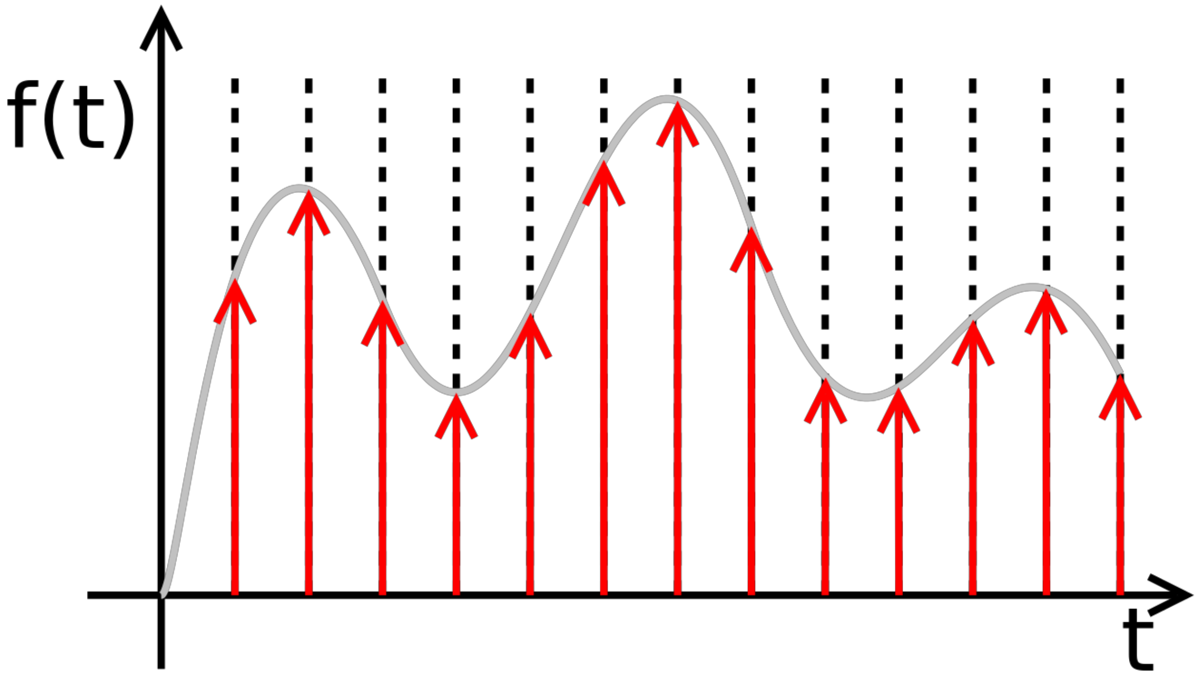
POLITECNICO
MILANO 1863

DIPARTIMENTO DI ELETTRONICA
INFORMAZIONE E BIOINGEGNERIA

1D discrete signals analysis

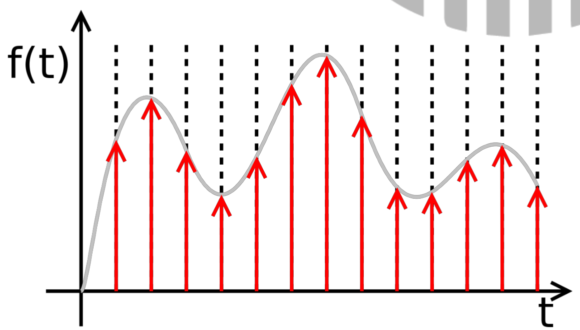
1D signals(t)

- x-axis represents time → $y = f(t)$ is a time-variant signal



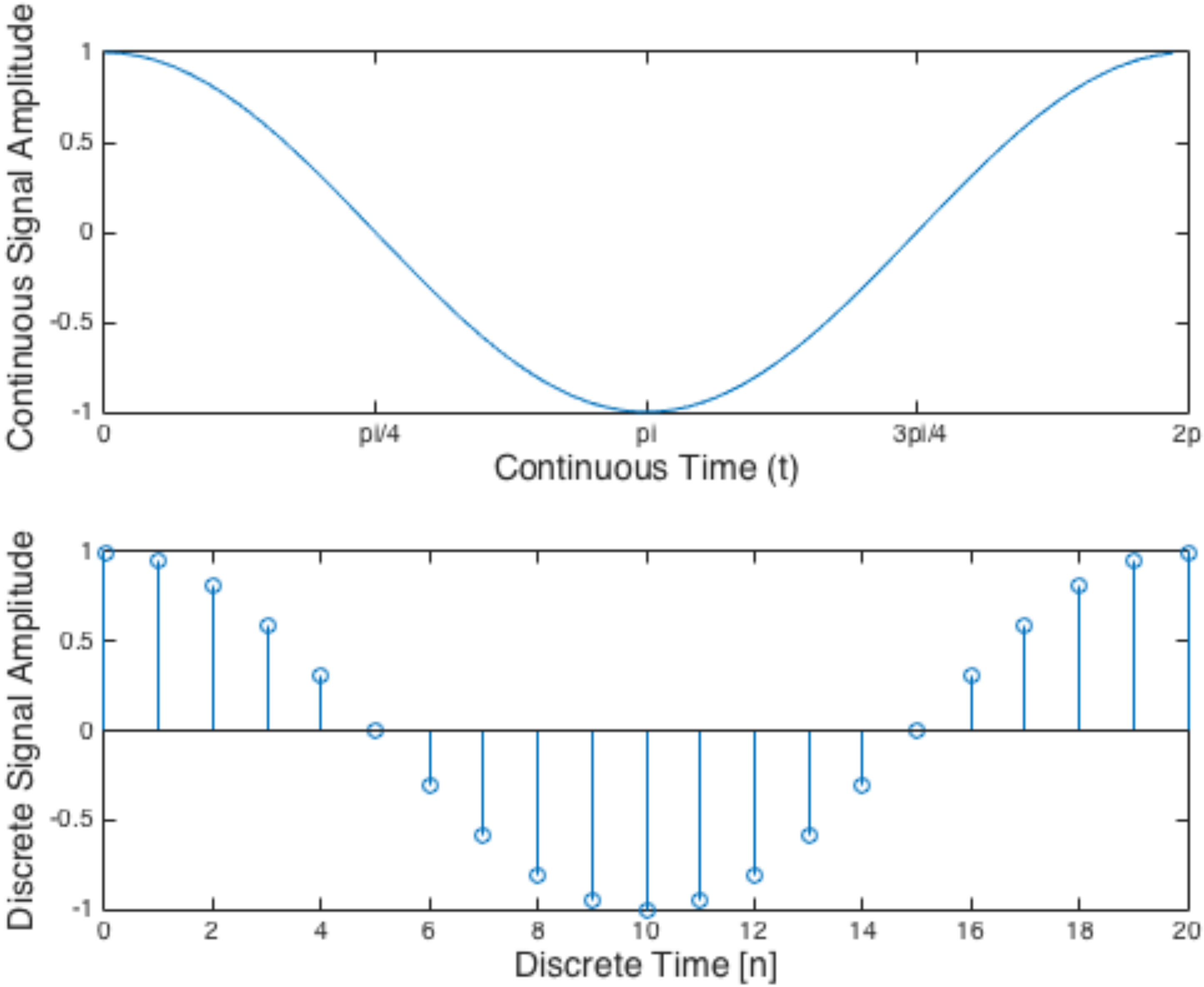
From continuous to discrete-time signals

- If we sample $y = f(t)$ every T_s time istant
- $t \rightarrow n \cdot T_s$, with $n = 0, 1, 2, \ldots$
- $y_n = f(t_n)$
- T_s = sampling time or sampling period



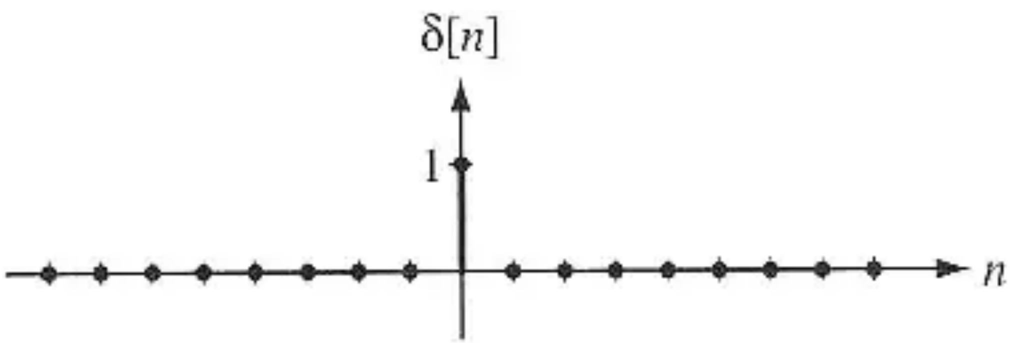
- $f_s = \frac{1}{T_s}$ = sampling frequency or sample rate
- We can represent y_n and t_n as two arrays with same # of elements and use MATLAB to process these signals

