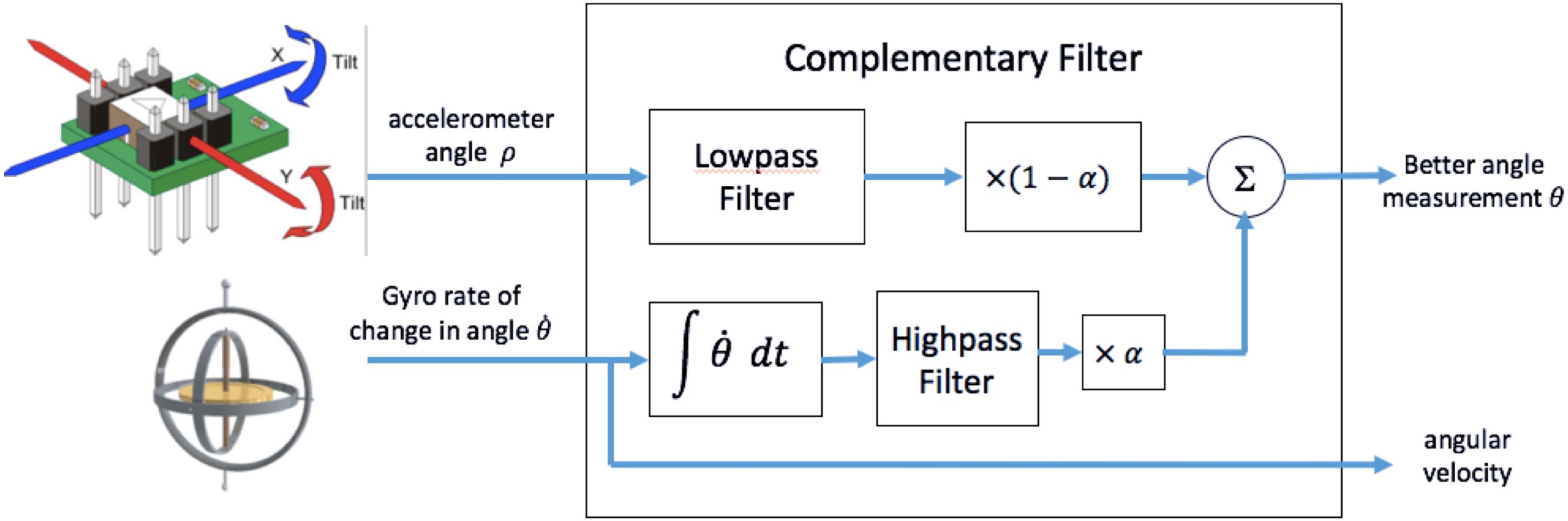




y[n] = (0.25, 0.5, 0.75, 1.0, 1.025,
0.975, 1.025, 1.0, 1.0, 1.075, 1.0, …..}

y[n] = (0.4, 0.64, 0.784, 0.87, 0.962,
0.897, 1.018, 0.971, 0.983, 1.07,
1.002…..}

Complementary Filter used with IMU



$$\text{angle } \theta_{new} = \alpha \times \left(\theta_{old} + \ \dot{\theta}\ dt \right) + (1-\alpha) \times \rho$$

