

MATLAB - Introduzione

Enrico Nobile

2^a Parte

1

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

M-files

- MATLAB può eseguire una serie di comandi memorizzati su file;
- Tali files sono chiamati M-files, poiché devono avere l'estensione ".m" (es. tridia.m);
- L'utilizzo di MATLAB avviene perlopiù attraverso la creazione e modifica di M-files;
- Esistono due tipi di M-files: *script files* e *function files*.

2

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Script files

- Un file *script* contiene una sequenza di comandi MATLAB;
- Se, ad esempio, il file script si chiama `jacobi.m`, il comando:

```
»jacobi
```

avrà come risultato l'esecuzione dei comandi contenuti nel file;
- Le variabili in un file script sono *globali* - sono cioè comuni a tutte le applicazioni attive nella sessione MATLAB - e pertanto il loro valore può venire modificato.

3

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Script files (cont.)

- I files script possono venire efficacemente usati per introdurre dati in matrici di ampie dimensioni, in modo da poter effettuare le correzioni utilizzando un Editor;
- Se, ad esempio, si introducono le seguenti linee in un file di nome "`dati_A.m`":

```
A=[ 1 2 3 4  
5 6 7 8]
```

l'istruzione MATLAB: `»dati_A` darà luogo all'assegnazione scritta nel file "`dati_A.m`";
- Un M-file può contenere chiamate ad altri M-file, incluso se stesso (ricorsivo).

4

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Function files

- I files *function* provvedono all'estensibilità di MATLAB;
- Si possono creare funzioni specifiche per il/i problema/i in esame, e che godono dello stesso status delle altre funzioni MATLAB;
- Per *default* le variabili in un file function sono *locali*, a meno di dichiararle *globali* con l'istruzione
 »global
- E' buona norma usare nomi lunghi (autoesplicativi), e con lettere MAIUSCOLE, per le variabili dichiarate globali.

5

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Function files (cont.)

Si consideri, ad esempio, il seguente file function:

```
function a=randint(m,n)
%RANDINT Genera una matrice random intera.
% randint(m,n) fornisce, in output, una
% matrice mxn, con gli elementi interi
% compresi fra 0 e 9.
a=floor(10*rand(m,n));
```

- Esempio d'uso:
 »mat=randint(4,4)
- Potremmo scrivere tale funzione in forma più generale;

6

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Function files (cont.)

```
function a=randint2(m,n,a,b)
%RANDINT2 Genera una matrice random intera.
% randint2(m,n) fornisce, in output, una
% matrice mxn, con gli elementi interi
% compresi fra 0 e 9;
% randint2(m,n,a,b) fornisce, in output,
% una matrice mxn, con gli elementi interi
% compresi fra a e b.
if nargin<4, a=0; b=9; end
a=floor((b-a+1)*rand(m,n))+a;
```

Tale funzione va memorizzata nel file `randint2.m`.

7

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Function files (cont.)

- La prima linea dichiara il nome della funzione, gli argomenti di input e gli argomenti di output:

```
function a=randint2(m,n,a,b)
```

Senza tale linea, il file verrebbe considerato di tipo script;
- Si può verificare che le variabili all'interno di `randint2` sono locali;
- Si noti che l'uso della variabile `nargin` ("number of input arguments") consente di assegnare valori di *default* qualora alcuni argomenti in input siano mancanti (es. `a` e `b` in `randint2`).

8

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Function files (cont.)

Una funzione può anche avere più argomenti di output:

```
function [media,devst]=stat(x)
% STAT Media e deviazione standard
% Dato un vettore x, stat(x) fornisce
% la media e la deviazione standard di x.
% Per una matrice x, stat(x) fornisce due
% vettori riga, con media e deviazione
% standard di ciascuna colonna.
[m,n]=size(x)
if m == 1
    m = n;    % caso di un vettore riga
end
media = sum(x)/m; devst = sqrt(sum(x.^2)/m - media.^2);
```

9

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Function files (cont.)

Memorizzata la funzione stat nel file `stat.m`, il comando:

```
» [xm,xd]=stat(x);
```

assegnerà il valore medio e la deviazione standard del vettore `x` alle variabili `xm` e `xd`.

Il comando:

```
» xm=stat(x)
```

assegnerà la media di `x` a `xm`.

10

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Function files (cont.)

- La funzione `stat` illustra alcune delle caratteristiche di MATLAB che consentono di ottenere codice efficiente (e compatto):
 - `x.^2` è la matrice dei quadrati degli elementi di `x`;
 - `sum` è una funzione vettoriale;
 - `sqrt` è una funzione scalare;
 - `sum(x)/m` è un'operazione matrice-scalare.

