COMANDI:

help e doc : info su un comando

look for .... : cerca comandi con quel termine

diary : salva la sessione di lavoro in un file, la sessione termina con diary off

.SCALARI

a=2.5 ; : imposto variabile (;) esegue, ma non ti fa visualizzare

(who) whos : elenca variabili

clear all : cancella le variabili

clear a : cancella variabile a

clc : pulisce la pagina

pi : π

i, j : $i = \sqrt{-1}$

.OPERAZIONI elementari

+ - * / ^ (elev a potenza)

.N COMPLESSI

z = a + i*b

real(z): restituisce a

imag(z): restituisce b

conj(z): restituisce il complesso coniugato -> a-ib

.FUNZIONI MATEMATICHE

abs(x)	$\|x\|$
sqrt(x)	$\sqrt{x}$
nthroot(x,n)	$\sqrt[n]{x}$
exp(x)	e^x
log(x)	log(x)
sin(x)	sin(x)
cos(x)	cos(x)
tan(x)	tan(x)
asin(x)	arcsin(x)

⟶ per l'elenco completo: help elfun

nthroot(x,n): restituisce la radice n-esima REALE

nthroot(-8,3) -> -2 n Reale

-8 ^ 1/3 ⟶ 1.0000 + 1.7321i n complesso

1.VETTORI

v=[1 2 3] vettore RIGA (spazio tra n°)

v=[1; 2; 3] vettore COLONNA (; tra n°)

v=[1:8] vettore di n° successivi (da 1 a 8)

v=[val.iniz : PASSO : val.fin]

linspace(v.iniz, v.fin, N) vettore con val iniz, finale e N coefficenti (elementi del vettore)

zeros(n,1) o zeros(1,n) vettore colonna (riga) con lunghezza n di ZERO

ones(n,1) o ones(1,n) vettore colonna (riga) con lunghezza n di UNO

length(v) : lunghezza vettore

size(v) : dimensione variabile, n° righe e colonne

v(3) : accede alla componente 3 di v

v(1:3) : estraggo componenti 1, 2 e 3

x([4,2,1]) : estraggo componenti 4, 2 e 1

x(1:2:5) : estraggo componenti 1, 3 e 5

.OPERAZIONI sui VETTORI

trasposizione (') trasforma vettore riga in colonna e viceversa ⟶ v(colonna) = v(:) (riga)

OPERAZIONI COMPONENTE per COMPONENTE (vettori riga)

somma (sottrazione) : v+w v-w devono essere di dimensioni uguali

Prodotto (divisione) per uno scalare: 2*v v/2

FUNZIONI PREDEFINITE

sum(v): somma tra le com

prod(v): prodotto tra le co

max(v): max tra le compon

min(v): min tra le compon

FUNZIONI MATEMATICHE valutate in VETTORI

>> x=[0, pi/6, pi/4, pi/3, pi];

>> sin(x); ⟶ ottengo sin valutato in ogni componente di x

funzione x sin(x)

>> x.* sin(x)   .* importante

HANIPOLAZIONE di SOTTOBLOCCHI di VETTORI e CONCATENAZIONE

v=[1 2 3 4 5]   w=[100 200]

>> v(end-1:end)=w   : sostituisce ultime 2 componenti di v con w -> v=[1 2 3 100 200]

>> v(3:4)=[ ]   : sostituisce le componenti indicate con il vettore vuoto [ ] -> v=[1 2 5]

>> z=[v w]   : concatena 2 vettori riga   z=[1 2 3 4 5 100 200]

OPERAZIONI su VETTORI

dot(v,w): prodotto scalare

v*w : v riga w colonna
v*w' : v,w riga
v'*w ; v,w colonna

⟶ di pari lunghezza

norm(v,2) : norma EUCLIDEA
norm(v,inf): norma INFINITO

⟶ NORME

2. MATRICI

>>A=[1 2 3; 4 5 6]; $A=\begin{bmatrix}1&2&3\\4&5&6\end{bmatrix}$ $B=\begin{bmatrix}1&1&1\\1&1&1\end{bmatrix}$ (3 colonne, 2 righe)   2x3

