

# Riconoscimento e Recupero dell'Informazione per Bioinformatica

## Lab. 01 – Introduzione a Matlab

03/10/2016

# Qualche info sul corso

- Docente: Pietro Lovato
  - Ricevimento: concordato via mail
  - email: [pietro.lovato@univr.it](mailto:pietro.lovato@univr.it)
- Dove: Aula Alfa
- Orario: Lunedì 10:30 – 13:30
- Tutor: Matteo Denitto
- Il corso: Implementazione in Matlab di algoritmi di Pattern Recognition visti nella parte di teoria

# Esame finale

- Non obbligatorio per la parte di lab
- Esercizio da risolvere al calcolatore, possibilità di prendere un punto bonus da sommare al voto finale dell'esame di teoria

# Breve introduzione alla Pattern Recognition

# Pattern Recognition

- Punto di partenza: l'uomo e la Pattern Recognition



Cos'è questa?



In che città mi trovo?

# Pattern Recognition



C'è una Vespa gialla?

# Pattern Recognition



Quanti tipi di fiori ci sono?



# Pattern Recognition

- Il processo che ci porta a rispondere a queste domande si chiama Pattern Recognition



Riconoscere che si tratta di una mela



Identificare l'oggetto più importante nella foto (l'Arena) ed associarlo alla città di Verona



Trovare nell'immagine tutti gli oggetti di tipo «vespa», ed identificare se ce n'è una di colore giallo




Riconoscere i fiori e distinguerli in due diverse tipologie (anche non sapendo che fiori sono)



# Pattern Recognition

Più in generale:

1. Prendere in ingresso un insieme di dati (un'immagine)  **pattern**: il dato che viene analizzato, l'entità di interesse
2. Effettuare un'analisi di tali dati per rispondere ad una domanda tipicamente legata al concetto di categoria o classe (che tipo di oggetto è? Quante categorie di oggetti ci sono? E' presente un dato di una certa categoria? Quali sono gli oggetti simili a questo?)

# Pattern Recognition

- E' un problema che l'uomo risolve facilmente (tramite processi complicati non ancora completamente chiari)
- La prospettiva informatica: realizzare sistemi **AUTOMATICI** di Pattern Recognition
  - Sistemi che siano in grado di risolvere problemi di Pattern Recognition senza l'intervento dell'uomo
  - Il problema viene studiato da molti anni, anche se è tipicamente molto difficile!

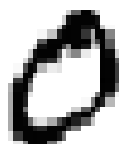
Esempio: riconoscere caratteri scritti a mano



Facile per l'uomo, difficile per il calcolatore!

