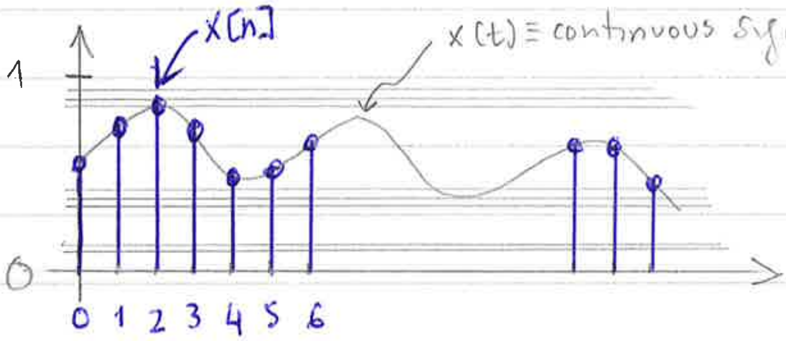
Biometric Recognition
SIGNALS IN 1D AND 2D

Lab exercise 1

* SIGNALS IN ONE DIMENSION (1D): SAMPLING *

2) discrete amplitude



t (continuous time)

n (sampling)

T = Sampling period

1) discrete time = one sample each T seconds to create x[n]

$$x(t) \xrightarrow{T} x[n] = x(nT)$$

continuous signal

discrete signal ($n = 0, 1, 2 \ldots$) integer

T = Sampling period

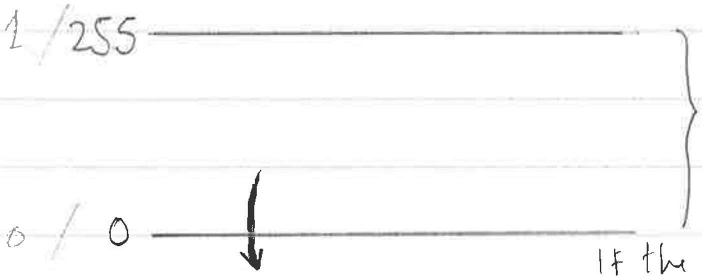
$F_S = \frac{1}{T}$ = sampling frequency

e.g. $F_S = 8000\ Hz = 8\ kHz$

$T = 1/8000 = 0.000125$ sec

2) discrete amplitude

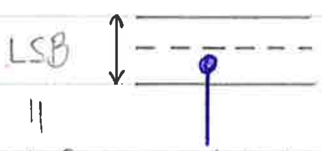
- only a finite number of values for x[n]
- Signal x(nT) at instant nT is discretized (i.e. rounded to the nearest possible value)



e.g. resolution of 8 bits (binary signal)

$2^8 = 256$ possible values (0 to 255)

If the input signal is between 0 and 1, then:



- distance between consecutive values is $\boxed{\frac{1}{2^8 - 1} = LSB}$
- average error in the discretization is $\boxed{e_q = \pm \frac{1}{2} LSB}$

LSB = Least Significant Bit (minimum change in the input signal required to guarantee a change in the discretized signal)

→ Signals from the "real" world change with time in a continuous way

$x(t)$ $\begin{cases} \text{- have values all the time} \\ \text{- can take any value between two limits, e.g } [0,1] \end{cases}$

→ this cannot be represented with a computer! (would consume infinite memory)