>>B=ones(2,3);

C=A+B   : somma di matrici di uguali dimensioni

D=A-B   : differenza di matrici di uguali dimensioni

>> s=B(1,2)+D(2,3) : estrae 2 elem dalle matrici e li somma

A' (apostrofo) : trasporre A : assegnata A calcola A^T definita da $(A^T)_{i,j} = A_{j,i}$

A * B   ; prodotto RIGHE per COLONNE

possiamo eseguire operazioni elemento x elemento per matrici di uguali dimensioni   A.*B   A./B   A.^B

MATRICI PARTICOLARI

I=eye(n) : matrice IDENTITÀ

H=hilb(n): matrice di HILBERT

FUNZIONI PREDEFINITE

sum(M): somma di og…

prod(M): prodotto di o…

max(M): max di ogni …

min(M): min di ogni c…

sort(M); riordina in col…

HANIPOLAZIONE di SOTTOBLOCCHI di MATRICI e CONCATENAZIONE

A=eye(4) B=hilb(2)

A(3:4,3:4)=B : sostituisci alle ultime 2 colonne e righe di A la matrice B

r=A(4,:)   : estrai 4^a riga di A

A(:,3)=[ ]   : elimina una colonna

D=[A B]   : concatenare ' ' ' in riga
E=[D;C]   : concatenare ; in colonna

⟶ Attenzione alle DIMENSIONI

det(A)   : determinante di A

rank(A): rango di A

C=inv(B): C=inversa di B

norm(A,1)   : $\|A\|_1$
norm(A,inf); $\|A\|_\infty$

⟶ NORME

GENERARE MATRICI TRIANGOLARI

L=tril(C)   : matrice triangolare inferi…

L=tril(C,-1) : matrice triangolare inferi…

U=triu(C)   : matrice triangolare superi…

U=triu(C,+1): matrice triangolare superio…

PROGRAMMARE CON MATLAB : script-file

file con estensione .m : es: myfile.m

le variabili rimangono in memoria

NOME : no spazi, no nomi di variabili o funzioni predefinite

OPERATORI RELAZIONALI

== ~= < > <= >=

1 : vero 0 : falso

- if (condizione == true) : eseguire più operazioni solo se è verificata una data condizione
  istruzioni
  end

- for contatore = inizio : passo : fine : ripete per un n° fissato di volte alcune istruzioni
  istruzioni
  end

- mat_trisup.m :nuovo script

n=input('inserisci dimensione matrice');

Per generare un nuovo *script-file* selezioniamo dal men

file ⟶ new ⟶ Mfile

mat_trisup.m

```
n=input('inserisci dimensione matrice ');
if(n<0)
  disp('dimensione della matrice inaccettab
  return;
end
A=eye(n)*n;
for i=1:n
    A(i,i+1:n)=i./[i+2:n+1];
end
A
```

Per eseguire il codice è sufficiente digitare mat_trisup
prompt nella finestra di comando. Scegliendo come di
otteniamo:

```
>> mat_trisup
inserisci dimensione matrice 4
A =
   4.00000   0.33333   0.25000   0.20000
   0.00000   4.00000   0.50000   0.40000
   0.00000   0.00000   4.00000   0.60000
   0.00000   0.00000   0.00000   4.00000
```

3. NUMERI FLOATING POINT o NUMERI MACCHINA

ogni n° REALE x in MATLAB appartiene approssimativamente agli intervalli:

$-1.80 \cdot 10^{308} \leq x \leq -2.23 \cdot 10^{-308}$ $\qquad$ $2.23 \cdot 10^{-308} \leq x \leq 1.80 \times 10^{308}$

realmax : più grande n. macchina

real min : più piccolo n. macchina

format short : 4 cifre decimali

format long : 15 cifre decimali

format short e : notazione esponenziale con 5 cifre significative

format long e : notazione esponenziale con 16 cifre significative

→ 5,0000 e -01 e : elevato a

eps ; ε_M (più piccolo n° t.c. float (1+eps)>1)

