

# Scientific Computing

Prof. Giulio GIUNTA

Prof. Mariarosaria RIZZARDI

(12 CFU)

annual Course

- Part I: Data Science and Simulation

first semester

- Part II: Geometrical Mappings and Transforms

second semester

- Single final Exam. Upon request, students can take a test on Part 1, at the end of the first semester, and then a test on Part 2, at the end of the second semester

# Scientific Computing

Prof. Giulio GIUNTA

giulio.giunta@uniparthenope.it

Prof. Mariarosaria RIZZARDI

rizzardi@uniparthenope.it

e-learning platform:

<http://elearning.uniparthenope.it>



# Scientific Computing

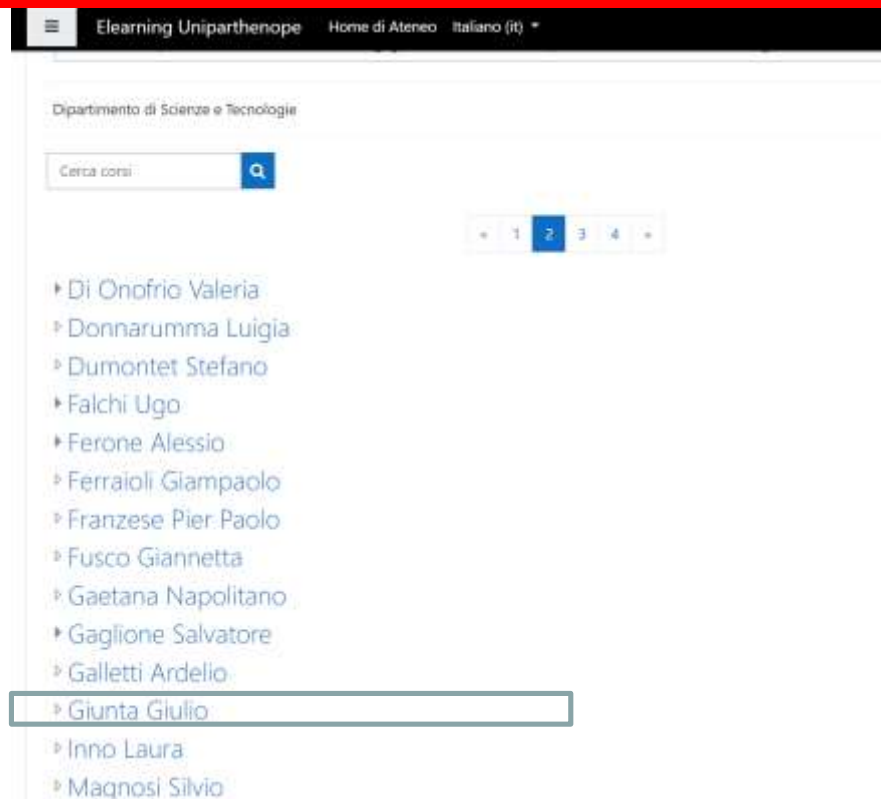
Prof. Giulio GIUNTA

giulio.giunta@uniparthenope.it

Prof. Mariarosaria RIZZARDI rizzardi@uniparthenope.it

e-learning platform:

<http://elearning.uniparthenope.it>




# Scientific Computing

Prof. Giulio GIUNTA      giulio.giunta@uniparthenope.it  
Prof. Mariarosaria RIZZARDI      rizzardi@uniparthenope.it

e-learning platform:

<http://elearning.uniparthenope.it>

 Elearning Uniparthenope   Home di Ateneo   Italiano (it) ▼


Giunta Giulio

Home / Corsi / Scuola Interdipartimentale delle Scienze dell'Ingegneria e della Salute / Dipartimento di Scienze e Tecnologie / Giunta Giulio

Categorie di corso:

Scuola Interdipartimentale delle Scienze dell'Ingegneria e della Salute / Dipartimento di Scienze e Tecnologie / Giunta Giulio

Giunta Giulio

scientific computing 

# Scientific Computing

Prof. Giulio GIUNTA

giulio.giunta@uniparthenope.it

Prof. Mariarosaria RIZZARDI rizzardi@uniparthenope.it

e-learning platform:

<http://elearning.uniparthenope.it>

The screenshot displays the Elearning Uniparthenope website interface. At the top, a navigation bar includes a menu icon, the site name 'Elearning Uniparthenope', and links for 'Home di Ateneo' and 'Italiano (it)'. Below this, the main heading 'Elearning UniParthenope' is followed by a breadcrumb trail: 'Home / Corsi / Cerca / scientific computing'. A search bar contains the text 'scientific computing' with a magnifying glass icon. Below the search bar, it states 'Risultati della ricerca: 3'. Three search results are listed, each with a heart icon and a lock icon. The first result, 'SCIENTIFIC COMPUTING', is highlighted with a blue border and lists 'Docente: Giulio Giunta' and 'Docente: Mariarosaria Rizzardi'. The second result, 'SCIENTIFIC COMPUTING - part 2', lists 'Docente: Giulio Giunta', 'Docente: NUNZIO NAPOLITANO', and 'Docente: Mariarosaria Rizzardi'. The third result, 'SCIENTIFIC COMPUTING - part 2 (EN)', lists 'Docente: Mariarosaria Rizzardi'. Each result also includes a category label: 'Categoria: INFORMATICA APPLICATA (MACHINE LEARNING E BIG DATA)' for the first two, and 'Categoria: Rizzardi Mariarosaria' for the third.

