PROVA PRATICA di ANALISI NUMERICA

Dott. S. De Marchi

`Verona,.....`

Il candidato dovrà scrivere su **ogni** foglio il cognome, nome, numero di matricola. I fogli per le stampe saranno forniti da chi fa assistenza.

1 Dato $n \geq 2$ si considerino le matrici $A_k, k = 1, ..., n-1$ di dimensione n definite come segue:

$$\begin{cases} a^k_{i,i} = n & \\ a^k_{i,j} = -1 & |i-j| = k \\ 0 & \text{altrimenti} \end{cases} .$$

Sia inoltre $\mathbf{b} = \texttt{ones}(\texttt{n}, 1)$.

Si risolvano gli $n-1$ sistemi lineari

$$A_k \mathbf{x} = \mathbf{b}$$

con il metodo iterativo di Jacobi. La M-function che implementa il metodo di Jacobi dovrà restituire la soluzione, il numero di iterazioni e il massimo autovalore in modulo della matrice di iterazione di Jacobi.

Provare servendosi di alcuni grafici, che all'aumentare di k (la grandezza della banda) il numero di iterazioni decresce fino a che $k = \texttt{floor}((n+1)/2)$ per poi stabilizzarsi. Perchè?

Facoltativo: sia ora $\mathbf{b} = 1 : n$ e $k = n-1$. Risolvere il sistema $A_{n-1}\mathbf{x} = \mathbf{b}$ sia con Jacobi che con Gauss-Seidel. Quale dei due metodi è più veloce?

2 È noto che se $f \in \mathcal{C}^1[a,b]$ e $x_0, ..., x_n$ sono $n+1$ punti distinti esiste un unico polinomio di grado $2n+1$ che interpola $f(x)$ e $f'(x)$ nei punti x_i. Tale polinomio è quello di *Hermite*

$$H_{2n+1}(x) = \sum_{i=0}^{n} \left[f(x_i) H_{n,i}(x) + f'(x_i) \hat{H}_{n,i}(x) \right] , \tag{1}$$

con $H_{n,i}(x)$ e $\hat{H}_{n,i}(x)$ polinomi di Hermite di grado n, definiti come

$$\begin{aligned} H_{n,i}(x) &= \left[1 - 2(x - x_i) L'_{n,i}(x_i) \right] L^2_{n,i}(x) \\ \hat{H}_{n,i}(x) &= (x - x_i) L^2_{n,i}(x) \end{aligned}$$

ove $L_{n,i}(x)$ è l'i-esimo polinomio di Lagrange di grado n.

Implementare (1) è computazionalmente costoso. Si può alternativamente usare lo schema di interpolazione nella forma di Newton nel seguente modo. Si considerano i punti

$$\{z_0, z_1, z_2, z_3, ..., z_{2n+1}\} = \{x_0, x_0, x_1, x_1, ..., x_n, x_n\}$$

e i corrispondenti valori della funzione f e della sua derivata prima f' nei punti, cosicchè il polinomio $H_{2n+1}(x)$ si può scrivere nella **forma equivalente**

$$\begin{aligned} H_{2n+1}(x) &= q_{0,0} + q_{1,1}(x-x_0) + q_{2,2}(x-x_0)^2 + q_{3,3}(x-x_0)^2(x-x_1) \\ &+ q_{4,4}(x-x_0)^2(x-x_1)^2 + \\ &+ \cdots + q_{2n+1,2n+1}(x-x_0)^2(x-x_1)^2\cdots(x-x_{n-1})^2(x-x_n)\ . \end{aligned} \tag{2}$$

Il problema è quindi ricondotto a determinare i coefficienti $q_{i,i}$, come per lo schema di interpolazione di Newton. In pratica $q_{0,0} = f[z_0]$, $q_{1,1} = f[z_0, z_1]$ ecc..