From continuous to discrete-time signals



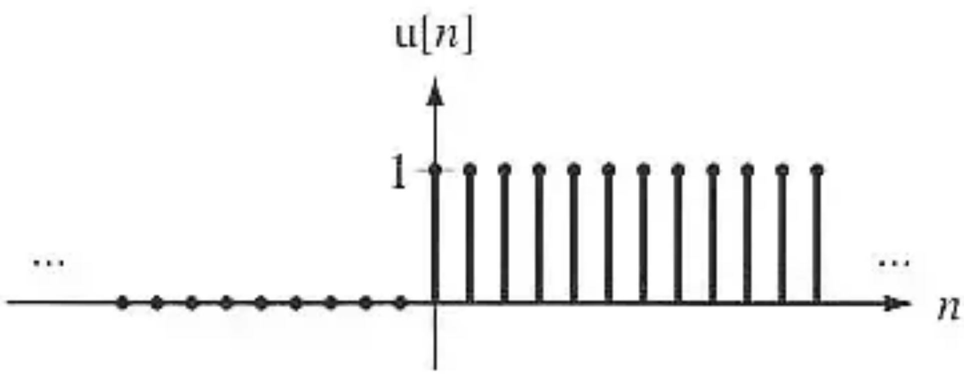
1D discrete impulse

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



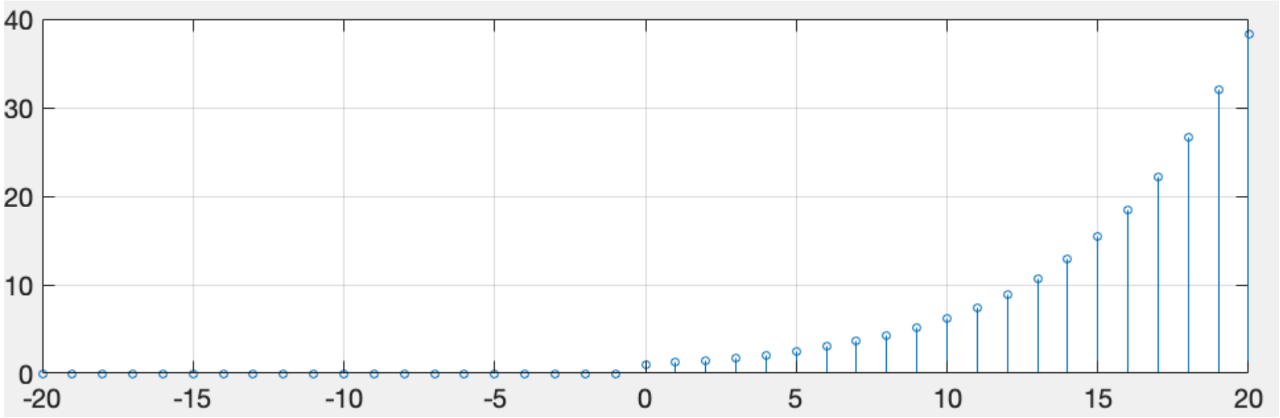
1D discrete unit step

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

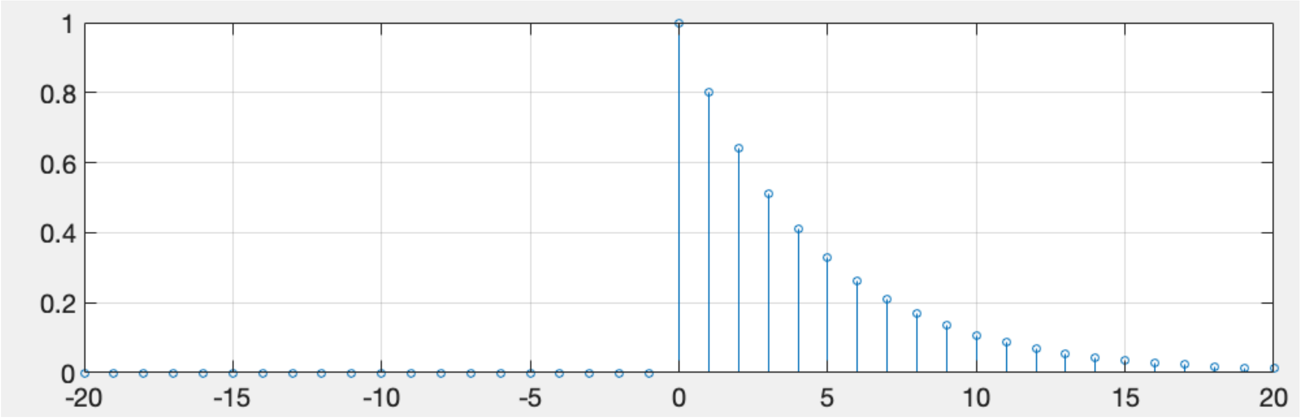


1D discrete exponential step

- $x(n) = a^n u(n)$
  - $a = 1.2$

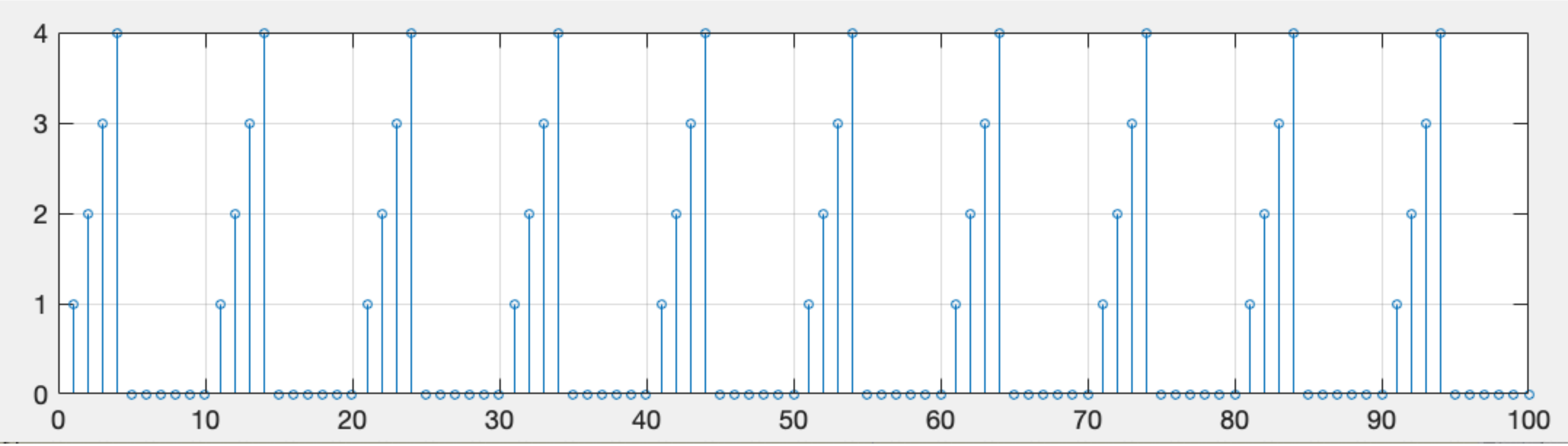


  - $a = 0.8$



1D discrete periodic signals

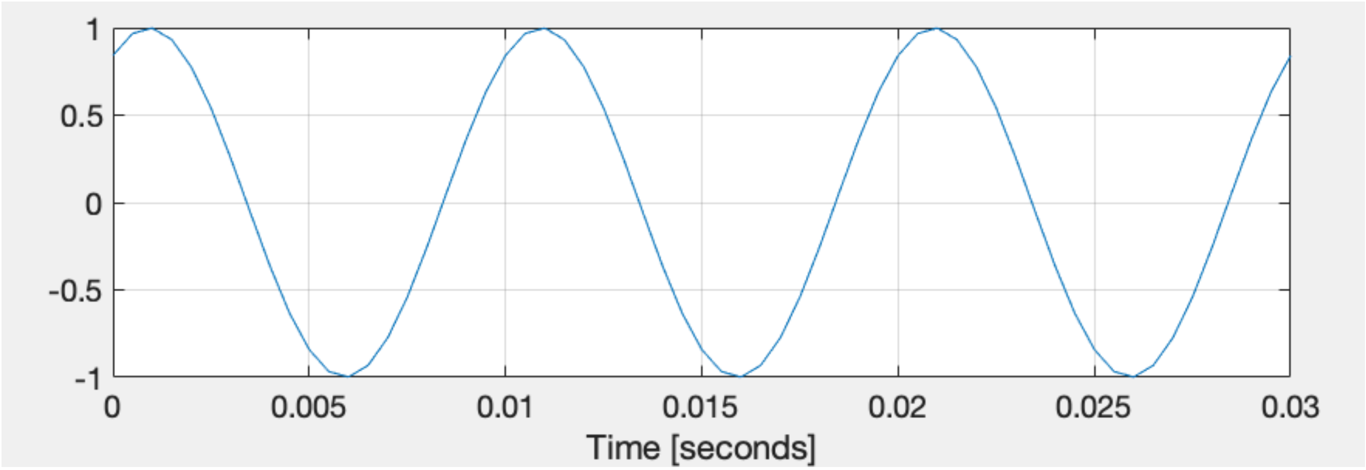
$$x(n) = x(n + kT)\ \forall k \in \mathbb{Z}$$



$$T = 10$$

1D continuous-time sinusoids

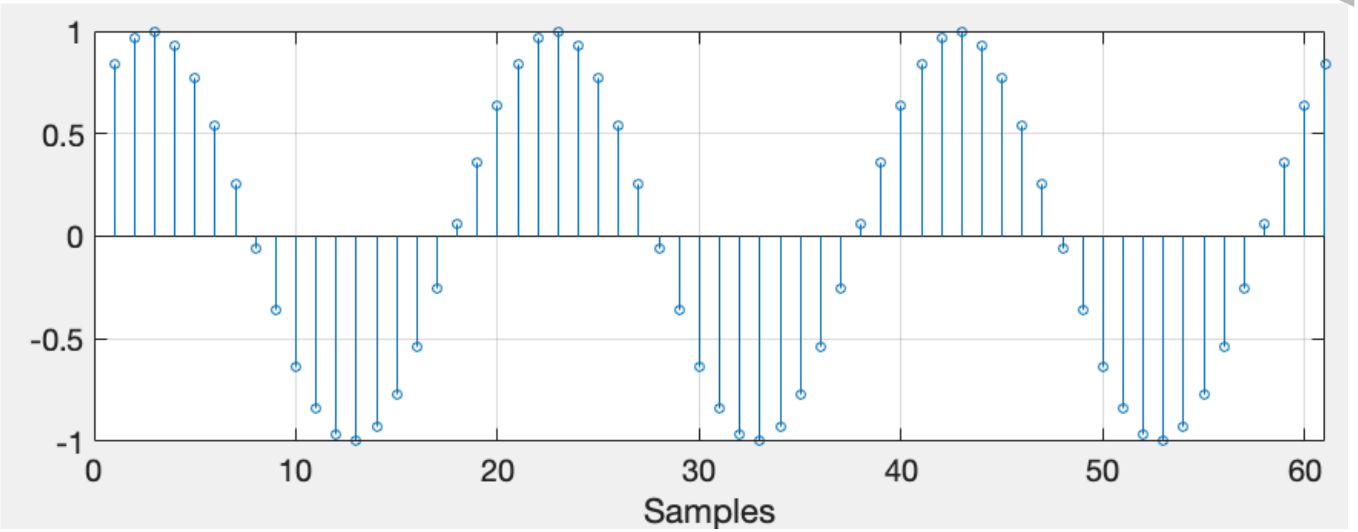
- $y(t) = A \cdot \cos(2\pi f_o t + \phi)$



- A = amplitude