Elearning Uniparthenope Home di Ateneo Italiano (it)

Elearning UniParthenope

Home / Corsi / Cerca / scientific computing

scientific computing

Risultati della ricerca: 3

SCIENTIFIC COMPUTING

Docente: Giulio Giunta  
Docente: Mariarosaria Rizzardi

Categoria: INFORMATICA APPLICATA (MACHINE LEARNING E BIG DATA)

SCIENTIFIC COMPUTING - part 2

Docente: Giulio Giunta  
Docente: NUNZIO NAPOLITANO  
Docente: Mariarosaria Rizzardi

Categoria: INFORMATICA APPLICATA (MACHINE LEARNING E BIG DATA)

SCIENTIFIC COMPUTING - part 2 (EN)

Docente: Mariarosaria Rizzardi

Categoria: Rizzardi Mariarosaria

# Scientific Computing

Prof. Giulio GIUNTA      giulio.giunta@uniparthenope.it  
Prof. Mariarosaria RIZZARDI   rizzardi@uniparthenope.it

Let's give a look to the teaching material on  
the Course page of the [e-learning platform](#)

# Scientific Computing - Part 1

## Teaching material on Microsoft Teams

All the lessons of the past academic year 2021/22 have been recorded (in English) and can be seen via Microsoft TEAMS - Team code of the lessons of prof. Giunta: **5612r12 (folder: File, Recordings)**

On that Team you will also find the slides of the lessons and the Online tutorial as Matlab live scripts **(folder: File, Course Material)**

Finally, on that Team you will find all the Matlab function developed in the Course **(folder: File, Course Material, Matlab files)** and even information on how to install Matlab on your laptop (enrolled students can download Matlab for free)

# Scientific Computing - Part 1

## reference Books

Gilbert STRANG

**Linear Algebra and Learning from Data**

Wellesley-Cambridge Press 2019

Cleve MOLER

**Numerical Computing with MATLAB**

SIAM, 2004

<http://www.mathworks.com/moler/>

You can even download NCM book and software from the e-learning platform



# Scientific Computing - Part 2

## reference Book

Marc Peter Deisenroth, A. Aldo Faisal, Cheng Soon Ong

### **Mathematics for Machine Learning**

Wellesley-Cambridge Press 2019

**Download from [github](#)**

Mariarosaria RIZZARDI

### **Sperimentare la Matematica con MATLAB**

Liguori Editore, 2007

**Introduction to Matlab (in Italian)**

**Download from the [e-learning platform](#)**

# Topics - Part I

- Numerical linear algebra: norms, subspaces, projections, eigenvalues / eigenvectors
  - QR, SVD factorizations
  - Applications in Data Science: dimensionality reduction, data analysis and PCA, bioinformatics, semantic indexing of texts, image analysis
- 
- Linear Algebra and Search Engines: Google's Page Rank Algorithm
  - Markov Chains (CM) and random processes
  - Probabilistic interpretation of the Page Rank algorithm with CM
  - CM in Data Science

# Topics - Part I

- Solving non-linear systems
  - Newton and fixed point methods
- Calculation of maxima and minima of functions of several variables, with hints to problems with constraints
  - Gradient descent methods, Newton like methods, and Simulated annealing
  - Applications to data analysis and modeling

3D graphics in Matlab

# Topics - Part I

- Finite differences
- Numerical resolution of ODE
- Initial value problems, explicit and implicit methods
- Boundary value problems
- Applications: COVID-19 epidemic model

- Numerical resolution of PDEs with finite difference methods
- Stationary equations (Laplace, Poisson)

# Topics - Part II

- Spaces and Transformations
- In-depth study on eigenvalues and autobvectors
- Conformal transformations
- Applications to computational graphics

- Least squares approximation in the continuous case
- Non-linear least squares
- Applications to modeling

- Insights on the Fourier Transform
- Transform 1D and 2D
- FFT algorithms
- Application to the processing of sounds and images

## Lab in Matlab

Teaching material downloadable from the e-learning platform (and also from Teams)

ONLINE Tutorial 0.0 Matlab Basic and LA Basic

Matlab programs from NCM book

Matlab programs developed during the course

ONLINE Tutorials (Matlab Live Scripts)

# Recap of Linear Algebra

.... with review of basic Matlab

non è ammissibile non conoscere alla perfezione questi concetti di base.

# scalar (or inner) product

Questo è una delle operazioni più importanti, due vettori, di base i vettori sono vettori colonna, i trasposti sono riga.  
E' definito nell'espressione rosa, cioè la sommatoria dei corrispondenti elementi.

$$x^T y (\equiv y^T x)$$

Questa immagine mostra appunto le dimensioni dei vettori in questione

=

il risultato è uno scalare, per questo si chiamerà prodotto scalare.

$$x^T y = \sum_{i=1}^n x_i y_i$$

Questa operazione presenta una proprietà molto importante:

La bilinearità, cioè la somma trasposta moltiplicato per Z (vettore colonna) è uguale alle somme dei singoli prodotti dei trasposti per Z

## bilinearity

$$(x + y)^T z = x^T z + y^T z$$

Questo è un modo alternativo, sempre bilinearità

$$x^T (y + z) = x^T y + x^T z$$

L'altra caratteristica è che se vogliamo fare l'inner / scalar product tra due vettore x e y moltiplicati per due scalari alpha e beta possiamo fare prima l'inner product dei due vettori e poi moltiplicare il risultato per i due scalari alpha e beta. Questa è la LINEARITA'

**x' \* y**



Questo è il comando per farlo in matlab

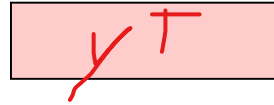
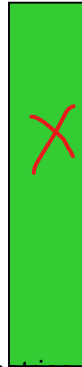
$$\alpha x^T \beta y = \alpha \beta x^T y$$



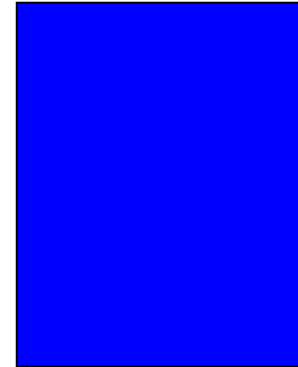
# outer (or external) product

Questo è diverso perché è l'inverso, cioè il primo elemento è un vettore colonna, mentre il secondo è un vettore riga. Il risultato è una matrice.

$$xy^T$$



=



La proprietà che ha questa operazione è detta RANK 1, è una matrice di rango 1.