11

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Osservazioni:

- Il simbolo `%`, in qualunque posizione di una linea, indica che il resto della linea è un commento;
- Le prime linee di commento verranno inoltre mostrate qualora si digiti il comando:
`>>help stat`
- Tali linee di commento (documentazione) dovrebbero *sempre* venire incluse in un function file;

12

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

La seguente funzione (massimo comune divisore di due interi) illustra l'uso di un messaggio di errore:

```
function a = mcd(a,b)
% MCD Massimo Comune Divisore
% mcd(a,b) è il massimo comune divisore
% di due interi a e b, con almeno uno non zero
a = round(abs(a)); b = round(abs(b));
if a == 0 & b == 0
    error('MCD non definito se a e b sono ambedue zero')
else
    while b ~= 0
        r = rem(a,b);
        a = b; b = r;
    end
end
end
```

13

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

- Altre caratteristiche avanzate sono illustrate dalla funzione `bisect`, non riportata per ragioni di spazio: si tratta di una funzione (illustrata a scopo *didattico*, per usi *professionali* usare la funzione MATLAB `fzero`) che calcola lo zero di una funzione con il metodo delle secanti;
- In particolare, si noti:
 - L'uso, oltreché di `nargin`, della variabile `nargout`;
 - Una relazione è un numero: 1 se *vera*, 0 se *falsa*;
 - Relazioni con `while` o `if`: non-zero = vero, 0 = falso;
 - La funzione MATLAB `feval` consente di utilizzare il nome ("stringa di caratteri") di una funzione come variabile di ingresso.

14

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Stringhe di caratteri, messaggi di errore e user-input

- Le stringhe di caratteri vanno racchiuse fra virgolette (*single quotes*); ad esempio

```
s = ('Stringa di testo')
```

asigna la stringa di testo alla variabile *s*;
- Le stringhe di testo possono venire visualizzate con la funzione `disp`:

```
disp('Calcolo eseguito correttamente')
```

15

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Stringhe di caratteri, messaggi di errore e user-input (cont.)

- Per i messaggi di errore è preferibile usare la funzione `error`:

```
error('Spiacente: la matrice dev'essere  
simmetrica...')
```

poiché, inserita nel M-file, comporta l'uscita dal file;
- In un M-file, l'introduzione interattiva di dati può ottenersi con la funzione `input`; ad esempio:

```
iter = input('Numero massimo di iterazioni ?')
```
- Più stringhe di caratteri possono venire raggruppate in una "cell array".

16

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Miscellanea



- Editing e managing files;
- Debugging;
- Comandi `dir`, `cd` e `what`;
- Comandi `delete` e `type`;
- Comando `more`;
- Comandi: `flops`; `tic`, `toc`, `clock`, `cputime`;
- Comando `diary`.

17

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Cenni di grafica

- MATLAB può produrre sia grafici 2D che numerosi grafici 3D;
- La nuova versione, 5.2, è notevolmente superiore alla 4.x dal punto di vista della visualizzazione 3D (illuminazione, z-buffer, diverse modalità di shading, *et.*);
- Il comando `plot` produce grafici `x-y`; se `x` e `y` sono due vettori di pari lunghezza, il comando `plot(x,y)` apre una finestra grafica e disegna il grafico `y(x)`;
- Ad esempio, i comandi:

```
x= -4:.01:4; y=sin(x); plot(x,y)
```

producono la rappresentazione della funzione seno nell'intervallo (-4, 4).

18

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Cenni di grafica (cont.)

E' inoltre possibile rappresentare curve definite in modo parametrico, ad esempio:

```
t=0:.001:2*pi;x=cos(3*t);y=sin(2*t);plot(x,y)
```

I grafici, inoltre, possono venire completati con i seguenti comandi, che hanno una stringa quale input:

grid	griglia
title	titolo del grafico
xlabel	titolo asse x
ylabel	titolo asse y
text	inserisci il testo nella posizione specificata
gtext	posiziona il testo in modo interattivo (mouse)

19

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Cenni di grafica (cont.)

- Le scale degli assi sono automatiche, se non diversamente specificato con il comando `axis`;
- Esistono vari modi per rappresentare più funzioni sullo stesso grafico; ad esempio:

```
x=0:.01:2*pi; Y=[sin(x)',sin(2*x)',sin(4*x)'];  
plot(x,Y)
```
- Il comando `hold` (switch) congela la finestra grafica, in modo che tutti i successivi grafici vadano a sommarsi;
- Il comando `subplot` può venire usato per suddividere la finestra grafica, in modo da rappresentare simultaneamente sino a 4 grafici.

20

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Cenni di grafica (cont.)