- f_0 = frequency
- ϕ = phase

1D discrete-time sinusoids

- $y(t) = A \cdot \cos(2\pi f_o t + \phi) \rightarrow y(n) = A \cdot \cos(2\pi f_o T_s n + \phi)$



- $t = n \cdot T_s$
- $n = 0, 1, 2, \ldots N$ = samples
- $T_s =$ sampling time or sampling period
- $\frac{1}{f_0} =$ period of the sinusoid

1D discrete-time sinusoids

- NB:

$$y(t) = A \cdot \cos(2\pi f_o t + \psi) \rightarrow y(n) = A \cdot \cos(2\pi f_o T_s n + \phi)$$

OR

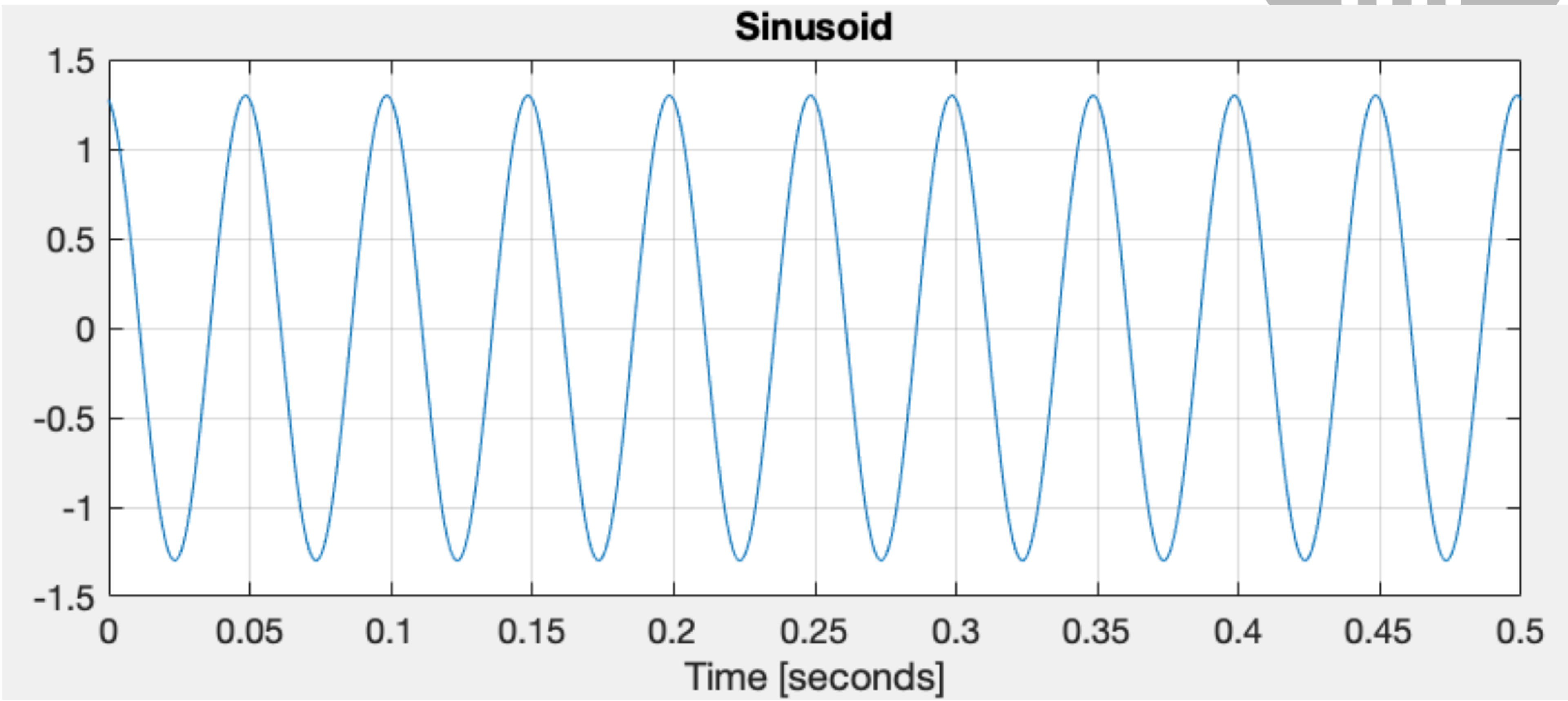
$$y(t) = A \cdot \cos(2\pi f_o t + \phi) \rightarrow y(n) = A \cdot \cos\left(2\pi \tilde{f}_o n + \phi\right)$$

- $t = n \cdot T_s$
- $n = 0, 1, 2, \ldots N$ = samples
- $\tilde{f}_o$ = normalized frequency = f_o / F_s = $f_o \cdot T_s$