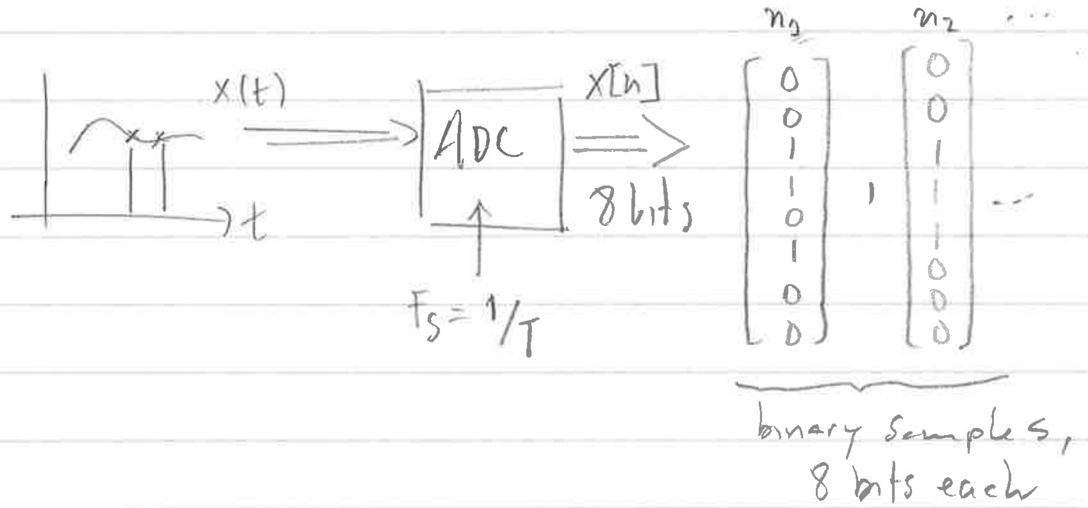
e.g. signals from $\begin{cases} \text{microphone (voice)} \\ \text{sensors (temperature, pressure...)} \\ \vdots \end{cases}$

→ Making a signal discrete both in amplitude and time ⟷ "sampling" (ADC)

↑ Analog to Digital Conversion

$x[n] = [x, x, x \; \ldots \; x]$ ≡ vector (array of values)

$x[0]$, $x[1]$, $x[2]$

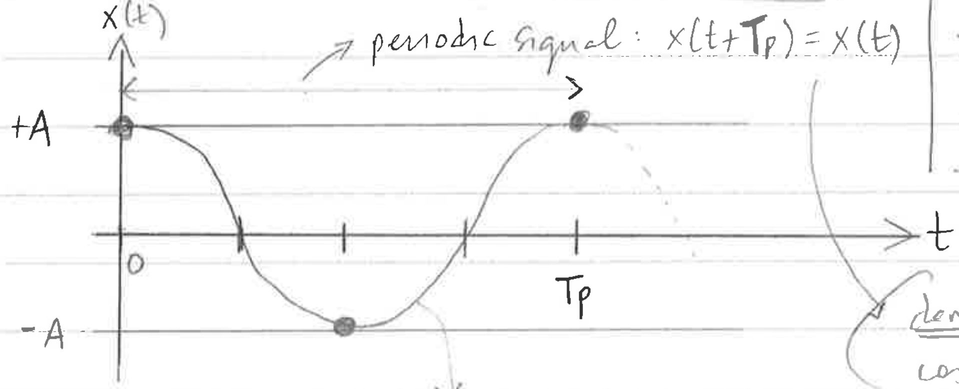


* COSINE / SINE SIGNALS * (SINUSOIDS)

$$x(t) = A \cdot \cos(\Omega t + \theta)$$

- A: amplitude
- Ω: frequency (rad/s)
- θ: phase angle (rad/s)

$$\boxed{\Omega = 2\pi \cdot F = \frac{2\pi}{T_p} \qquad F = \frac{1}{T_p} \text{ (hertzs)}}$$



$A \cdot \cos(\Omega t)$, with $\theta = 0$

$$\boxed{\begin{array}{lll} \Omega = \frac{2\pi}{T_p} & \text{(rad/s)} & \\ \Omega = 2\pi F & & \\ F = 1/T_p & \text{(Hz)} & = \text{frequency} \\ T_p & \text{[sec.]} & = \text{period} \end{array}}$$

frequency: number of oscillations per second

period: time until the signal has the same value

dem ($\theta = 0$):

$$\cos(\Omega(t+T_p)) = \cos\left(2\pi F\left(t + \frac{1}{F}\right)\right) = \cos(2\pi F t + 2\pi) = \cos(\Omega t)\cancelto{1}{\cos(2\pi)} - \sin(\Omega t)\cancelto{0}{\sin(2\pi)} = \cos(\Omega t)$$