- Per i grafici 3D sono disponibili i seguenti comandi (NB: non esagerare...):

```
plot3; mesh; meshz; surf; surfc; surf1 ...
```

Esempio:

```
» xx=-2:.1:2;  
» yy=xx;  
» [x,y]=meshgrid(xx,yy);  
» z=exp(-x.^2-y.^2);  
» mesh(z)
```

21

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Cenni di grafica (cont.)

MATLAB prevede inoltre comandi per rappresentazioni specifiche:

- Grafici a barre (2D e 3D);
- Grafici ad area;
- Istogrammi;
- Grafici a torta;
- Campi vettoriali (2D e 3D);
- Curve isolivello - contours (2D e 3D);
- Disegno interattivo;
- Animazioni;
- Immagini (numerosi formati);
- Sistema grafico orientato agli oggetti (Handle Graphics).

22

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Cenni di grafica (cont.)

Modellazione 3D:

– patches (uno o più poligoni);

Esempio di lettura e visualizzazione di un file JPEG:

```
» imfinfo r10.jpg
» X = imread('r10.jpg');
» image(X)
» axis image
```

23

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Cenni di grafica (cont.)

Stampa dei grafici/figure (Hardcopy):

- Il comando “print” può venire eseguito sia dal Menu File, sia da comando, ed in quest’ultimo caso è più flessibile;
- Possibilità di invio diretto alla stampante oppure su file, in vari formati (es. PostScript).

24

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Miglioramento Prestazioni (cenni)

Per applicazioni critiche può risultare importante ridurre i tempi di calcolo. Vediamo un esempio:

```
t1=cputime;x=0;
for k=1:10001;
    y(k)=exp(-x)*pi;x=x+0.01;
end
cpu1=cputime-t1
t2=cputime;y=zeros(10001,1);x=0;
for k=1:10001;
    y(k)=exp(-x)*pi;x=x+0.01;
end
cpu2=cputime-t2
t3=cputime;
x=0:0.01:100;y=exp(-x)*pi;
cpu3=cputime-t3
```

25

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Miglioramento Prestazioni (cont.)

- Si può notare il netto guadagno in prestazioni ottenuto “vettorizzando” il calcolo.
- Per un’applicazione particolarmente complessa, è opportuno, prima di qualsiasi intervento migliorativo, eseguire il “profiling” (comando profile).

26

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Utilizzo e gestione della memoria (cenni)

- MATLAB ha la necessità di disporre di un'area contigua di memoria per ogni matrice;
- In caso di "frammentazione" della memoria disponibile, utilizzare il comando "pack";
- La memoria non più utilizzata da MATLAB (liberata ad esempio dal comando "clear") non viene restituita al sistema operativo.

27

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse

- Una matrice descrive, spesso, le relazioni lineari fra le *componenti di un modello*;
- All'aumentare del numero n di componenti, la dimensione della matrice, $n \times n$, aumenta;
- La soluzione di un sistema di n equazioni, richiede n^2 locazioni di memoria, ed un numero di operazioni proporzionale a n^3 ;
- Considerando la potenza di calcolo di comuni PCs e workstations, il tempo di calcolo può divenire inaccettabile se, trattando la matrice come *piena*, n diventa molto grande.

28

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

- Spesso, nei modelli di dimensioni maggiori, ogni componente è in relazione solo con alcuni degli altri componenti, per cui, nella matrice relativa, appaiono molti zeri;
- Tali modelli, e le matrici risultanti, sono detti *sparsi*;
- Definizione di matrice sparsa:

Una matrice si definisce sparsa se il numero di elementi nulli è tale da rendere conveniente il tenerne conto

29

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

- La sparsità riguarda l'efficienza:
 - Evitando operazioni su elementi nulli - tempo di calcolo;
 - Evitando di memorizzare molti elementi nulli - memoria richiesta;
- Un importante parametro descrittivo di una matrice sparsa S , di dimensioni $m \times n$, è $\text{nnz}(S)$, il *numero di elementi non nulli in S* ;
- In generale, il tempo di calcolo per operazioni con matrici sparse è proporzionale a $\text{nnz}(S)$;
- L'utilizzo di matrici sparse rende praticabile la manipolazione di matrici di dimensioni rilevanti;

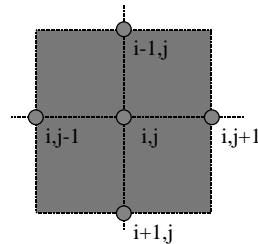
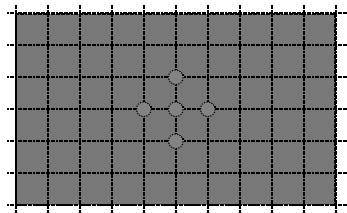
30