How to plot 1D discrete-time signals

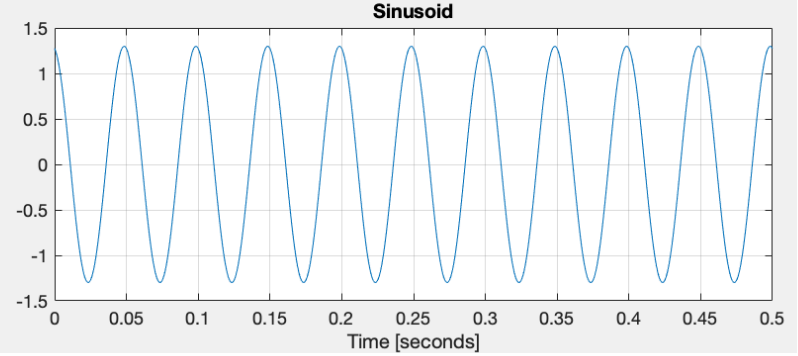
```
close all
clearvars
clc

%% parameters

T_s = .001; % sampling time
T_f = .5; % temporal duration [seconds]
f_0 = 20; % sinusoid frequency
phi = .2; % phase
A = 1.3; % amplitude

%% temporal axis

t = 0:T_s:T_f;

%% y-axis

y = A*cos(2*pi*f_0*t + phi);

%% plot

figure(1); % open new figure and call it Figure 1
plot(t, y); % --> NB: dimensions must be consistent!
grid; % insert a grid
title('Sinusoid'); % title
xlabel('Time [seconds]'); % label of x-axis
set(gca, 'fontsize', 18) % increase fontsize
```

How to plot 1D discrete-time signals

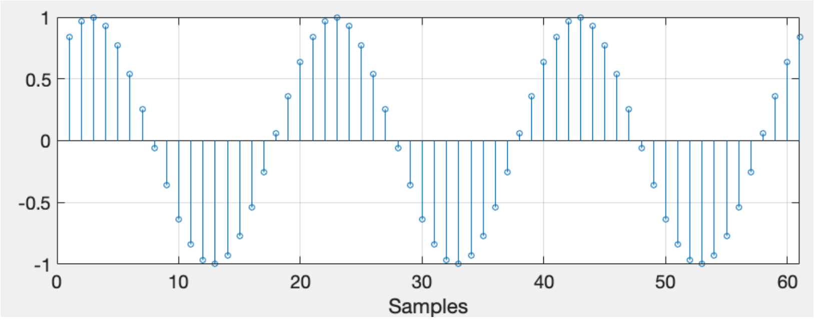


How to plot 1D discrete-time signals

- Use ‘plot(x-axis, y-axis)’ or ‘plot(y-axis) for a continuous line



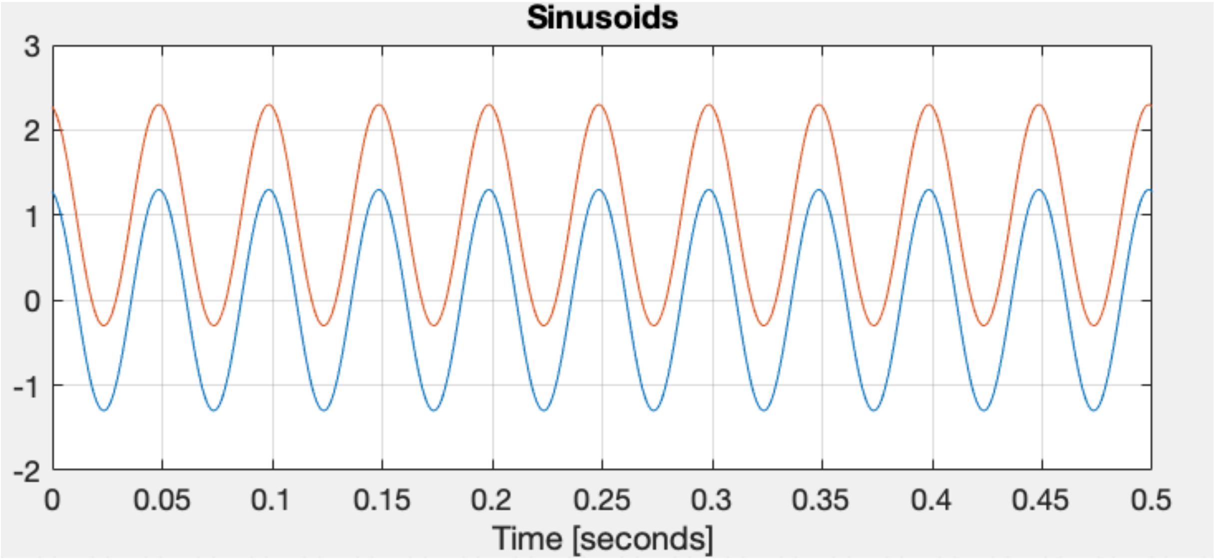
- Use ‘stem(x-axis, y-axis)’ or ‘plot(y-axis)’ for highlighting the single samples



How to plot 1D discrete-time signals

- ‘hold on’ allows to insert multiple plots into the same figure

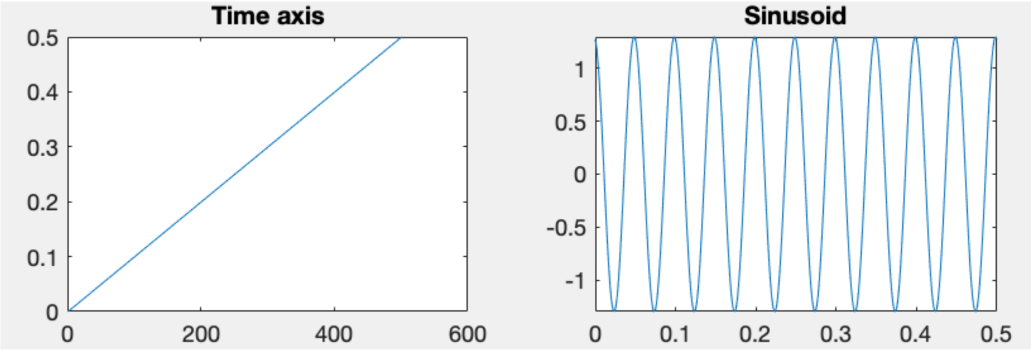
```
figure(1); % open new figure and call it Figure 1
plot(t, y); % --> NB: dimensions must be consistent!
grid; % insert a grid
title('Sinusoids'); % title
xlabel('Time [seconds]'); % label of x-axis
set(gca, 'fontsize', 18) % increase fontsize

hold on,
plot(t, y + 1);
```



How to plot 1D discrete-time signals

- Once a Figure has been opened, you can insert whatever you want:
  - 'xlabel' and 'ylabel'
  - 'title'
  - 'legend'
  - grid
  - markers, colors, linestyle etc...
- You can put multiple non-overlapping plots inside the same figure: 'subplot(#rows, #cols, #plot index)

```
figure(2)
subplot(1, 2, 1)
plot(t)
title('Time axis');
set(gca, 'fontsize', 18)
subplot(1, 2, 2)
plot(t, y)
title('Sinusoid');
set(gca, 'fontsize', 18)
```



Exercises

Exercise 1: discrete-time sinusoids

- Build a signal x(n) as the sum of three different sinusoids sin(2πft) at the normalized frequencies $\omega_1 = 0.11, \omega_2 = 0.09, \omega_3 = 0.3$. The sampling period is T = 0.3 seconds, and the signal is defined for t in [0, 100] seconds.
- Plot the signal as a function of time.
- Compute the period P for each of the three sinusoids.
- Check: build the signal x1(n) as the sum of the three sinusiods, written as sin(2πt/P).
- Plot x1 in the same figure. Are the signals totally overlapped?

Exercise 2: exam 7/11/2018 [3pts]

- Given the signal $x(t) = A\cos(2\pi f t)$
  1. Write the script 'es2.m' to create the signal x(n) as x(t) from 0 to 0.5 seconds, sampled at Fs (sampling rate) = 1000Hz; A = 0.8, f = 50Hz.
  2. Write the function 'sinusoid.m' which takes as input the time-axis, the amplitude, the frequency, the phase of a discrete sinusoid and return the signal.
  3. Generate the same signal as 1. with 'sinusoid.m'
  4. In 'es2.m', plot the signal as a function of n.
  5. In 'es2.m', plot the signal as a function of time samples.