4. SISTEMI LINEARI

$Ax=b$

x = A\b : soluzione del sistema

5. METODI DIRETTI

ELIMINAZIONE GAUSSIANA

SOSTITUZIONE in AVANTI

SOSTITUZIONE all' INDIETRO

FATTORIZZAZIONE LU

$A=LU$

L : triang inf. (coeff diagonali = a 1)

U : triang sup.

individuo L, U e scrivo il sistema lineare equivalente

$LUx=b$, pongo $y=Ux$ ⟶ $Ly=b$ $\quad Ux=y$

quando A è fattorizzabile in A=LU si ottiene $Ax=b$

quando A non è fattorizzabile in A=LU , si calcola U: triang sup , L: triang inf (coeff diagonali = a 1) e P: matrice di permutazione

$PA=LU$ $\qquad Ax=b \rightarrow PAx=Pb$

$Ly=Pb$

$Ux=y$

6. METODI ITERATIVI

CRITERIO di ARRESTO

fisso 2 valori: nitmax e toll, si deve verificare uno dei due casi:

- il n° delle iterazioni raggiunge nitmax
- avviene che $\max\left(\frac{\|x^{(k)}-x^{(k-1)}\|_\infty}{\|x^{(k)}\|_\infty}, \frac{\|Ax^{(k)}-b\|_\infty}{\|b\|_\infty}\right) < toll$

METODO di JACOBI

$x^{k+1} = D^{-1}(E+F)x^k + D^{-1}b$

$B_J = D^{-1}(E+F)$ — file: jacobi.m

`[x,nit]=jacobi(A,b,x0,toll,nitmax)`

METODO di GAUSS-SEIDEL

$x^{k+1} = (D-E)^{-1}Fx^k + (D-E)^{-1}b$

$B_{GS} = (D-E)^{-1}F$ — file gs.m

`[x,nit]=gs(A,b,x0,toll,nitmax)`

- A : matrice
- b : termine noto
- x0 : vettore iniziale
- toll : tolleranza del test d'arresto
- nitmax : n° max di iterazioni consentite
- x : vettore soluzione
- nit : n° di iterazioni effettuate

definizione: matrice quadrata di ordine N è **a DIAG DOMINANTE per righe (colonne)** se gli elementi diagonali sono MAGGIORI in i restanti elementi della stessa riga (colonna) in valore assoluto:

$$|a_{ii}| > \sum_{j=1, j\neq i}^{n} |a_{ij}| \qquad \left(|a_{ii}| > \sum_{j=1, j\neq i}^{n} |a_{ji}|\right)$$

se la disuguaglianza vale con $\geq$ la matrice è **a DIAG in senso DEBOLE**

condizioni suff per la convergenza

- A diagonalmente dominante in senso FORTE $\Longrightarrow$ i due metodi J e GS convergono
- A simmetrica definita positiva $\Longrightarrow$ GS converge
- se esiste una norma di matrice naturale t.c. $\|B\| < 1$ allora il metodo iterativo (0) con matrice di iterazione B è convergent

condizione necessaria e suff per la convergenza

metodo iterativo (0) con matrice di iterazione B, converge $\Longleftrightarrow \rho(B) < 1$

matrici tridiagonali

A tridiagonale J converge $\Longleftrightarrow$ GS converge

GS è due volte più veloce, richiede la metà delle iterazioni del metodo di Jacobi