Discretization of cosine signals

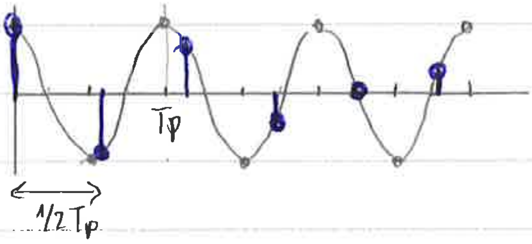
$$x(t) = A\cos(2\pi F \cdot t)$$

↓ take samples with frequency F_s (number of samples per second that I take), i.e. $\left\{ t = \frac{n}{F_s},\ t = n \cdot T_s \right\}$ or every $T_s = \frac{1}{F_s}$ seconds

$$x[n] = A\cos\left(2\pi \frac{F}{F_s} n\right) = A\cos(2\pi f n) \qquad \boxed{f = \frac{F}{F_s}}$$

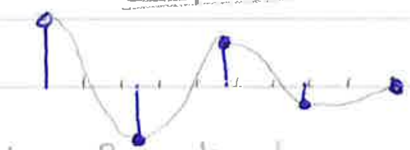
$f \equiv$ "frequency" of the discretized signal
- $F \equiv$ frequency of original (continuous) signal
- $F_s \equiv$ sampling frequency to discretize the signal

Is there a relation between F (frequency of the signal I want to discretize) and Fs (sampling frequency that I choose to discretize) ??



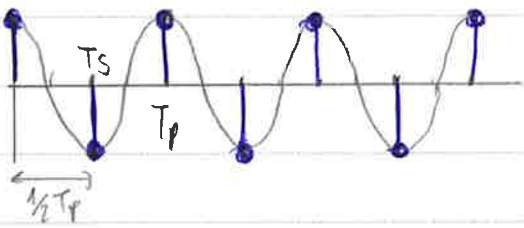
$$\frac{F_s < 2F}{T_s > \frac{1}{2} T_p}$$

bad representation!



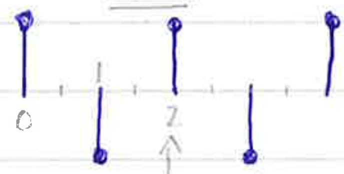
lose information!

limit!

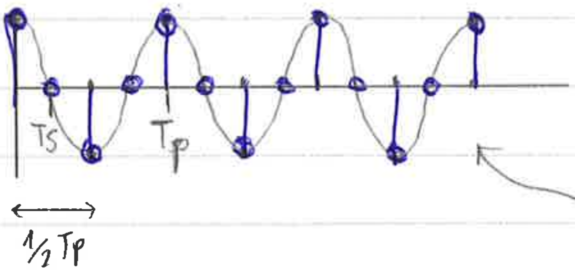


$$\frac{F_s = 2F}{T_s = \frac{1}{2} T_p}$$

N = 2

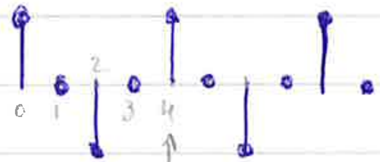


periodic each N = 2 samples



$$\frac{F_s > 2F}{T_s < \frac{1}{2} T_p}$$

e.g. $T_s = \frac{T_p}{4} \Rightarrow F_s = 4F \Rightarrow N = 4$



periodic each N = 4 samples

therefore...

Relation: $\boxed{F_s > 2F}$ in order to be able to "keep" the relevant information in the sampled sinusoid

F_s: Sampling frequency that I choose

F: Frequency of the signal that I want to discretize

And for this reason: $\boxed{f = \frac{F}{F_s} \in [0, \frac{1}{2}]}$

And the sampled signal is periodic with period $\boxed{N = \frac{1}{f} = \frac{F_s}{F}}$

Compromise with the sampling frequency for signals of the "real world"

- Voice: $F_{MAX} \simeq 4\,kHz \Rightarrow$ sampling $F_s = 8\,kHz$ in telephony
- Hearing range of humans: $F_{MAX} \simeq 20\,kHz \Rightarrow$ Sampling $F_s = 44{,}1\,kHz > 20\,kHz$ used in digital audio (CD) or $F_s = 48\,kHz$ in professional music