Exercise 3: exam 10/09/2018 [3pts]

- Generate 5 cosine tones with the following parameters:

	Amplitude	Frequency [Hz]	Phase [deg]
x1	1.0	220	0
x2	0.75	440	45
x3	0.5	660	90
x4	0.25	880	135
x5	0.125	1100	180

- All 5 signals have a duration of 1 second and a sampling rate of 8000 Hz
- Generate the signal x6 as the sum of the five signal generated in point 1

Exercise 4: signal shift

- Generate the signal x(n) = (0.8)^n u(n), n = 1:20
- Generate the signal y1(n) = x(n-5), n = 1:20
- Generate the signal y2(n) = x(n+5), n = 1:20
- Hint: Consider using ‘circshift’ instead of for loops.
- Plot the signals in the same figure.

Exercise 5: periodic sequences

- Generate the signal x(n) = u(n-5) –u(n-10), considering n = 1:15.
- Generate the periodic signal xp(n) with period N = 15, considering n = 1:200.
- Hint: Consider using ‘repmat’ instead of for loops.
- Plot the periodic signal xp(n) considering only 8 periods.

Random sequences

How to generate random variables

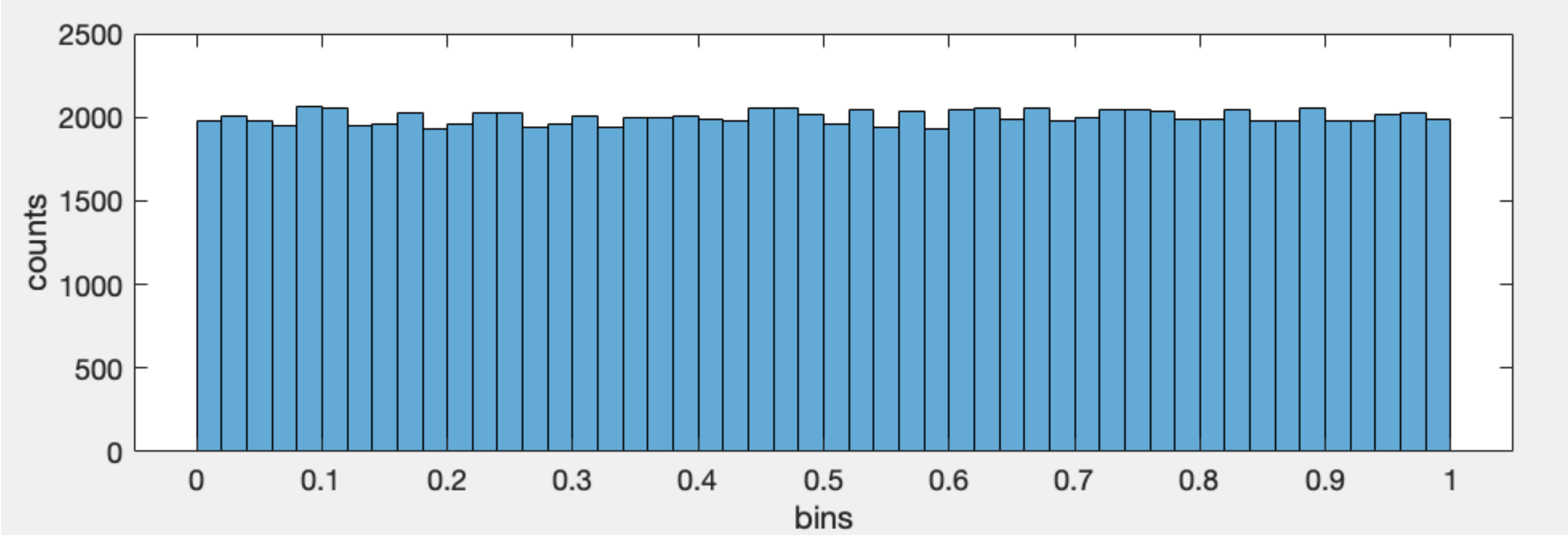
- Uniformly distributed random variables
  - 'rand(matrix size)' full of $\sim U(0, 1)$ variables
- Gaussian distributed random variables
  - 'randn(matrix size)' full of $\sim N(0, 1)$ variables

Histograms are directly related to the probability density function of variables (PDF)

How to generate random sequences

- Generate 100000 random variables with pdf ~ $U(0, 1)$

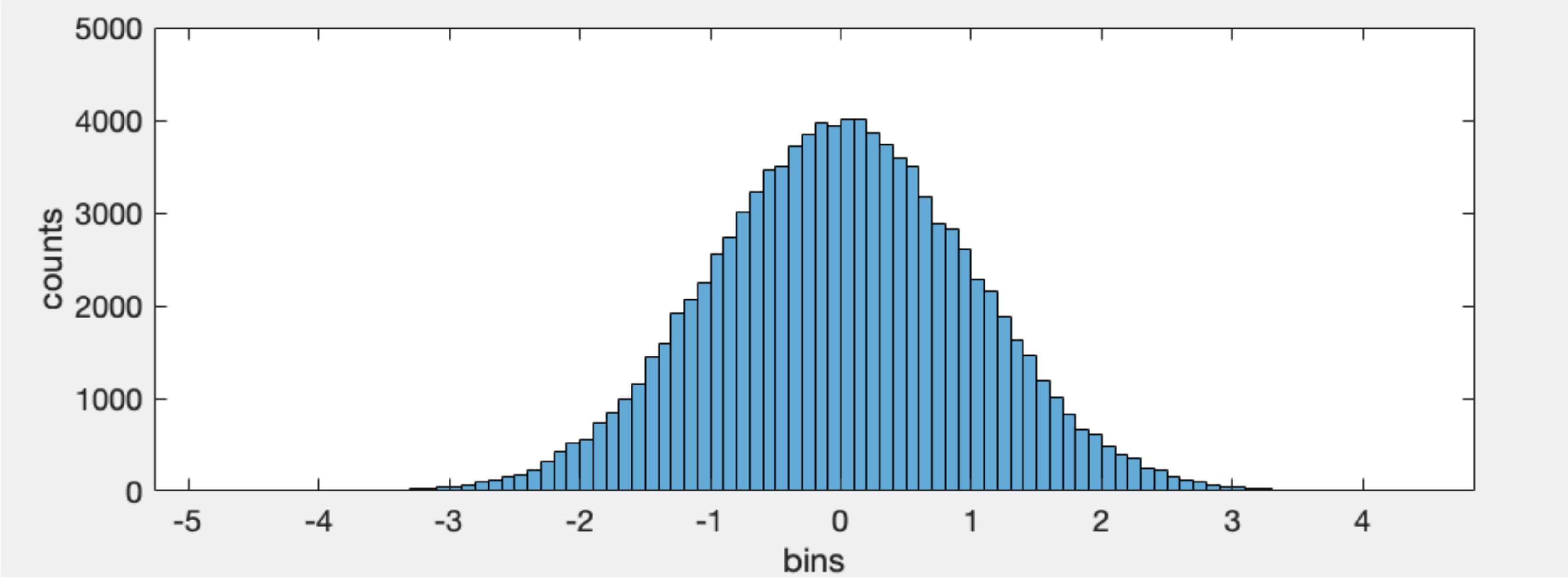
```
A = rand(1, 1e5);
h = histogram(A);
```



How to generate random sequences

- Generate 100000 random variables with pdf ~ $N(0, 1)$

```
A = randn(1, 1e5);
h = histogram(A);
```



Exercise 6: listening to random noise

- Generate noise(n) as a set of uniformly distributed random variables between –A and A, A=0.05. The time axis has a 2 seconds duration with Fs=11.025 KHz.
- Generate x(n) as sinusoidal sequence with frequency 220 Hz
- Generate the signal y(n) = x(n) + noise(n)
- Normalize the signal in [-1, 1]
- Play y(n), testing different values of A

Exercise 7: Gaussian pdf

- Generate g(n) as a set of 10000 realizations of random variables distributed as ~ $N(0, 1)$.
- Estimate the mean and the variance
- Hint: you can use 'mean' and 'var'
- Create h(n) = a * g(n) + b, with a = 0.1; b = 4.
  - Is h(n) still distributed as $\sim N(\mu, \sigma^2)$?
  - Estimate the mean and the variance of h(n).

Linear Time-Invariant Systems (LTI)

Definition of LTI

- The defining properties of any LTI system are *linearity* and *time invariance*.
  - Linearity = input-output relationship is LINEAR
  - Time invariance = the output does not depend on the particular time the input is applied. If the output due to x(t) is y(t), the output due to x(t-k) is y(t-k).
- The system can be completely characterized by its impulse response h(t).