where

α = scaling factor chosen by users and is typically between 0.7 and 0.98

ρ = accelerometer angle

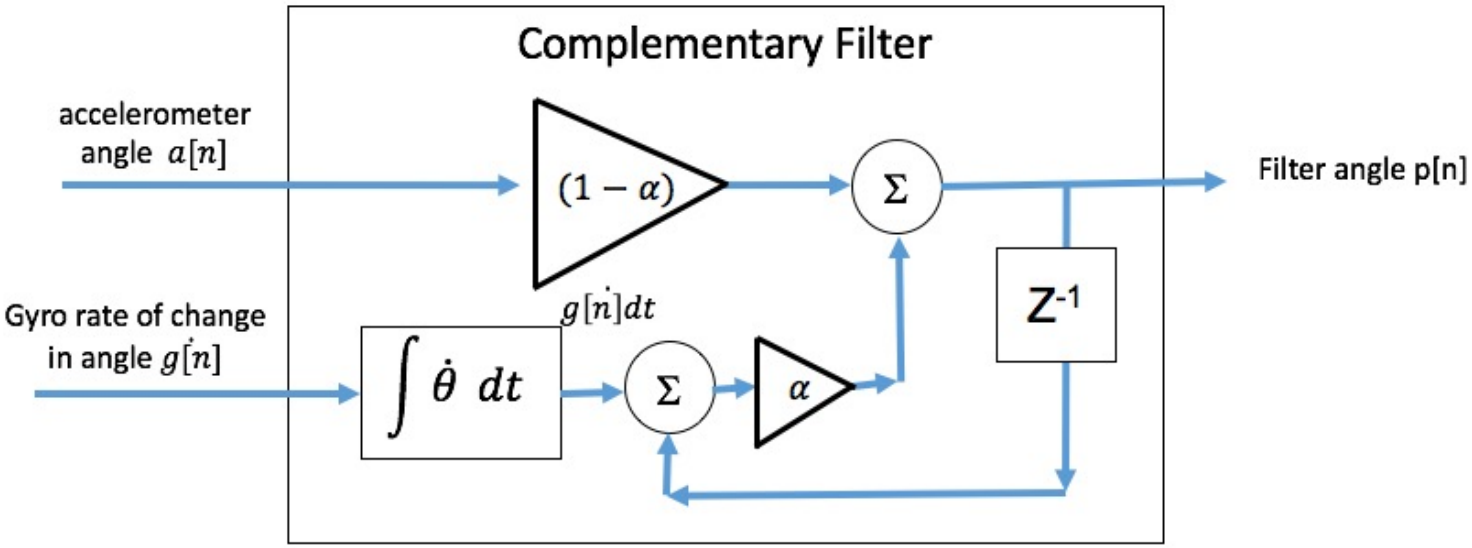
θ_{new} = new output angle

θ_{old} = previous output angle

$\dot{\theta}$ = gyroscope reading of the rate of change in angle

dt = time interval between gyro readings

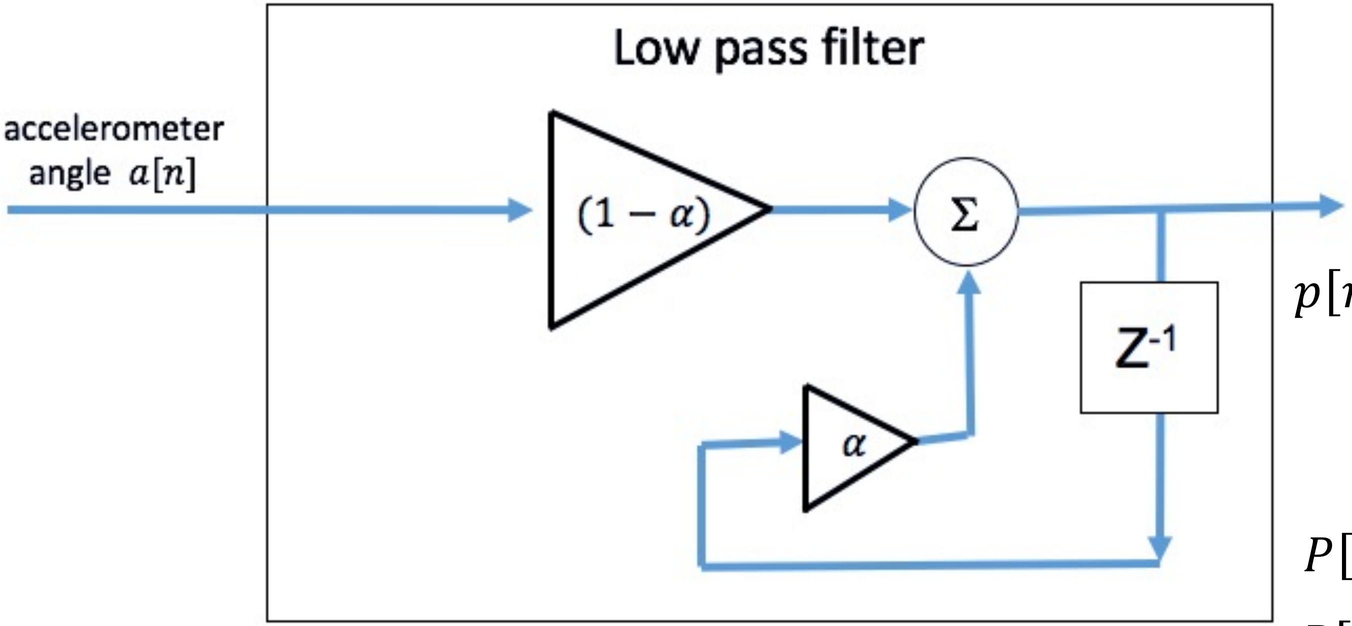
Signal flow diagram model



$$p[n] = \alpha(p[n-1] + g[n] \times dt) + (1-\alpha)a[n]$$

```
def read_imu(dt):
    global g_pitch
    alpha = 0.7     # larger = longer time constant
    pitch = int(imu.pitch())
    roll = int(imu.roll())
    g_pitch = alpha*(g_pitch + imu.get_gy()*dt*0.001) + (1-alpha)*pitch
```