E' ottenuto nel seguente modo:

ogni colonna è un multiplo di X cioè X rimane fissa e si moltiplica per tutti gli y,  $x_1y_1$   $x_1y_2$   $x_1y_3$

mentre ogni riga è un multiplo di Y trasposto cioè y rimane fissa e x cambia  $x_1y_1$   $x_2y_1$

rank 1

- ❖ each column is a multiple of  $x$
- ❖ each row is a multiple of  $y^T$

$$\mathbf{x} * \mathbf{y}'$$

Questo è il comando in MATLAB.

Queste sono operazioni che riguardano i vettori.

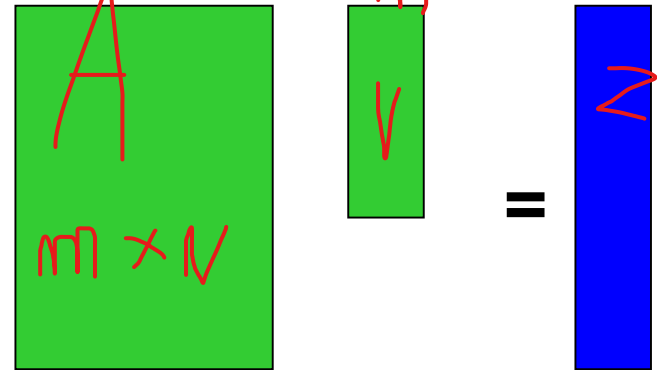
Questa operazione è fondamentale, è il prodotto matrice vettore, il risultato è un vettore colonna.

## matrix – vector product

$$\mathbf{z} = \mathbf{A} * \mathbf{v}$$

Comando in matlab

$$\mathbf{A}\mathbf{v} = \mathbf{z}$$



### various interpretations (2)

$A$  è  $M \times N$   $V$  è  $N$  dimensionale, quindi il risultato sarà un vettore  $z$  di dimensione  $M$

Esistono due interpretazioni: la prima è che la  $i$ -esima componente di  $Z$  è ottenuta con il prodotto scalare tra l' $i$ -esima riga di  $A$  e  $V$

1 – the  $i$ -th component of  $z$  is the **scalar product** of the  $i$ -th row of  $A$  and the column vector  $v$

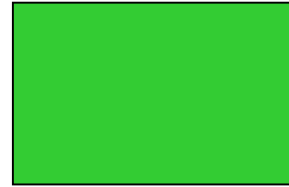
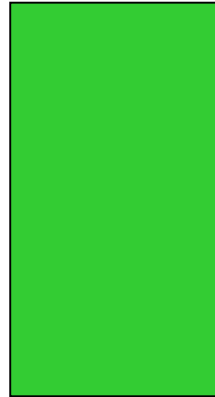
La seconda interpretazione è che il vettore  $Z$  è la combinazione lineare delle colonne di  $A$  (cioè la somma della prima colonna di  $A$  con la prima componente di  $V$ , La seconda colonna di  $A$  con la seconda componente di  $V$  etc.)

2 – the column vector  $z$  is the **linear combination** of the columns of the matrix  $A$ , i.e. the sum of the first column of  $A$  times the first component of  $v$ , and the second column of  $A$  times the second component of  $v$ ,.....

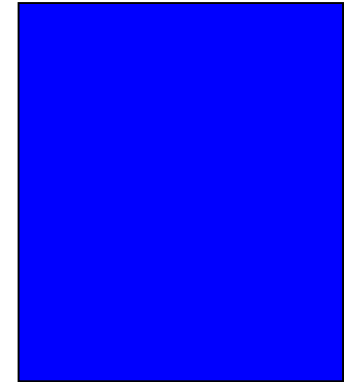
# matrix – matrix product

Prodotto matrice matrice.  $A$  è  $M \times N$ ,  $B$  deve essere per forza  $N \times P$ . Otteniamo  $C$  che è una matrice  $M \times P$ .

$$AB = C$$



=



Questo è il comando in matlab

$$C=A*B$$

various interpretations (3)

Ci sono tre modi di interpretare, basta leggerli.

- 1 – the  $i,j$ -th entry of  $C$  is the **scalar product** of the  $i$ -th row of  $A$  and the  $j$ -th column of  $B$
- 2 – the  $j$ -th column of  $C$  is the **matrix-vector product** of the matrix  $A$  and the  $j$ -th column of  $B$
- 3 – the matrix  $C$  is the sum (over  $i$ ) of the **outer products** of the  $i$ -th column of  $A$  and the  $i$ -th row of  $B$

Queste sono alcune proprietà delle matrici.

## properties

Questa è chiara, il trasposto del prodotto è a fare il prodotto tra i trasposti.

$$(AB)^T = B^T A^T$$

exercise:  
verify with Matlab

Questa è un'altra proprietà è la stessa cosa di scrivere...

$$Av = z \iff v^T A^T = z^T$$

L'inverso del prodotto è uguale al prodotto tra le inverse

$$(AB)^{-1} = B^{-1} A^{-1}$$

inverso di 3 è 1/3 etc.

L'inverso della matrice A è la matrice che moltiplicata per A dà la matrice di identità.

**inv(A)**

Comando di inversione

Anche questa è facile, l'inversa della trasposta è la trasposta dell'inversa.

$$(A^T)^{-1} = (A^{-1})^T$$

Il calcolo dell'inversa è complessa, costa molto se A è grande.