# Esempio

Quello che vede l'uomo

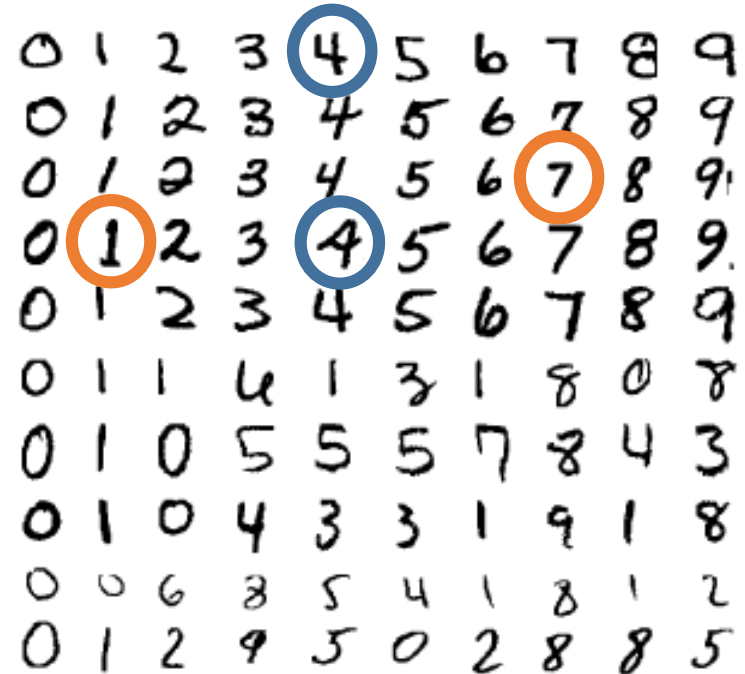


Quello che vede il calcolatore



# Esempio: perché è difficile per un calcolatore?

- C'è molta variabilità:
- Oggetti della stessa classe possono essere diversi
- Oggetti di classi diverse possono essere molto simili



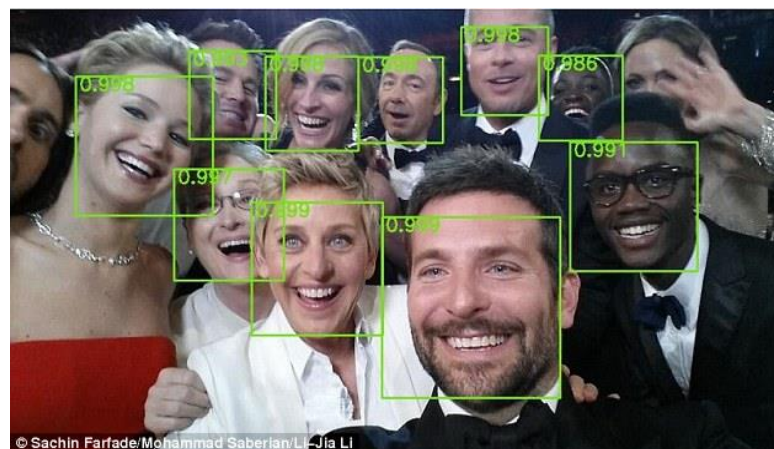
Demo:

<https://webdemo.myscript.com/#/demo/write>

## Altri esempi classici

Distinguere diverse persone sulla base del volto

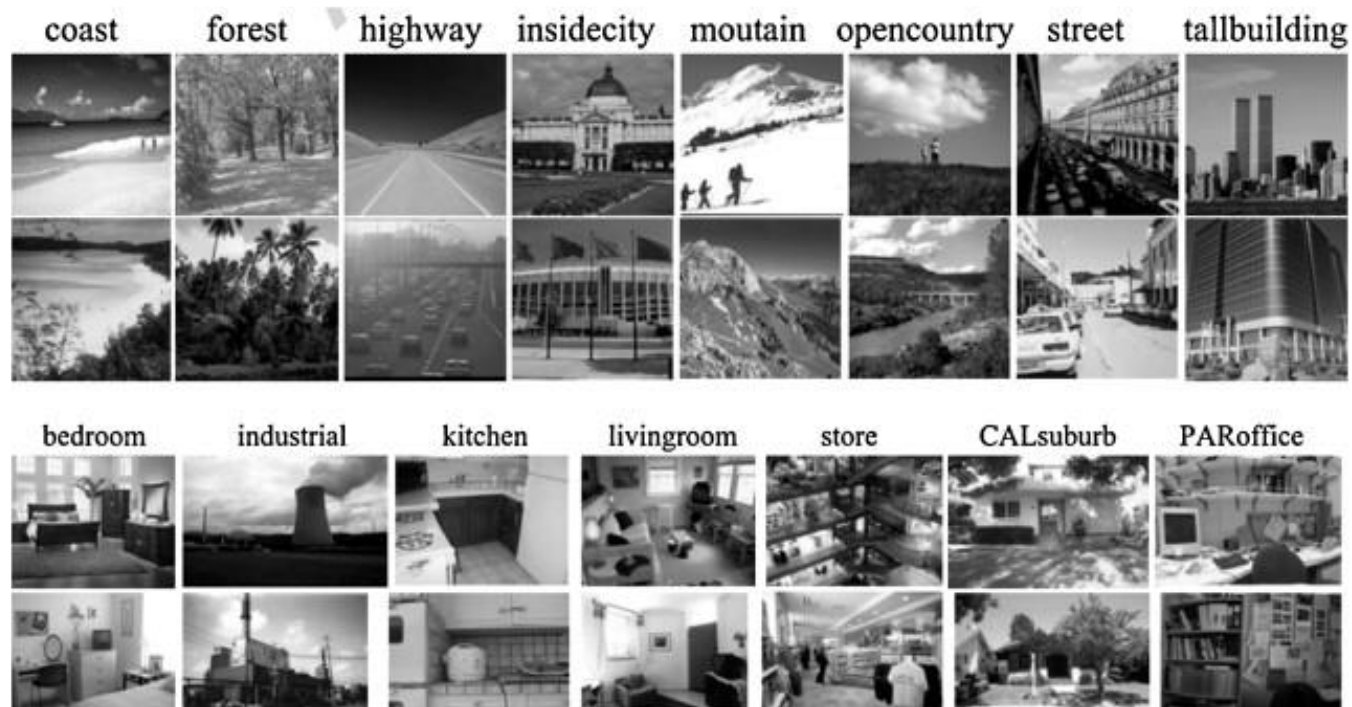
Pattern: la parte dell'immagine che contiene la faccia