7. GRAFICI in 2D

`plot(x,y)`: comando per disegnare un grafico x, y sono 2 vettori di uguale dimensione $\longrightarrow$ con `plot(x,y,'sp`

data la funzione devo individuare un n° significativo di punti $x_i \in [a,b]$ in cui valutare la funzione — scelgo COLORE, SIM

b	blue	.
g	green	o
r	red	x
c	cyan	+
m	magenta	*
y	yellow	s
k	black	d
w	white	v
		^
		<
		>
		p
		h

doc LineSpec: tutte le opzio

★ **se voglio due grafici sulla stessa finestra:** negli esami

```
>>x=linspace(v.inis,v.fin,N);
>>fx=...;
>>gx=...;
```

a questo punto ci sono 2 opzioni:

1.
```
>>plot(x,fx)
>>hold on
>>plot(x,gx)
```

2.
```
>>plot(x,fx,x,gx)
```

creazione di più finestre grafiche

```
>>x=linspace(v.inis,v.fin,N);
>>fx=...;
>>gx=...;
>>figure(1)
>>plot(x,fx)
>>figure(2)
>>plot(x,gx)
```

$\longrightarrow$ finestra 1: fx, finestra 2: gx funzioni anonime

>>f=@(x) (sin(x)+x).^2 : per funzioni complesse

ATTENZIONE: usa il punto. se si vuole che la funzione operi sui VETTORI

ad una funzione così definita NON sono associati valori numerici, ciò è verificabile con whos f

per calcolare i valori di f basta definire il vettore x e y=f(x)

plot(x,y): grafico

>> f=@(a,x) 2*x+a : funzione che dipende da più VARIABILI/PARAMETRI

B.POLINOMI e VETTORI

$p(x) = -3x^4 + 2x^3 + x - 5$

>>p=@(x) -3*x.^4 + 2*x.^3 + x - 5 : p come funzione anonima

>> p(2): p valutato in 2

un polinomio di grado n ha n+1 coefficienti, ad un polinomio di grado n si associa un vettore di lunghezza n+1

>> p=[-3 2 0 1 -5]   ricorda gli ZERO quando la potenza non appare

OPERAZIONI tra e su POLINOMI

.valutazione di un polinomio in uno o più punti

x= un qualsiasi valore in memoria   x=2

p= polinomio es: $p(x) = x^2 + x - 1$ p=[1 1 -1]

w=polyval(p,2) : per calcolare w=p(2)

w=polyval(p,x): per calcolare w=(p(0),p(4),p(5)) vettore, se x=[0 4 5]

.prodotto o divisione per uno SCALARE

a: variabile in $\mathbb{R}$

$z(x) = a\,p(x)$ ⟶ z=a*p

$z(x) = \frac{p(x)}{a}$ $a \neq 0$ ⟶ z=p/a

p=[4 2 1]

3*p

.somma o differenza tra polinomi

$z(x) = p(x) \pm q(x)$: può essere fatto SOLO se i due polinomi hanno lo stesso grado (vettori lunghi uguali), altrimenti "a

z=polysum(p,q)

.prodotto tra polinomi

$s(x) = p(x)\,q(x)$

s=conv(p,q)   NO p*q!

.divisione tra polinomi

$v(x) = q(x)\,u(x) + r(x)$ q(x): quoziente r(x)= resto

[q,r]=deconv(v,u)

conv(q,u)+r

.derivata di un polinomio

$s(x) = p'(x)$

s=polyder(p)

.primitiva di un polinomio

$s(x) = \int p(x)\,dx$

s=polyint(p): fornisce la primitiva con c=0

9. INTERPOLAZIONE POLINOMIALE

p=polyfit(x,y,n)

x: ascisse dei punti assegnati

y: ordinate dei punti assegnati

n: grado del polinomio voluto

p: vettore dei coeff del polinomio calcolato

istruzioni per l'uso:

. x e y due vettori

n=length(x)-1

p=polyfit(x,y,n)

★ interpola valori assunti da f in n+1 nodi equispaziati in [a,b] n: grado polinomio interpolatore

x=linspace(a,b,n+1)

p=polyfit(x,f(x),n)

NODI non equispaziati: NODI di CHEBYSHEV

$n \in \mathbb{N}$ n nodi $\in [-1,1]$

$c_k = \cos\frac{\pi(2k-1)}{2n}$ $k=1,2,\dots,n$

se abbiamo un qualsiasi intervallo [a,b]