Non entra nel dettaglio di questa formula.

$$(A + xy^T)^{-1} = A^{-1} - \frac{A^{-1}xy^T A^{-1}}{1 + y^T A^{-1}x}$$

Sherman – Morrison Formula

## properties

Data una matrice  $A$   $m \times n$  il RANGE di  $A$   $R(A)$ , è l'insieme dei vettori colonna di  $m$  componenti ottenuti da una combinazione lineare delle colonne di  $A$  con un qualsiasi vettore  $X$  di  $n$  componenti.

$$A \in \mathbb{R}^{m \times n}$$

$$R(A) = \{y \in \mathbb{R}^m : y = Ax \quad x \in \mathbb{R}^n\}$$

Range, column space

null space di  $A$  è l'insieme dei vettori  $x$  di  $n$  dimensioni per cui la combinazione lineare tra  $A$  e  $x$  dà un vettore zero.

$$N(A) = \{x \in \mathbb{R}^n : Ax = 0\}$$

Null Space

$$\text{rank}(A) = \dim[R(A)]$$

il rank di  $A$  è = alla dimensione del Range di  $A$

Rank of a matrix

rank + la dimensione di null space =  $n$ .

$m \geq n$

$$\text{rank}(A) + \dim[N(A)] = n$$

$$m \geq n$$

# Kronecker product (tensor)

$$A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{m_B \times n_B}$$

Il prodotto di Kronecker serve per costruire particolari matrici ed è definito in questo modo:

$$A \otimes B = S$$

$$S = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \\ a_{m1}B & \cdots & \cdots & a_{mn}B \end{pmatrix}$$

il prodotto di Kronecker definisce una matrice a blocchi di dimensione

the Kronecker product defines a block matrix of size

$$(m \times m_B) \times (n \times n_B)$$

where  $A$  determines the structure of the block matrix and  $B$  is the repeated basic block

dove  $A$  determina la struttura della matrice a blocchi e  $B$  è il blocco di base ripetuto

# Kronecker product (tensor)

$$A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{m_B \times n_B}$$

$$A \otimes B = S$$

Questo è un esempio in matlab.

$$S = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \\ a_{m1}B & \cdots & \cdots & a_{mn}B \end{pmatrix}$$

```
kron(eye(3,3), 2*ones(2,2))
```

```
ans =
```

2	2	0	0	0	0
2	2	0	0	0	0
0	0	2	2	0	0
0	0	2	2	0	0
0	0	0	0	2	2
0	0	0	0	2	2

# Kronecker product (tensor)

Si applica anche ai vettori questa operazione.

```
kron([1 0 1],[10 20 30 40 50])
```

```
ans =
```

```
10 20 30 40 50 0 0 0 0 0 10 20 30 40 50
```

```
kron([1 0 1; 0 1 0],[10 20 30 40 50])
```

```
ans =
```

```
10 20 30 40 50 0 0 0 0 0 10 20 30 40 50  
0 0 0 0 0 10 20 30 40 50 0 0 0 0 0
```



# Kronecker product (tensor)

exercise: verify with Matlab that:

$$u^T A v = (v \otimes u)^T A^{(s)}$$

$A^{(s)}$  ( $A$  stacked) is the vector formed by collecting the columns of  $A$

Prendi la matrice e da questa generi un vettore colonna composto da tutte le colonne messe insieme

tipo 1 2 3  
4 5 6

Diventa:

1  
4  
2  
5  
3

$$\mathbf{A\_stacked} = \mathbf{A}(:)$$

Concetto di norma cioè di grandezza di un vettore, che poi estendiamo anche alle matrici. Quindi una norma è il modo di associare al vettore un numero che ne indica la grandezza. Esistono infinite norme ma tutte devono soddisfare le tre condizioni presenti nella slide.

## vector norms

La norma è sempre maggiore di 0, è 0 solo quando il vettore è nullo

$$x \neq 0 \Leftrightarrow \|x\| > 0 \quad ; \quad \|0\| = 0$$

la norma di un multiplo di un vettore è = al multiplo per la norma del vettore

$$\|cx\| = |c| \cdot \|x\|$$

vale la proprietà triangolare, la norma della somma è minore = delle singole norme sommate.

$$\|x + y\| \leq \|x\| + \|y\|$$

esistono infinite norme.

$$\|x\|_{0.5}, \|x\|_1, \|x\|_2, \|x\|_p, \dots, \|x\|_\infty$$

$$\|x\|_1 = \sum_{i=1}^n |x_i|$$

**norm(x, 1)**

$$\|x\|_p = \left( \sum_{i=1}^n x_i^p \right)^{1/p}$$

**norm(x, p)**

$$\|x\|_2 = \left( \sum_{i=1}^n x_i^2 \right)^{1/2}$$

**norm(x)**

$$\|x\|_\infty = \max_i (|x_i|)$$

**norm(x, inf)**

unit spheres

the set of vectors whose norm is less or equal to 1

exercise: visualize in Matlab the unit spheres for these 4 norms

(use the file `Draw_unit_circles_for_norms`)

Si ottengono queste norme in matlab, dopo la virgola si mette la norma che vogliamo ottenere, solo il vettore vuol dire norma 2

qui mostra come si calcolano le diverse norme.

il cerchio di raggio 1 è il luogo dei vettori i quali hanno norma 2 = 1

il quadrato è il luogo dei vettori i quali hanno norma infinito = 1

$$\|x\|_1 = \sum_{i=1}^n |x_i|$$

$$\|x\|_2 = \left( \sum_{i=1}^n x_i^2 \right)^{1/2}$$

$$\|x\|_p = \left( \sum_{i=1}^n x_i^p \right)^{1/p}$$

$$\|x\|_\infty = \max_i (|x_i|)$$

Questa è un'altra proprietà che vale solo quando per qualunque coppia coniugata  $p$  e  $q$  vale questa condizione

$$|x^T y| \leq \|x\|_p \|y\|_q$$

$$\frac{1}{p} + \frac{1}{q} = 1$$

Holder inequality

Questa disuguaglianza viene chiamata Cauchy-Schwarz e dice quindi: il prodotto scalare di due vettori il primo trasposto, è  $\leq$  del prodotto scalare delle norme.