Demo: <https://skybiometry.com/demo/face-grouping/>

# Altri esempi classici

## Riconoscimento di scene a partire da immagini

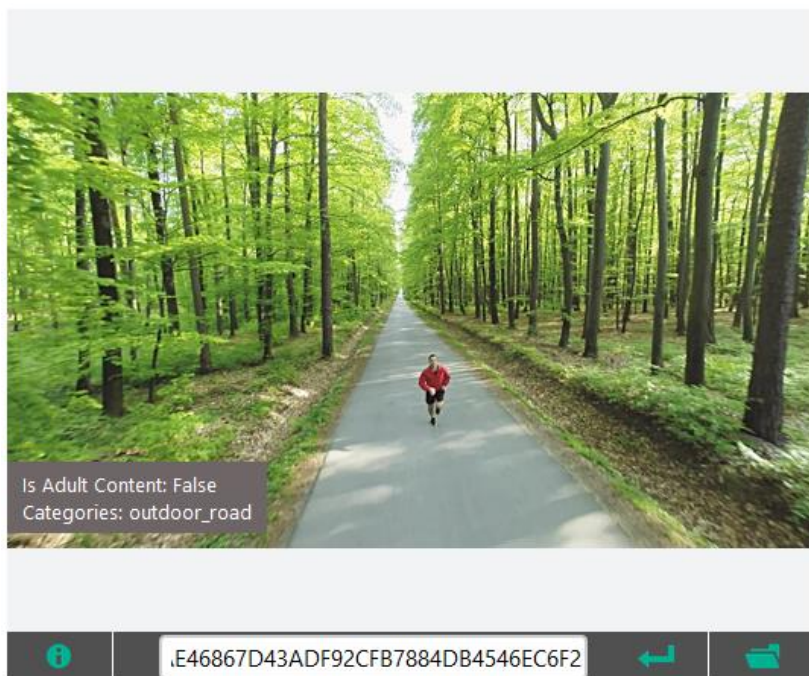


Demo: <http://places.csail.mit.edu/demo.html>



# Altri esempi classici

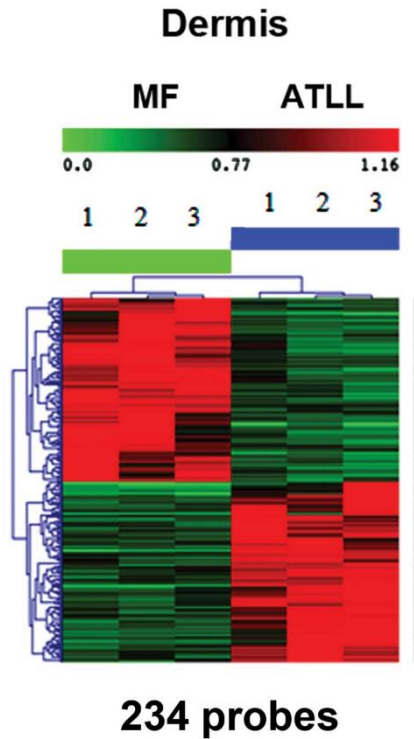
Più in generale: *image understanding* (il task più difficile!)