Output of LTI discrete systems

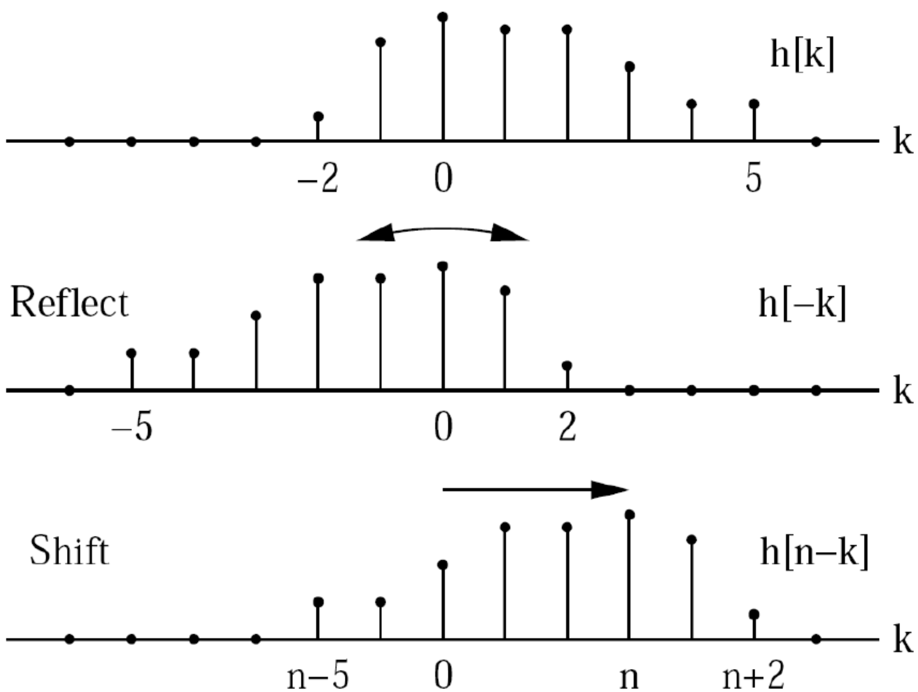
- The output of LTI discrete systems is always the convolution between the input signal and the impulse response h(n).

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

Discrete signal convolution

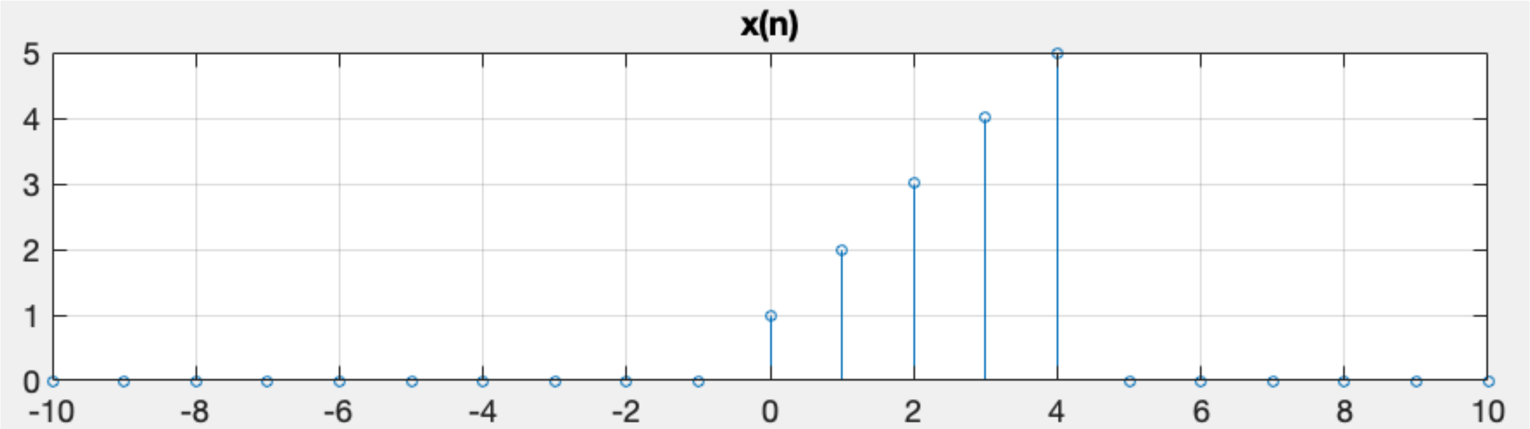
$$y(n) = x(n) * h(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k)$$

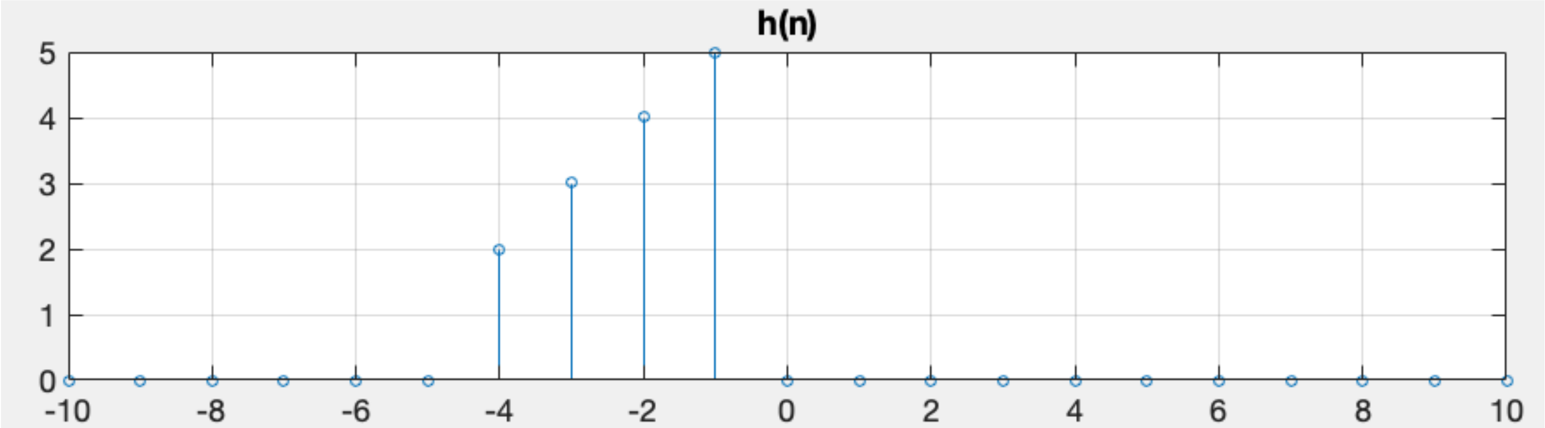
NB: $h(n-k) = h(-(k-n))$ → Operation order:



Discrete signal convolution: example

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

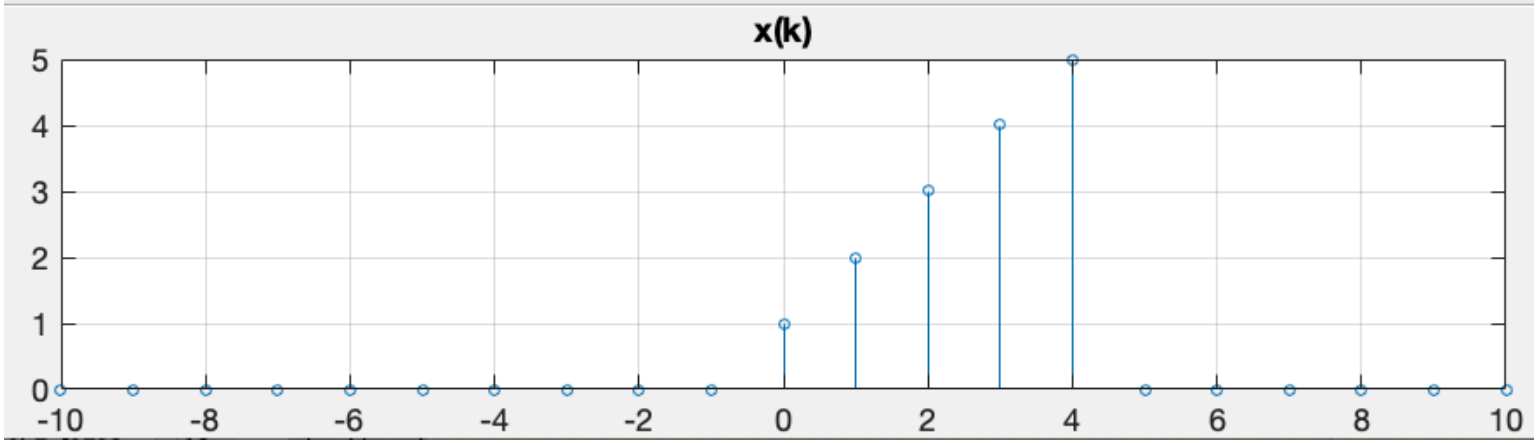


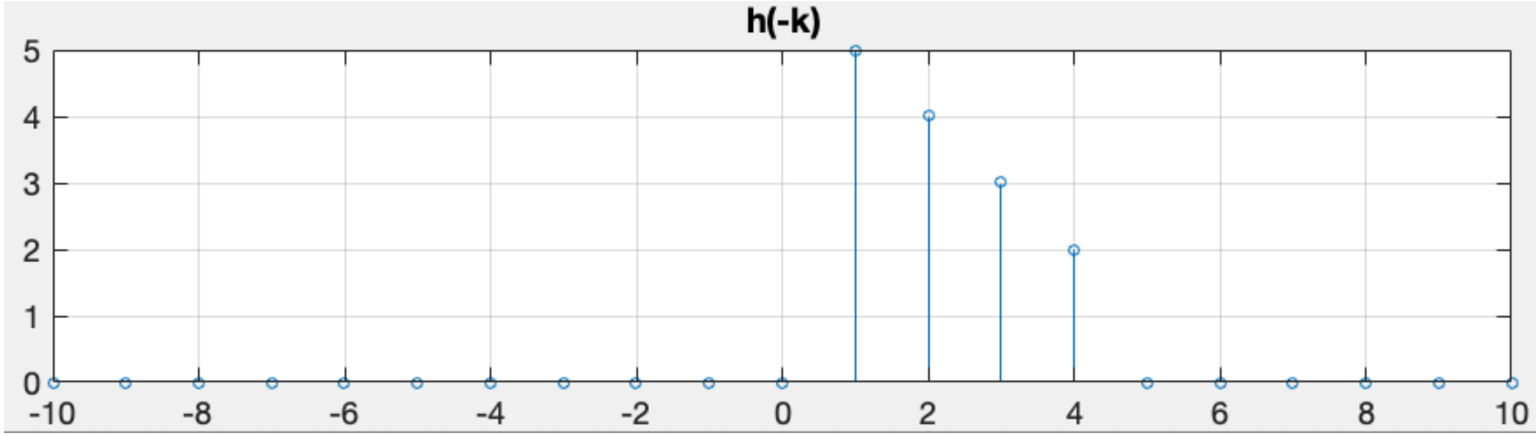


Discrete signal convolution: example

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

$n = 0$

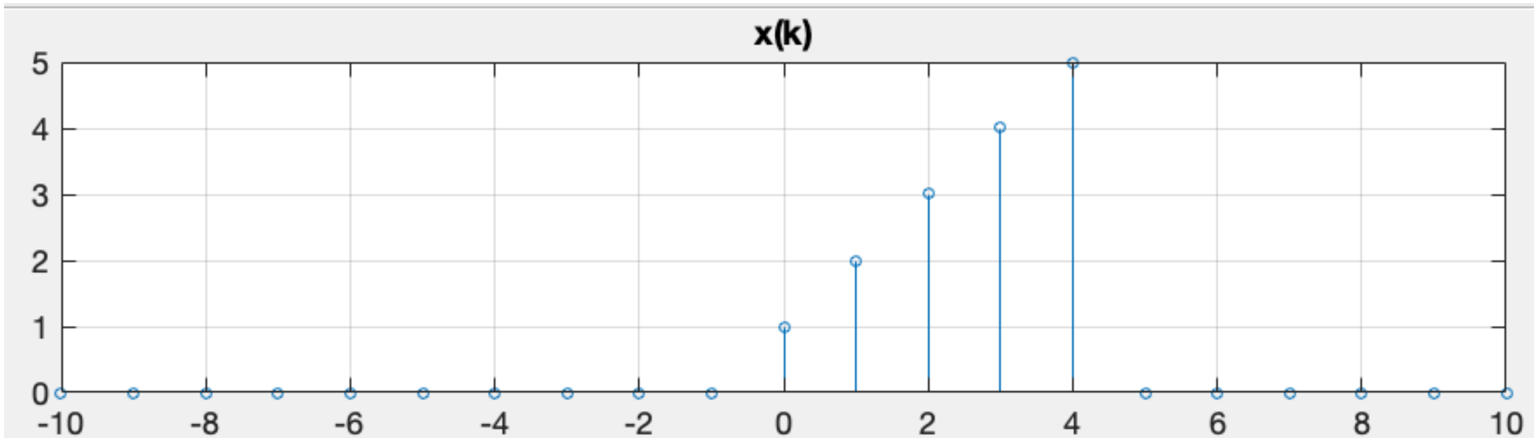


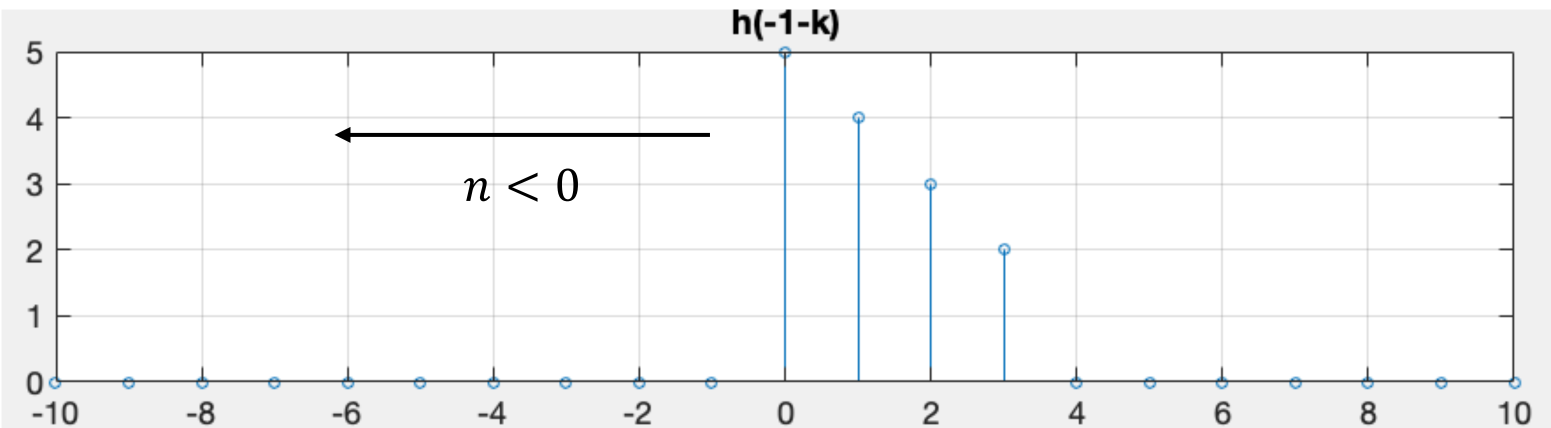


Discrete signal convolution: example

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

$n = -1$

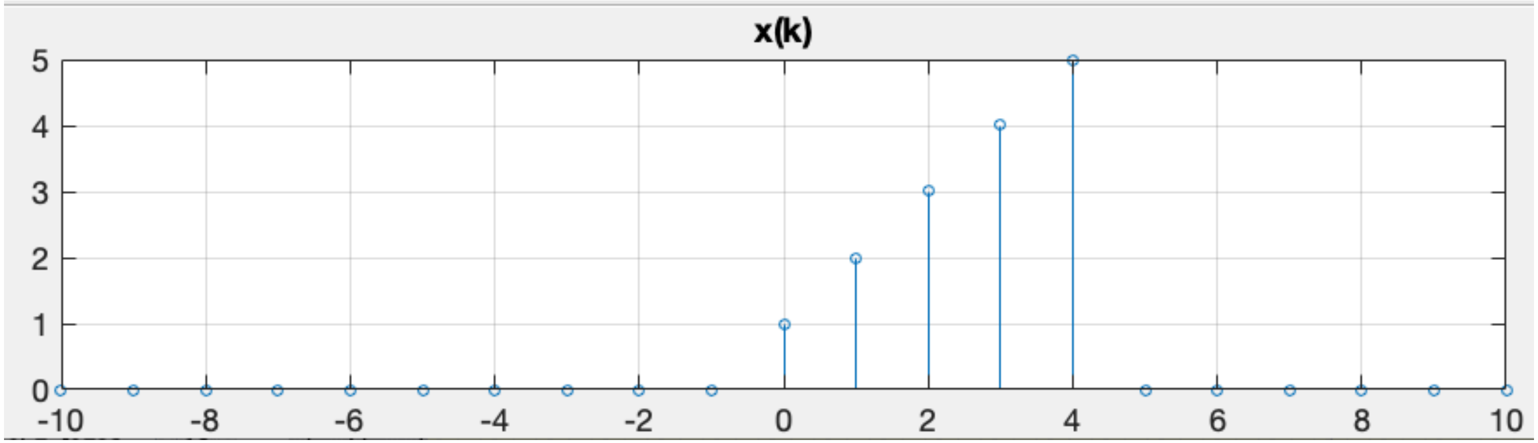


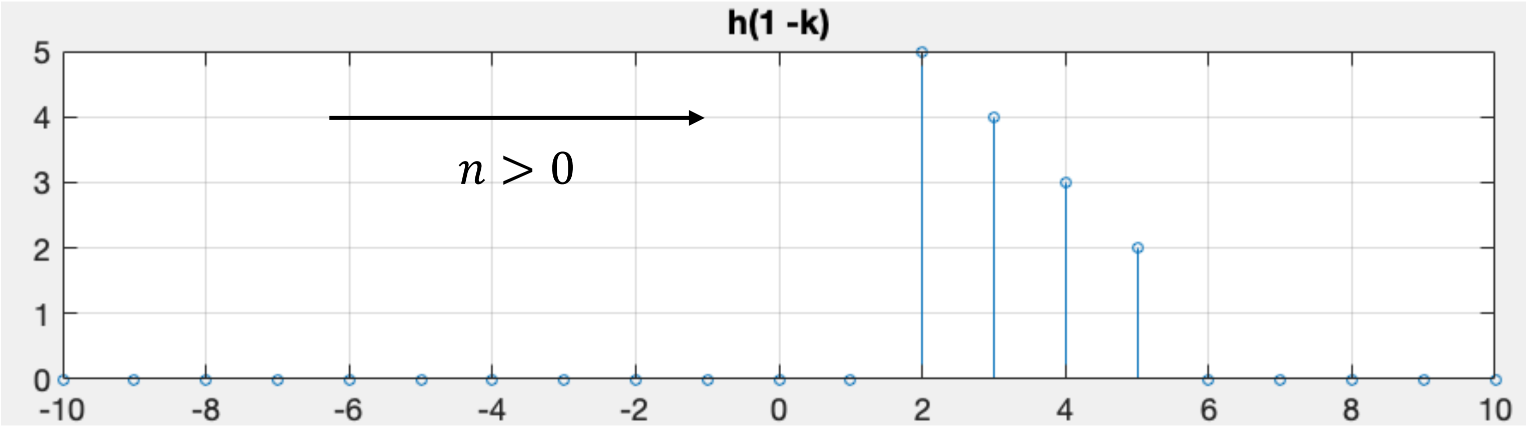


Discrete signal convolution: example

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

$n = 1$

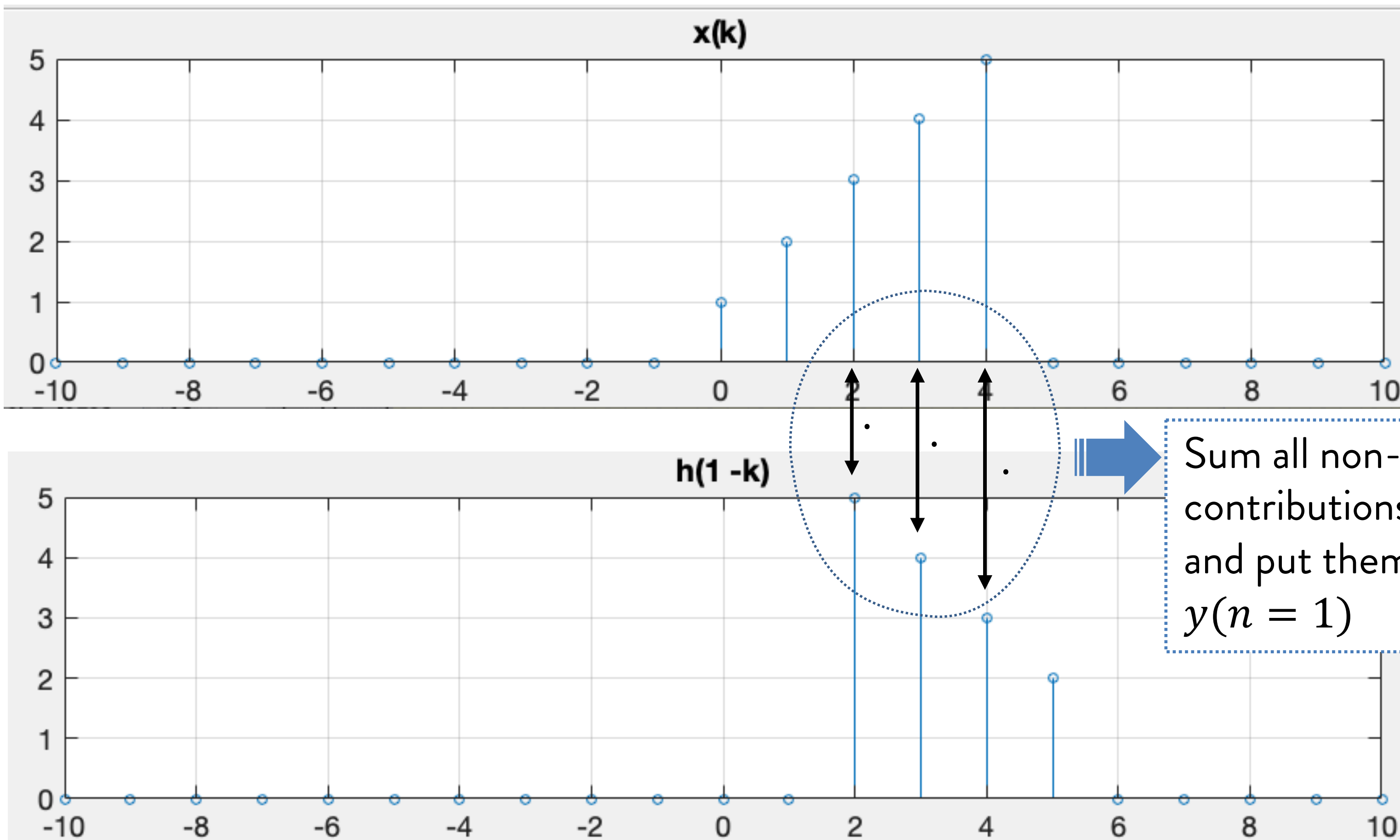




Discrete signal convolution: example

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

$n = 1$



Sum all non-zero contributions and put them in $y(n=1)$

Properties of convolution

- Commutativity: $x(n) * y(n) = y(n) * x(n)$
- Associativity: $(x(n) * y(n)) * z(n) = x(n) * (y(n) * z(n))$