$$|x^T y| \leq \|x\|_2 \|y\|_2$$

Cauchy – Schwartz inequality

exercise: verify with Matlab that

$$\|x\|_{\infty} \leq \|x\|_2 \leq \sqrt{n} \|x\|_{\infty}$$

$$\|x\|_{\infty} \leq \|x\|_1 \leq n \|x\|_{\infty}$$

$$\|x\|_2 \leq \|x\|_1 \leq \sqrt{n} \|x\|_2$$

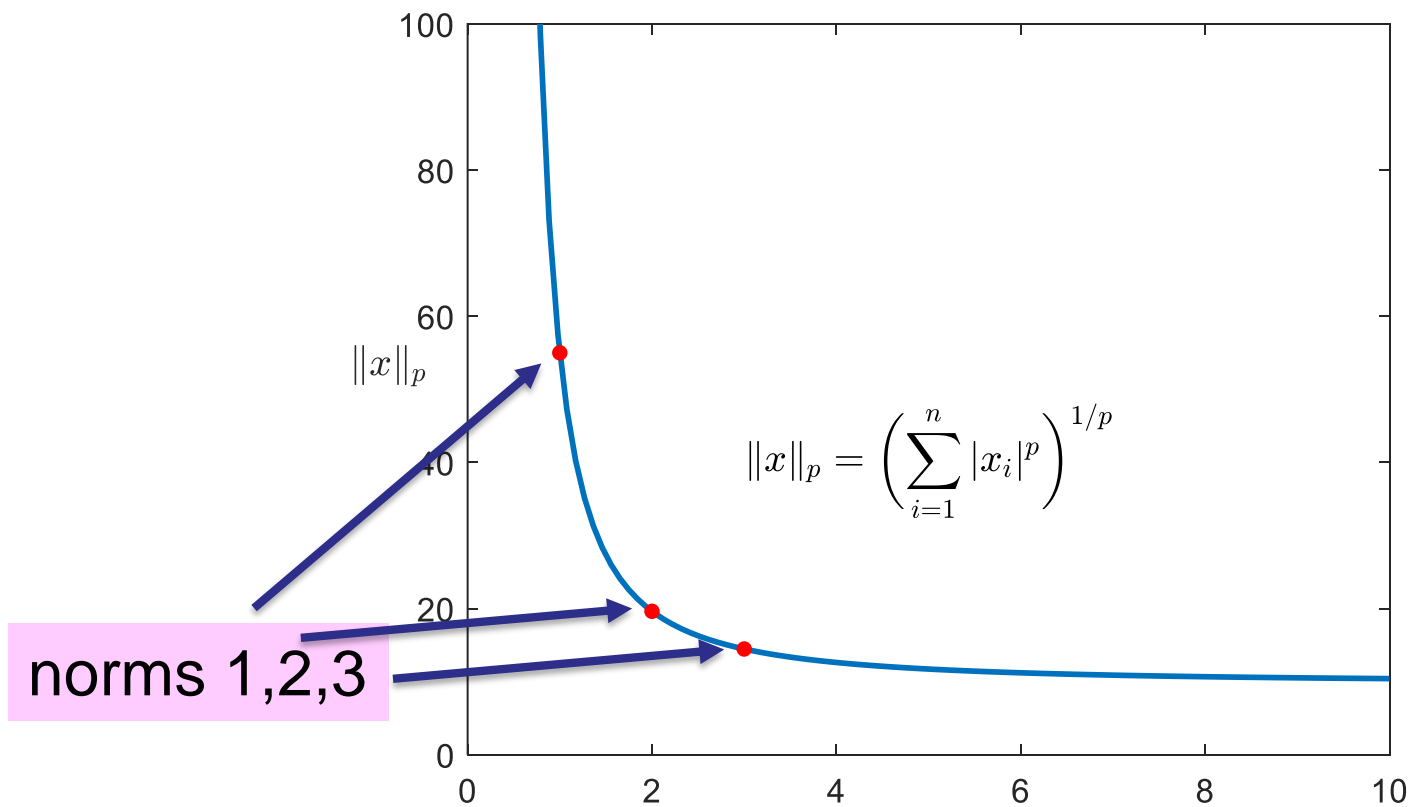
$$c_1 \|x\|_{\alpha} \leq \|x\|_{\beta} \leq c_2 \|x\|_{\alpha}$$

**equivalence** of vector norms

**exercise:** in Matlab plot the value of a norm of a vector as a function of the order of the norm

$$(p, \|x\|_p)$$

$x$  fixed vector,  $p$  in  $[0.5, 10]$



```

% script DrawNorms. Graph of the p-norm of the
% vector x = [1..10] vs p (pvals)
n = 10; m = 100;
x = 1:n;
y = zeros(m,1);
pvals = linspace(0.5,10,m);
for i = 1:m
    y(i) = norm(x,pvals(i));
end
plot(pvals,y,'LineWidth',2)
ylim([0 100])
hold on
plot([1 2 3],[norm(x,1) norm(x,2) norm(x,3)],'.r','MarkerSize',
16)
hold off
options={'Interpreter','latex','FontSize',14};
ylabel('$\|x\|_p$',options{:},'Rotation',0)
s = '$\|x\|_p = \biggl(\sum_{i=1}^n |x_i|^p \biggr)^{1/p}$';
text(options{:},'String',s,'Position',[3 40])

```

# matrix norms

Le norme matriciali sono un po complicate da calcolare ma l'idea è definirle in termini di norma di vettori, si parla di norme indotte. La norma più famosa è la Frobenius norm

Questa è la definizione di norma indotta.