Feature Name	Value
Description	{ "type": 0, "captions": [ { "text": "a man walking down a path in the forest", "confidence": 0.6227632795903301 } ] }
Tags	[ { "name": "grass", "confidence": 0.9999678134918213 }, { "name": "tree", "confidence": 0.9997695088386536 }, { "name": "outdoor", "confidence": 0.99479585886001 }, { "name": "path", "confidence": 0.9891777038574219 }, { "name": "plant", "confidence": 0.9248211111111111 }, { "name": "trail", "confidence": 0.9000771045684814 }, { "name": "red", "confidence": 0.6111111111111111 }, { "name": "wooded", "confidence": 0.6415477991104126 }, { "name": "forest", "confidence": 0.6111111111111111 }, { "name": "wood", "confidence": 0.4206943213939667 }, { "name": "dirt", "confidence": 0.4206943213939667 } ]
Image Format	jpg?v=1&c=IWSAsset&k=2&d=62CA815BF81CE4804839F66EF06C533ED1DB44E5599D888AE46867D
Image Dimensions	640 x 360
Clip Art Type	0 Non-clipart
Line Drawing	0 Non-LineDrawing

Demo: <https://www.microsoft.com/cognitive-services/en-us/computer-vision-api>

# Pattern Recognition e Bioinformatica



Riconoscere quali geni  
sono correlati ad una  
malattia

Description	Actual binding interface	Predicted binding interface
Sindbis virus capsid protein (dimer)		
Triose Phosphate Isomerase (dimer)		
Hemoglobin $\beta$ Chain (interface with $\alpha$ chains)		

Identificare il sito attivo  
di una proteina

... e molte altre applicazioni  
che vedremo in questo corso

# In questo corso: Pattern Recognition in Matlab

- Matlab (*MATrix LABoratory*) è un software per il calcolo scientifico, particolarmente sviluppato per quanto riguarda la gestione e l'elaborazione di vettori e matrici.
- E' un linguaggio interpretato, ovvero ogni linea di un programma Matlab viene letta, interpretata, ed eseguita sul momento.

# Introduzione a Matlab

- Matlab ha un'interfaccia grafica interattiva e una linea di comando (prompt), sulla quale si possono scrivere dei comandi

```
>> 2+2
```

```
ans =
```

```
4
```

# Dichiarazione di variabili

La creazione di variabili in Matlab segue le seguenti regole:

- Tutte le variabili sono **matrici**
- Non si dichiara il tipo di variabile

## Dichiarazione di variabili (2)

```
>> a = 5                % variabile scalare (1x1)
>> b = [4 6]           % vettore riga (1x2)
>> c = [-5; 2]          % vettore colonna (2x1)
>> d = [2 3; -1 7]      % matrice quadrata (2x2)
```

### Operatori importanti:

- Separatori di riga: punto e virgola (;)
- Separatori di colonna: spazio bianco o virgola (,)

## Dichiarazione di variabili (3)

- **NOTA:** Matlab stampa il risultato dell'operazione, a meno che il comando non sia seguito da un punto e virgola.

```
>> a = 2+2      % stampa a video il risultato  
>> a = 2+2;    % non stampa
```



# Workspace

Per avere informazioni sulle variabili che sono state inizializzate:

- Tramite interfaccia grafica, controllando la finestra / scheda “workspace”