Lowpass filter the accelerometer data



$$p[n] = \alpha p[n-1] + (1-\alpha)a[n]$$

z-Transform

$$P[z] = \alpha z^{-1}P[z] + (1-\alpha)A[z]$$

$$P[z] - \alpha z^{-1}P[z] = (1-\alpha)A[z]$$

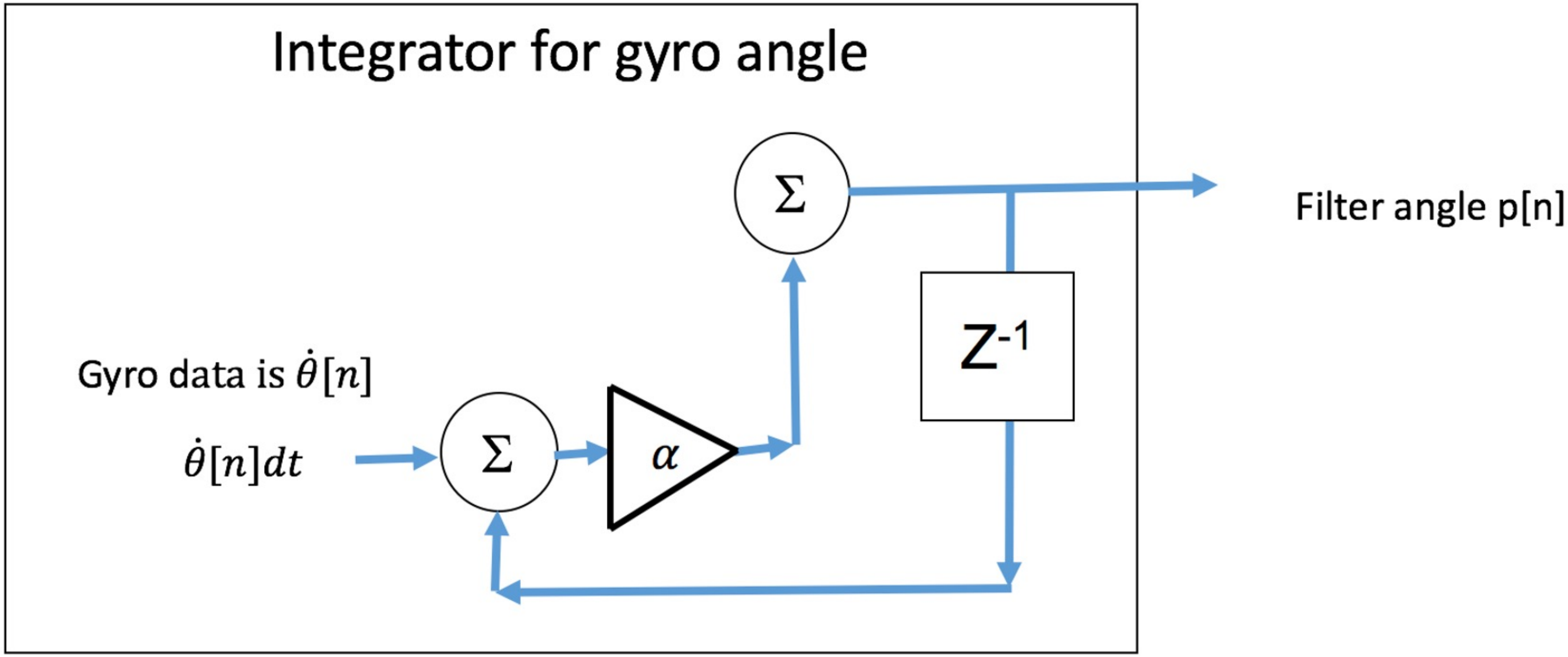
$$H[z] = P[z]/A[z] = \frac{1-\alpha}{1-\alpha z^{-1}}$$

- Now assume that gyroscope reading is zero (i.e. steady state tilt), g[n] = 0.
- Now the system is exactly the same as that in Lecture 12, slide 12.
- Therefore the accelerometer data a[n] is lowpass filtered!

Time Constant $\tau \approx \frac{\alpha}{1-\alpha} dt$

and dt is the sampling period

Integrating the gyroscope reading

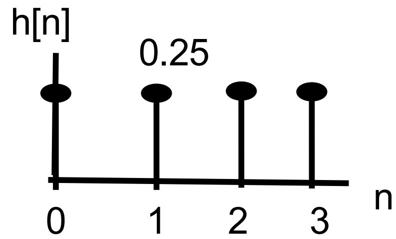


$$p[n] = \alpha(p[n-1] + \dot{\theta}[n]dt)$$

Three Big Ideas (1)

1. A discrete time system can be characterized by its impulse response:

$$h[n] = b_0\delta[n] + b_1\delta[n-1] + b_2\delta[n-2] \ldots . +b_k\delta[n-k]$$



Impulse response of a 4-tap moving average filter

2. Once we know the impulse response $h[n]$, and the input sequence $x[n]$, we can find the output $y[n]$ by convolution:

$$y[n] = x[n] * h[n] = \sum_{m=0}^{\infty} x[m]h[n-m]$$

Three Big Ideas (2)

3. Convolution operation can be performed in four steps:

   1) Reflect impulse response at original to get h[n-m]
   2) Multiply input sequence x[m] with h[n-m]
   3) Sum the product of the two sequences to get one output y[n]

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

   4) Advance the impulse response by one sample period and repeat to get the next y[n]