**induced norms**, i.e.  
defined in terms of a  
vector norm

$$\|A\| = \max_{\|x\|=1} \|Ax\|$$

è come se fosse la norma 2

**Frobenius norm**

$$\|A\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2}$$

`norm(A, 1)`

`norm(A, 2)`

`norm(A)`

`norm(A, inf)`

`norm(A, 'fro')`



# matrix norms

Questo è il modo in cui si calcolano le varie norme delle matrici.

exercise: verify with Matlab that

norma 1, massimo tra le somme degli elementi su ogni colonna

$$\|A\|_1 = \max_j \sum_{i=1}^m |a_{ij}|$$

stessa cosa di 1 ma sulle righe

$$\|A\|_\infty = \max_i \sum_{j=1}^n |a_{ij}|$$

$$\|A\|_2 = \max_{\|x\|_2=1} \|Ax\|_2$$

La norma 2 al momento la lascia sotto forma di norma indotta.

$$\|A\|_F = \|A^{(s)}\|_2$$

la norma di frobenius è la norma 2 della slackced della matrice.

exercise: verify with Matlab that

$$\|AB\|_p \leq \|A\|_p \cdot \|B\|_p$$

$$\|Ax\|_p \leq \|A\|_p \cdot \|x\|_p$$

$$\|A\|_2 \leq \|A\|_F \leq \sqrt{n} \|A\|_2$$

$$p = 1, 2, \inf$$

un'altra definizione di prodotto scalare è questa qui: cioè prod delle norme \* cos dell'angolo compreso tra l'angolo di due vettori. E da qui deriva la disuguaglianza di swartz. Cos alpha è un numero in val ass minore di 1 quindi la seconda parte dell'equazione di sotto è sicuramente maggiore uguale del prod scalare.

## inner product and angles

$$x^T y = \|x\|_2 \cdot \|y\|_2 \cdot \cos(\alpha)$$

$\alpha$  is the angle between the two vectors  $x$  and  $y$

più che altro questa equazione ci serve per calcolare il coseno dell'angolo tra i due vettori perché dividiamo il prod scalare per il prod delle norme.

$$\cos(\alpha) = \frac{x^T y}{\|x\|_2 \cdot \|y\|_2}$$

$$x^T y \leq \|x\|_2 \cdot \|y\|_2$$

il prod scalare = a 0 quando l'angolo è 0 (cioè  $\pi/2$ , 90°) cioè quando sono perpendicolari tra loro

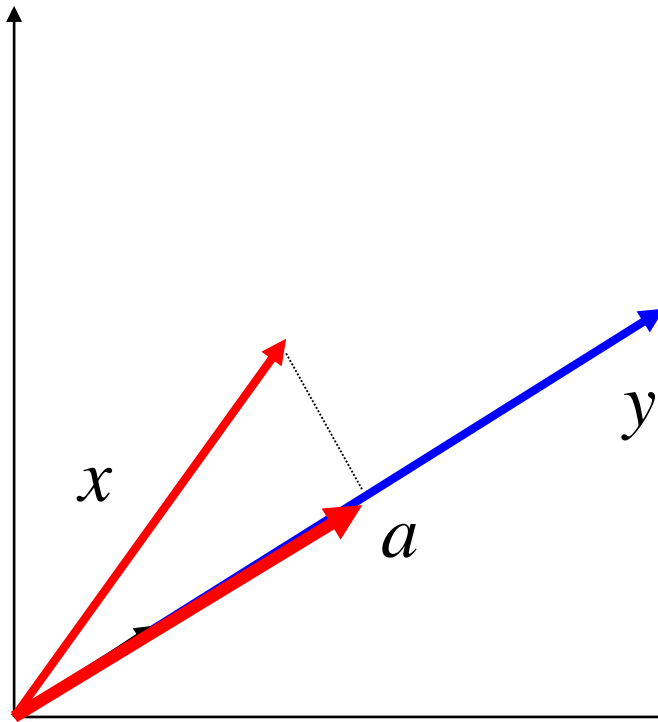
$$x^T y = 0 \iff x \perp y$$

## inner product and projections

$$x^T y = \|x\|_2 \cdot \|y\|_2 \cdot \cos(\alpha)$$

il prodotto scalare ha anche a che fare con le proiezioni ortogonali di un vettore su un altro. Per fare la proiezione calcolo la lunghezza del segmento  $oa$  in quel modo. Il vettore  $A$  è = alla lunghezza di  $oa$  \* il versore di  $y$  (il versore di  $y$  è  $y / \text{norma 2 di } y$ )

orthogonal projection of the vector  $x$  onto  $y$



alla fine  $a$  è la proiezione di  $x$  su  $y$ .