## Workspace (2)

### Alcuni comandi utili:

```
>> who          % visualizza tutte le variabili  
                % definite dall'utente  
  
>> whos        % visualizza tutte le variabili  
                % con indicata la loro  
                % dimensione  
  
>> [r,c] = size(A) % dimensioni di A  
  
>> clear A      % cancella la variabile A  
  
>> clear all    % cancella tutte le variabili
```

# Operazioni fondamentali

- Importante: Le operazioni fondamentali sono definite fra *matrici*

```
>> C = A + B
```

Somma fra matrici:  $C_{i,j} = A_{i,j} + B_{i,j}$

```
>> C = A * B
```

Prodotto fra matrici:  $C_{i,j} = \sum_k A_{i,k} B_{k,j}$

```
>> C = A / B
```

Divisione fra matrici:  $C = AB^{-1}$

```
>> C = A^3
```

Elevamento a potenza:  $C = A * A * A$

# Operazioni matriciali

Qualche considerazione:

- Le operazioni matriciali sono ben definite solo se le matrici hanno dimensioni compatibili.
- Matlab stampa un messaggio di errore ogni volta che le dimensioni delle matrici non sono corrette rispetto all'operazione che si vuole eseguire.

# Operazioni elemento per elemento

- Per eseguire operazioni tra matrici *elemento per elemento* bisogna aggiungere un punto davanti all'operatore.

```
>> C = A .* B
```

Prodotto:  $C_{i,j} = A_{i,j}B_{i,j}$

```
>> C = A ./ B
```

Divisione:  $C_{i,j} = \frac{A_{i,j}}{B_{i,j}}$

```
>> C = A .^ 3
```

Elevamento a potenza:  $C_{i,j} = A_{i,j}^3$

## Altre operazioni importanti

In questo corso, useremo molto le seguenti funzioni:

```
>> sum(A)      % somma gli elementi di A
```

```
% colonna per colonna
```

```
>> prod(A)     % moltiplica gli elementi
```

```
% di A colonna per colonna
```

```
>> mean(A)     % media dei numeri in ogni
```

```
% colonna di A
```

## Help in linea

Matlab offre un help in linea molto completo.

```
>> help comando
```

Per avere una spiegazione più dettagliata:

```
>> doc comando
```



## Estrazione di sotto-matrici

- Una volta definita una matrice, si possono utilizzare le seguenti operazioni sulle *sotto-matrici* di A:

```
>> A (2, 3)
```

Estrae l'elemento  $A_{2,3}$

```
>> A (:, 3)
```

Estrae la colonna  $[A_{1,3}; \dots; A_{n,3}]$

```
>> A (1:4, 3)
```

Estrae la sotto-colonna  $[A_{1,3}; \dots; A_{4,3}]$

```
>> A (end, :)
```

Estrae la sotto-riga  $[A_{n,1}, \dots, A_{n,m}]$

# Manipolazione di sotto-matrici

- E' possibile modificare i valori di una matrice nel seguente modo (es. A matrice 3x3):

```
>> A(1,1) = 2 % modifico l'elemento 1,1
```

```
>> A(:,3) = [2; 3; 2] % modifico la colonna 3
```

```
>> A(1:2,3) = [0; 0] % sotto-colonna 3
```

```
>> A(1:2, 2:3) = [1 1; 1 0] % sotto-matrice
```

## Definizione di vettori particolari

- Vettori di punti equispaziati con dimensione arbitrariamente grande

```
>> x = [0:0.1:1]
```

```
% x = [inizio:passo:fine]
```

```
>> y = linspace(0,1,10)
```

```
% linspace(inizio, fine, lunghezza vettore)
```

# Definizione di matrici particolari

```
>> A = eye(n) % matrice identità nxn
```

```
>> A = zeros(n,m) % matrice di soli zeri nxm
```

```
>> A = ones(n,m) % matrice di soli uni nxm
```

```
>> A = rand(n,m) % matrice aleatoria (valori  
% compresi tra 0 e 1) nxm
```