Example (lab exercise)

$$f = \frac{F}{F_s} = \frac{200}{10000} = \frac{2}{100}$$

```
>> n = 0:199;
   F = 200;      freq of the sinusoid
   Fs = 10000;   sampling rate
```

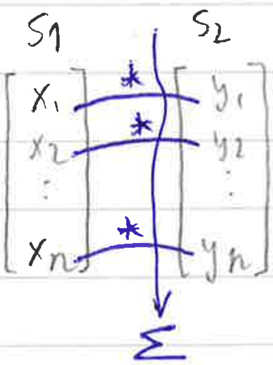
N = 50 (see figure obtained in your exercise!)

```
>> cos = 1 * cos(2*pi*(F/Fs)*(n+Ø))
   figure(21); subplot(311); stem(n, cos)
```

Show in the same figure
1) change in frequency of the sinusoid F=300; F=100
2) change in phase angle Ø
3) change in amplitude A=2, A=0.5

* SCALAR PRODUCT: COMPARISON OF TWO SIGNALS *

- Compare two signals with the normalized scalar product (must have the same length!)



1) element-wise multiplication (.* in Matlab)
2) Sum of all multiplications

$$\boxed{\langle S_1, S_2 \rangle = x_1 y_1 + x_2 y_2 + \dots + x_n \cdot y_n}$$ SCALAR PRODUCT

the result is a number!

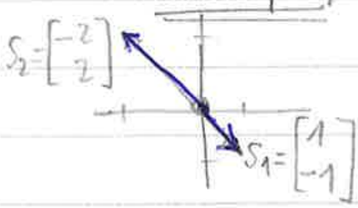
- To make the result independent of the "energy" of the signal, it is divided by the "length"/"norm" of S_1 and S_2

$$\|S_1\| = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$$
$$\|S_2\| = \sqrt{y_1^2 + y_2^2 + \dots + y_n^2}$$

$$\Rightarrow \boxed{\frac{\langle S_1, S_2 \rangle}{\|S_1\| \, \|S_2\|}}$$ NORMALIZED SCALAR PRODUCT

and due to this normalization, the result is always in [-1, 1]

Examples



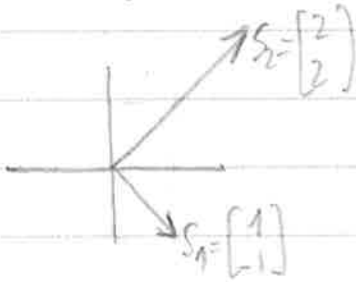
$$\boxed{\langle S_1, S_2\rangle = -2\cdot 1 + 2\cdot(-1) = -2-2 = -4}$$

$$\|S_1\| = \sqrt{(-2)^2+2^2} = \sqrt{4+4} = \sqrt{8}$$

$$\|S_2\| = \sqrt{1^2+1^2} = \sqrt{2}$$

$$\boxed{\frac{\langle S_1, S_2\rangle}{\|S_1\|\,\|S_2\|} = \frac{-4}{\sqrt{8}\sqrt{2}} = \frac{-4}{\sqrt{16}} = \frac{-4}{4} = -1}$$

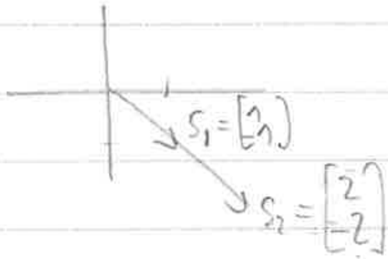
completely opposite vectors



$$\langle S_1, S_2\rangle = 2-2 = 0$$

$$\boxed{\frac{\langle S_1, S_2\rangle}{\|S_1\|\,\|S_2\|} = 0}$$

perpendicular



$$\langle S_1, S_2\rangle = 2+2 = 4$$

$$\|S_1\| = \sqrt{1+1} = \sqrt{2}$$

$$\|S_2\| = \sqrt{2^2+2^2} = \sqrt{8}$$

$$\boxed{\frac{\langle S_1, S_2\rangle}{\|S_1\|\,\|S_2\|} = \frac{4}{\sqrt{2}\sqrt{8}} = 1}$$