$f:[-1,1] \longrightarrow [a,b]$ t.c. $f(t) = \frac{b-a}{2}t + \frac{a+b}{2}$

e quindi i nodi di chebyshev in [a,b] sono definiti da:

$x_k = \frac{b-a}{2}c_k + \frac{a+b}{2}$ $\forall k=1,\dots,n$

SPLINE LINEARE INTERPOLANTE

$$s(x) = \begin{cases} c_{1,1}(x-x_1)+c_{1,2} & \text{se } x \in [x_1,x_2] \\ c_{2,1}(x-x_2)+c_{2,2} & \text{se } x \in [x_2,x_3] \\ \dots & \dots \\ c_{m,1}(x-x_m)+c_{m,2} & \text{se } x \in [x_m,x_{m+1}] \end{cases}$$

★ s(x):griddedInterpolant(x,y,'linear')

calcola spline s(x) e il valore assunto in z=0, disegna grafico, evidenziando con un cerchietto i punti interpolati e con un asterisc

```
s1=griddedInterpolant(x,y,'linear')
z=0;
s1z=s1(z)
plot(x,y,'o',x,y,z,s1z,'*g');
grid on
```

teorema:

m+1 nodi distinti $a=x_1<x_2<\dots x_{m+1}=b$ e $f \in C^2([a,b])$ allora

$$\max_{a\le x\le b} |f(x)-s(x)| \le CH^2$$

dove $H = \max_{1\le i\le m} |x_{i+1}-x_i|$, $C = \frac{1}{8}\max_{a\le t\le b}|f''(t)|$

per stimare l'errore di approssimazione $\max_{a\le x\le b}|f(x)-s(x)|$ calcoliamo la NORMA DISCRETA che calcola il massimo del valore ass

in un numero di punti selezionato es: 1000 o 10000

10. APPROSSIMAZIONE di FUNZIONI o DATI

tabella di dati

plot(x,y,'*'): otteniamo il grafico

METODI di LINEARIZZAZIONE

N+1 coppie (x_i, y_i), per $i=0,\dots,N$ di dati voglio determinare

a) $y(x) = Ce^{Ax}$

b) $y(x) = Cx^A$

$C, A \in \mathbb{R}$

caso a) $y(x) = Ce^{Ax}$

$\ln y = \ln(Ce^{Ax})$

$\ln y = \ln C + \ln e^{Ax}$

$\ln y = \ln C + Ax$

$Y = \ln y$, $a_0 = \ln C \longrightarrow Y = Ax + a_0$

i coeff della retta di regressione lineare $r(x) = a_1 x + a_0$

che approssima nel senso dei minimi quadrati i dati $(x_i, \ln y_i)$

individuano i coeff $A = a_1$ e $C = e^{a_0}$ cercati

caso b) $y(x) = Cx^A$

$\ln y = \ln(Cx^A)$

$\ln y = \ln C + \ln x^A$

$\ln y = \ln C + A \ln x$

$Y = \ln y$, $X = \ln x$ e $a_0 = \ln C \longrightarrow Y = AX + a_0$

i coeff della retta di regressione lineare $r(X) = a_1 X + a_0$

che approssima nel senso dei minimi quadrati i dati $(\ln x_i, \ln y_i)$

individuano i coeff $A = a_1$ e $C = e^{a_0}$ cercati

INTEGRAZIONE NUMERICA

data f a valori reali approssima $\int_a^b f(x)\,dx$

q=integral(f,a,b)

- f : funzione integranda
- a,b: estremi di integrazione
- q: approssimazione dell'integrale

I: valore esatto dell'integrale

Q: valore approssimato

$|Q-I| < 1e-10$ e $|Q-I|/I < 1e-6$ (1e-10 e 1e-6 sono valori di default)

11. FORMULE di QUADRATURA SEMPLICI

148