$$\overline{oa} = \|x\|_2 \cdot \cos(\alpha) = \frac{x^T y}{\|y\|_2}$$

vector  $a$  = length  $oa$  times the versor of  $y$

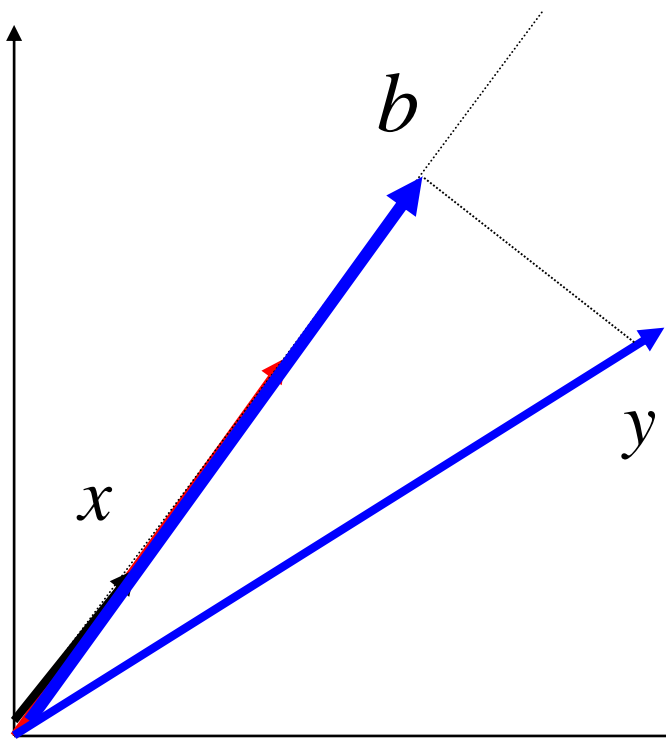
$$a = \overline{oa} \frac{y}{\|y\|_2} = \frac{x^T y}{\|y\|_2^2} y$$

la stessa cosa di prima vale se vogliamo proiettare  $y$  su  $x$

inner product  
and projections

$$x^T y = \|x\|_2 \cdot \|y\|_2 \cdot \cos(\alpha)$$

orthogonal projection of the vector  $y$  onto  $x$



cambia il denominatore ovviamente

$$\overline{ob} = \|y\|_2 \cdot \cos(\alpha) = \frac{x^T y}{\|x\|_2}$$

vector  $b$  = length  $ob$  times the versor of  $x$

$$b = \overline{ob} \frac{x}{\|x\|_2} = \frac{x^T y}{\|x\|_2^2} x$$

# linear systems of equations

$$A \in \mathbb{R}^{n \times n}$$

L'operazione inversa rispetto al prod matrice vettore è la risoluzione di un sistema lineare. Qui conosciamo  $a$  e  $b$  e voglia determinare  $x$  invece con il prod ho  $A$  e  $x$  ottengo  $b$ .

$$Ax = b$$

può essere visto come un cambio di base per rappresentare il vettore  $b$  : dalla rappresentazione nella base standard alla rappresentazione nella base delle colonne di

may be seen as a change of basis to represent the vector  $b$  :

- ✓ from the representation in the standard basis
- ✓ to the representation in the basis of the columns of  $A$

Si risolve con la fattorizzazione  $lu$  con pivoting parziale. E' un operazione cubica in  $n$  dove  $n$  è il numero di righe e colonne. E' il miglior modo per risolverlo perché produce un residuo piccolo.

$$PA = LU$$

**LU factorization**  
(Gaussian elimination)  
with partial pivoting

$$LUx = Pb$$

$$Lp = Pb$$

$$Ux = p$$

$$O\left(\frac{n^3}{3}\right)$$

provides a small  
residual  $b - Ax$

# linear systems of equations in Matlab

Si risolve in matlab così.

$$\mathbf{x} = \mathbf{A} \backslash \mathbf{b}$$

La fattorizzazione LU la si fa così.

$$[\mathbf{L}, \mathbf{U}] = \text{lu}(\mathbf{A})$$

Questo ci fa ottenere la matrice degli scambi data dal pivoting

$$[\mathbf{L}, \mathbf{U}, \mathbf{P}] = \text{lu}(\mathbf{A})$$

ci permette di scegliere il metodo di pivoting e ci mostra passo passo cosa accade.

$$\text{lugui}(\mathbf{A})$$

Questo risolve più sistemi con la stessa matrice, e i vettori di termini noti sono le colonne di B

$$\mathbf{X} = \mathbf{A} \backslash \mathbf{B}$$

solves several systems: same matrix, vectors of known terms given by the columns of  $\mathbf{B}$

Se abbiamo due sistemi lineari dove il secondo è molto simile al primo infatti possiamo scriverlo anche in quel modo con  $\delta$ , allora la domanda principale è se due sistemi sono vicini, possiamo aspettarci che le loro soluzioni siano vicine?

## Conditioning of linear systems

two systems:

Diciamo che  $z$  e  $q$  sono due perturbazioni di vettori  $x$  e  $b$ .

La risposta alla nostra domanda non è affermativa e quanto è grande ce lo dice un teorema che ora esponiamo.

$$Ax = b$$

$$Az = q$$

$$A(x + \Delta x) = (b + \Delta b)$$

relevant question:

if the two systems are “close together”, can we expect that even their two solutions are “close together”?

question: if

$$\Delta b$$

is small

then

$$\Delta x$$

is small ?



Definiamo prima di tutto l'errore assoluto e relativo. Abbiamo  $\hat{x}$  che è un'approssimazione di  $x$ . Si dice che  $\hat{x}$  è un'approssimazione di  $x$  con un certo errore assoluto  $\Delta x$  dove l'errore assoluto è quella formula lì dovrò dire la norma di  $\hat{x} - x$  (norma non precisata.)

# absolute error and relative error

Definiamo poi l'errore relativo che è l'errore assoluto diviso la norma del vettore  $x$

suppose  $\hat{x}$  be an approximation of  $x$

$$\varepsilon_a = \|\hat{x} - x\| = \|\Delta x\|$$

absolute error

$$\varepsilon_r = \frac{\|\hat{x} - x\|}{\|x\|} = \frac{\|\Delta x\|}{\|x\|}$$

relative error

E' importante ricordare una proprietà dell'errore relativo. Supponiamo che l'errore relativo sia dell'ordine di  $10^{-d}$  allora la più grande componente di  $\hat{x}$  ha  $d$  cifre significative corrette.

$$\frac{\|\hat{x} - x\|_{\infty}}{\|x\|_{\infty}} \cong 10^{-d}$$

the maximum entry of  $\hat{x}$  has at most  $d$  correct significant digits

example:  $x = (1.234, 0.05674)^T, \hat{x} = (1.235, 0.05128)^T$

$$\varepsilon_r \cong 0.0043 \cong 10^{-3}$$

Il teorema di cui parlava prima dice che l'errore relativo della soluzione è minore = dell'errore relativo sui dati \* un indice di condizionamento che può essere molto grande. Quindi b può essere molto piccolo ma x dovrà essere molto grande

## conditioning of linear systems

two systems:

$$Ax = b$$

$$Az = q$$

$$A(x + \Delta x) = (b + \Delta b)$$

Theorem

$$\frac{\|\Delta x\|}{\|x\|} \leq \kappa(A) \frac{\|\Delta b\|}{\|b\|}$$

l'indice di condizionamento dipende solo da  $A$   
ed è il prod della norma di  $A$  per la norma della sua inversa.