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

- Ad esempio, consideriamo la matrice rappresentativa della discretizzazione dell'equazione di *Laplace* (o *Poisson*) su una griglia 64×64 (v. Termocinetica - conduzione):



31

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

- Tale problema consiste di $64 \times 64 = 4096$ incognite;
- Senza tenere conto della sparsità, la matrice risultante sarebbe di dimensione $4096 \times 4096 = 16777216 = 1.6777216 \times 10^7$, e richiederebbe (4 byte per reale - singola precisione) una memoria (reale o virtuale) > 64 Mbyte;
- Gli elementi non nulli sono $20224 (64 \times 64 \times 5 - (4 \times 64))$, che corrisponde a circa 0.12%;
- Sfruttando la sparsità della matrice, la memoria richiesta si riduce a ca. 0.25 Mbyte, ed il tempo di calcolo si riduce di un fattore > 100 .

32

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

- MATLAB ha due modalità di memorizzazione delle matrici: *full* e *sparse*;
- Le matrici sparse sono memorizzate attraverso un vettore di numeri reali (o due vettori per numeri complessi), che contiene gli elementi non nulli, di dimensione nnz ;
- L'informazione relativa agli indici degli elementi non nulli, è contenuta in un vettore di interi, organizzato per colonne, di dimensione $nnz + n$ (n è il numero di colonne della matrice);
- Le matrici sparse **NON** sono create automaticamente;
- Le operazioni con matrici sparse producono matrici sparse;
- Le operazioni miste producono, in generale, matrici sparse, a meno che l'operazione dia luogo a matrici full (es. addizione).

33

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

- Istruzioni sparse e full;
- Forma generale dell'istruzione sparse:
» `S = sparse(i,j,s,m,n,nzmax);`
- Per maggiori informazioni: `help sparse`;
- Vedere inoltre la directory *spafun*, dove sono presenti numerose funzioni per operare con matrici sparse;
- Vediamone alcune di nostro interesse (Termocinetica):
`diag;` `spdiags;` `find;`
`numgrid;` `nested;` `spy;`

34

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

Visualizzazione di matrici sparse:

whos	nnz	nonzeros
nzmax	spy	find

La funzione `find` fornisce gli indici ed i valori degli elementi non nulli (matrici sparse o full):

```
[i,j,s] = find(S)
```

<code>i</code>	vettore indici di riga degli elementi non nulli
<code>j</code>	vettore indici di colonna degli elementi non nulli
<code>s</code>	vettore degli elementi non nulli

35

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

Operazioni con matrici sparse:

- Gran parte delle funzioni standard di MATLAB possono essere applicate a matrici sparse;
- Operazioni specifiche con matrici sparse:

Permutazione:	<code>colperm</code>	<code>colmmd</code>
	<code>symrcm</code>	<code>symmmd</code>

Fattorizzazione:

- LU
- Cholesky
- QR
- Fattorizzazioni incomplete

36

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

Esempi:

```
[L,U,P] = lu(S)
```

con

```
P * S = L * U
```

```
R = chol(S)
```

```
[Q,R] = qr(S)
```

Fattorizzazioni incomplete: `luinc` `cholinc`

utili come *precondizionatori* per metodi iterativi per la soluzione di sistemi di equazioni con matrici sparse.

37

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

Sistemi di equazioni lineari con matrici sparse:

- Metodi diretti (LU, Cholesky);
- Metodi iterativi:

```
bicg          cgs          pcg
```

```
bicgstab     gmres       qmr
```

`pcg` limitato a matrici simmetriche definite positive.

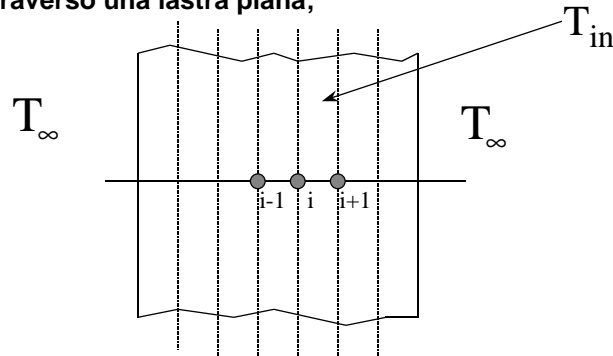
38

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

- Vediamo un esempio (file `heat1d.m`): transitorio termico attraverso una lastra piana;



39

MATLAB - Introduzione

E. Nobile - DINMA - Sezione di Fisica Tecnica, Università di Trieste

Matrici sparse (cont.)

- E' interessante notare l'uso di:
divisione a sinistra \;
`spdiags`;
`spy`;
`subplot`;
`moviein`, `getframe`, `movie`.

40