Same direction

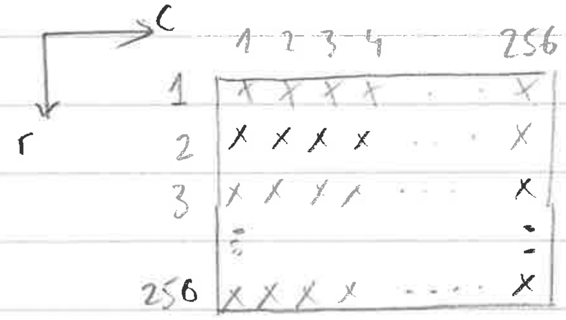
Another interpretation of $\frac{\langle S_1, S_2\rangle}{\|S_1\|\,\|S_2\|}$ is as $\cos(\varphi)$

where φ is the angle between S_1 and S_2

* SIGNALS IN TWO DIMENSIONS (2D) : IMAGES *

image ⟷ matrix (array in two dimensions)
data structure

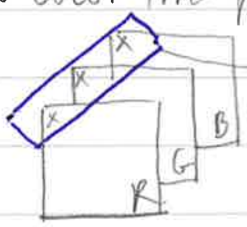
Example 256x256 photo



X = point of the image (pixel)

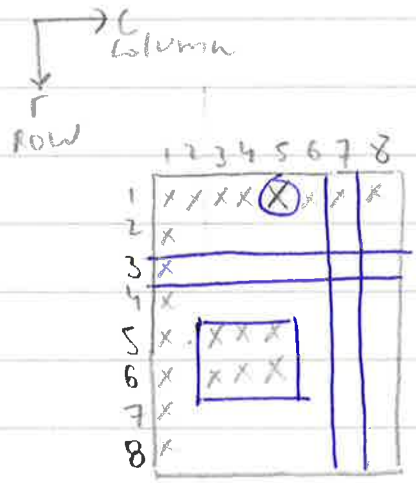
black white

- Gray-scale images : $x \in [0, 255]$
- Color images (RGB) : three values per pixel
  i.e. "three images"



each pixel = $\begin{bmatrix} R \\ G \\ B \end{bmatrix}$ with $R, G, B \in [0, 255]$

In Matlab : how to index (access to pictures)



I
8x8

One pixel : $I(r, c)$ e.g. $I(1, 5)$

A row of pixels : $I(r, :)$ e.g. $I(3, :)$

A column : $I(:, c)$ e.g. $I(:, 7)$

An area of pixels : $I(r_1 : r_2, c_1 : c_2)$ e.g. $I(5:6, 3:5)$

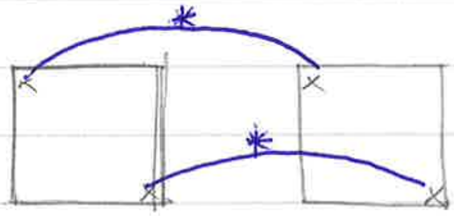
In Matlab: assign values to an image

I = ones(8,8) ⇒ image of 8x8 with all 1's

$\underbrace{\text{I(5:6, 3:5)}}_{\text{area}}$ = zeros(2,3) ⇒ modify an area and set it to all 0's

* COMPARISON OF IMAGES WITH SCALAR PRODUCT *

Same idea as before



B_1 B_2

1) pixel wise multiplication
2) sum of all products

$$\boxed{\frac{\langle B_1, B_2 \rangle}{\|B_1\| \cdot \|B_2\|}}$$

In Matlab

$\langle B_1, B_2 \rangle$ ⇒ sum(sum(B1.*B2))

$\|B_1\|$ ⇒ norm(B1)

Example

```
I = ones(8,8);
I1 = I
I1(5:6, 3:5) = zeros(2,3);
Skalprod = sum(sum(I.*I1));
NormSP = SkalProd/norm(I)/norm(I1);
```