- Distributivity: $(x(n) + y(n)) * z(n) = x(n) * z(n) +$

  $y(n) * \ z(n)$
- Convolution by pulse: $x(n) * \delta(n) = \ x(n)$
- Convolution by a shifted pulse : $x(n) * \delta(n-k) = \ x(n-k)$

Exercise 8: Convolution

- Given x(n) = [3, 11, 7, 0, -1, 4, 2] , n in [-3, 3]
- Given h(n) = [2, 3, 0, -5, 2, 1], n in [-1, 4]
- Define both signals for n in [-7,7].
- Compute y(n) as x(n) convolved with h(n), n in [-7, 7].
- Use also the MATLAB function ‘conv’.
- Which is the support of the convolution?

Operations on signals

- Discrete delay → $y(n) = x(n-k)$

- Moving average → $y(n) = \frac{1}{M}\sum_{m=0}^{M-1} x(n-m)$

↓

y(n) can be seen as the output of LTI systems

Exercise 9: LTI systems

- Given x(n) = [3, 11, 7, 0, -1, 4, 2] , n in [-3, 3]
- Create y(n) = x(n – 5), n in [0, 10], without using ‘circshift’ or ‘for’ loops.
- Create $y(n) = \frac{1}{3}\sum_{m=0}^{2} x(n-m)$, n in [0, 10], without using ‘circshift’ or ‘for’ loops.
- Hint: y(n) has the form of a convolution… (you can use MATLAB function ’conv’).