**condition number**

$$\kappa(A) = \|A\| \|A^{-1}\|$$

Questo indice è sempre  $\geq 1$ ,  $= 1$  in casi particolari x esempio se abbiamo la matrice identità.  
Se la moltiplichiamo per una qualunque costante non cambiamo l'indice.

## conditioning of linear systems

$$\kappa(A) = \|A\| \|A^{-1}\|$$

$$\kappa(A) \geq 1$$

$$\kappa(I) = 1$$

$$\kappa(P) = 1$$

$$\kappa(cA) = \kappa(A)$$

se la matrice è diagonale l'indice è il rapporto tra il più grande e più piccolo degli elementi nella diagonale.

$$\kappa(D) = \frac{\max |d_{ii}|}{\min |d_{ii}|}$$

$D$  diagonal matrix

Fino ad ora però nel problema del condizionamento abbiamo visto che  $A$  è la stessa. Ma cosa accade se perturbiamo anche  $A$  (perturbano = la cambiamo di poco)

In questo caso C è vicino ad A q è vicino a b e z vicino a x. In questo caso il teorema è in quella forma.

## conditioning of linear systems

two systems:

$$Ax = b$$

$$Cz = q$$

$$(A + \Delta A)(x + \Delta x) = (b + \Delta b)$$

L'unica cosa è che si sommano gli errori su B e A. Il regista di tutto è sempre l'indice di condizionamento,

### Theorem

$$\frac{\|\Delta x\|}{\|x\|} \leq \kappa(A) \left( \frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta b\|}{\|b\|} \right)$$

Ricordare che se l'indice è circa = 1 allora il sistema è ben condizionato, se è molto maggiore di 1 è mal condizionato.

## conditioning of linear systems

A lui INTERESSA QUESTA  
REGOLA APPLICATIVA IN  
ROSSO

if

$$\kappa(A) \approx 1$$

the linear system is  
well-conditioned

if

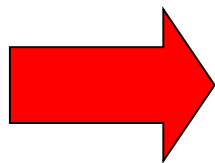
$$\kappa(A) \gg 1$$

the linear system is  
ill-conditioned

La regola applicativa, che è la cosa più importante è se l'indice è dell'ordine  $10^q$  allora vuol dire che la soluzione sarà nota con  $q$  cifre significative in meno rispetto alle cifre significative note di  $A$  e  $b$

**thumb rule:**

$$\kappa(A) = 10^q$$



**the solution can be known  
with  $q$  significative digits  
less than the number of  
significative digits of the  
data  $(A, b)$**

x ES. Se abbiamo un sistema dove i dati hanno 16 cifre decimali e l'indice è alla 10 perdiamo 10 cifre quindi la solu non può avere più di 6 cifre accurate. E' una proprietà del problema e non degli algoritmi risolutori.

Se  $A$  è mal condizionata allora vuol dire che è quasi singolare. L'inverso dell'indice è la minima distanza tra  $A$  è la più vicina matrice singolare. La formula prevede il rapporto. In pratica se l'indice di  $A$  è grande allora la quantità è piccola quindi  $A$  è VICINO ALLA MATRICE SINGOLARE. Questa è l'interpretazione della formula.

## conditioning of linear systems

**$A$  ill-conditioned**



**$A$  near singular**

$$\frac{1}{\kappa(A)} = \min \left( \frac{\|A - B\|}{\|A\|} : B \text{ is singular} \right)$$

Ricorda anche un'altra cosa, se  $x^*$  è la soluzione calcolata con Gauss con pivoting allora il residuo relativo è piccolo.

## conditioning of linear systems

$$Ax = b$$

if

$x^*$

is the solution computed by  
Gaussian elimination with pivoting

the the **relative residue** is small

Il residuo relativo è definito come la norma del residuo diviso il prodotto tra norma di  $A$  e norma di  $x^*$ . Questa quantità è vicina all'epsilon macchina ( $10^{-16}$ )

$$r_{rel} \equiv \frac{\|b - Ax^*\|}{\|A\| \|x^*\|} \leq \rho \varepsilon_{\text{machine}}$$

Quindi gauss è un algoritmo ottimo perché garantisce che il residuo è piccolo, Ma anche l'errore è piccolo?

$\rho$  is a *small number*

Il teorema risponde a questa domanda, ovviamente NO. l'errore non è piccolo per forza.

## conditioning of linear systems

question: if the **residue** is small, then  
is the **errore** small?

Il teorema dice che quando l'indice aumenta il residue ci dà meno informazioni relativamente all'errore.

$$r_{rel} \cdot \frac{1}{K(A)} \leq \varepsilon_r \leq r_{rel} \cdot K(A)$$

questa è una conseguenza del fatto che  $\kappa(A)$  è mal posta.

when the condition number increases, the **residue** gives  
less information about the **error**



## conditioning of linear systems

estimate of the condition number of a matrix in Matlab

**cond (A , ... ) ,**

**condest (A)**      1-norm

**rcond (A)**      1-norm (inverse of cond. numb.)