# Funzioni matriciali

```
>> C = A'
```

Trasposta di A  $C_{i,j} = A_{j,i}$

```
>> C = inv(A)
```

Inversa di A (matrici quadrate)  $C = A^{-1}$

```
>> C = det(A)
```

Determinante di A (matrici quadrate)

```
>> n = norm(A)
```

Norma 2 di A

```
>> [V,D] = eig(A)
```

Autovalori (e autovettori) di A

Per una lista delle funzioni matematiche elementari si veda:

```
>> help elfun
```

# Script

Matlab permette di scrivere dei programmi mediante *script*

- Uno script è un insieme di comandi Matlab.
- Gli script sono semplici file di testo con estensione “.m” (consiglio: utilizzare l'editor di Matlab!)
- Per eseguire uno script bisogna che la cartella dove si trova Matlab sia la stessa dove è contenuto il file .m
- Per eseguire uno script è possibile:
  - Digitare il nome del file (senza “.m”) nella linea di comando
  - Premere F5 dall'editor Matlab
  - Cliccare su “Run” nella barra in alto

## Esercizio

- Capire il contenuto dello script `Lezione1Lab_es1.m` e provare ad eseguire l'esercizio proposto.
- Formula della distanza euclidea fra due vettori  $a$  e  $b$  di lunghezza  $n$ :

$$d = \sqrt{\sum_{i=1}^n (a_i - b_i)^2}$$

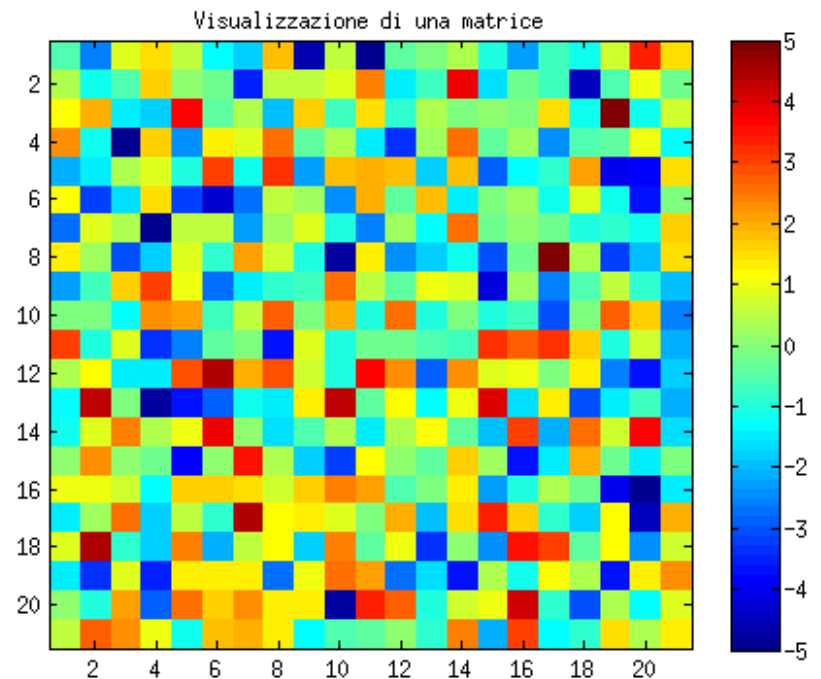


# Visualizzazione di vettori e matrici

- Visualizzazione intuitiva di matrici:

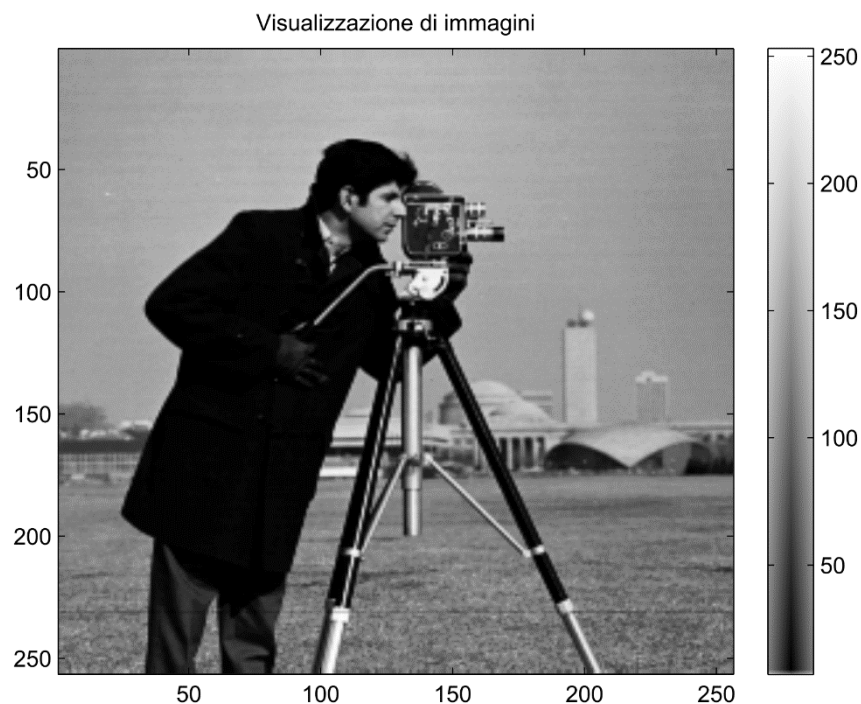
```
>> imagesc(A)
```

- Di default, il colore “blu scuro” corrisponde al valore più basso della matrice



## Visualizzazione di vettori e matrici (2)

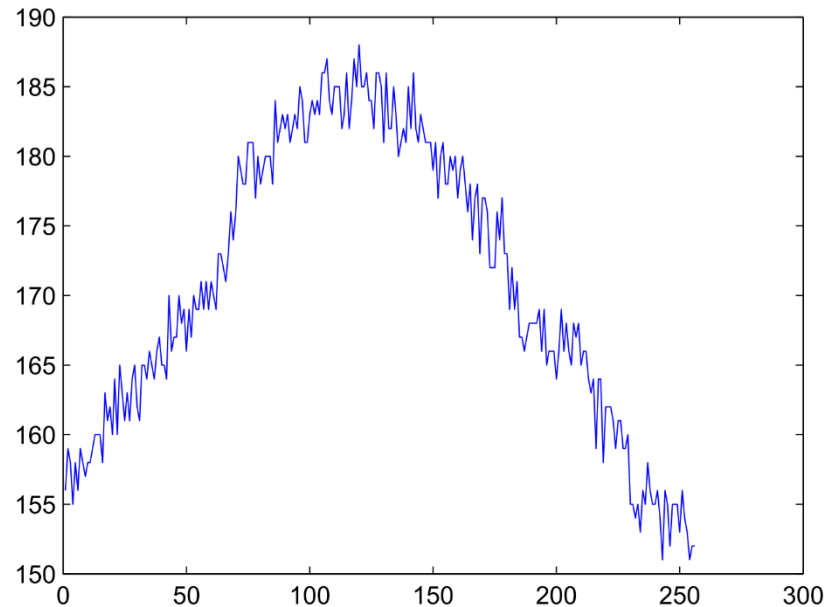
- Un'immagine non è altro che una matrice
- In particolare, un'immagine in bianco e nero è
- una matrice in cui
- ogni cella indica
- la “luminosità” del
- pixel corrispondente



# Visualizzazione di vettori

- Comando plot: dati due vettori  $x$  e  $y$  ***della stessa dimensione***, vengono visualizzati sul piano 2D
- Un punto generico  $i$  corrisponde alla coppia

$$(x_i, y_i)$$



# Esercizio

- Capire il contenuto dello script `Lezione1Lab_es2.m` e provare ad eseguire l'esercizio proposto