Sia **Algoritmo 1** lo schema alle differenze divise che calcola i suddetti coefficienti, restituendo l'array $(q_{0,0}, q_{1,1}, ..., q_{2n+1,2n+1})$.

Scrivere un programma Matlab che implementa l'**Algoritmo 1** e costruisce il polinomio di interpolazione di Hermite mediante (2). Si consideri $f(x) = \sin(\mathrm{e}^x - 2), \quad x \in [0, 2]$.

Produrre anche il grafico di f e del polinomio interpolante.

Qual è il massimo errore tra la funzione e l'interpolante? L'errore ha analogie con l'errore d'interpolazione classico (solo sui dati)?

$\diamond\diamond$

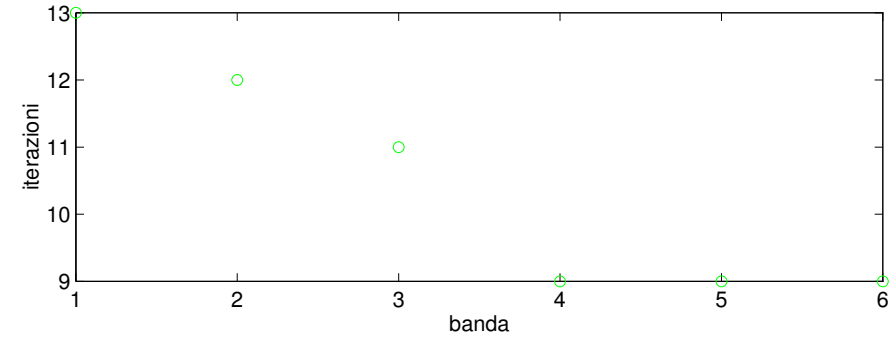
Chi si presenta per il *I o II modulo* deve risolvere il primo o il secondo esercizio in 1 ora e mezza.
Chi si presenta per il *I e II modulo* deve risolvere i due esercizi in 2 ore e mezza.

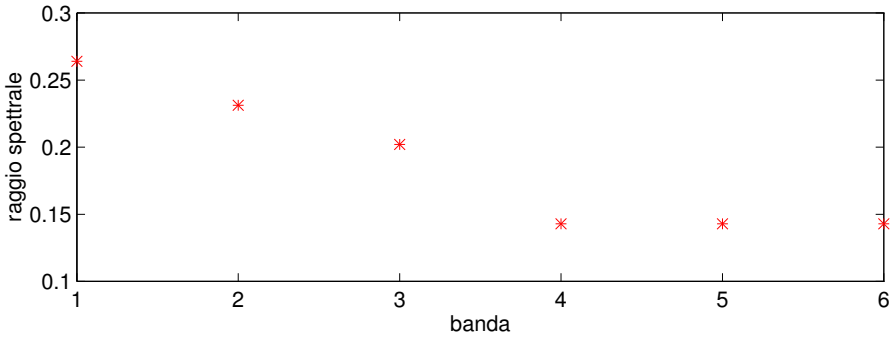
SOLUZIONI

```
%------------------------------
% I Compito
%------------------------------

clear;

n=input('Dimensione matrice = ');
for k=1:n-1,
% definisco la matrice a
a=zeros(n);
for i=1:n,
  a(i,i)=n;
   for j=1:n,
    if(abs(j-i)==k)
      a(i,j)=-1;
    end;
   end;
b(i)=1;
end;

x0=zeros(n,1);
nmax=100;
toll=1.e-8;
k

[xn,iter,maxe]=jacobi1(a,b',x0,nmax,toll);
it(k)=iter;
mme(k)=maxe;
end;

subplot(211);
plot(it,'og');
xlabel('banda');
ylabel('iterazioni');
subplot(212);
plot(mme,'*r');
xlabel('banda');
ylabel('raggio spettrale');

% ----------------------
```

```
% Parte Facoltativa
%------------------------
b=1:n;
[xv,iter]=gaussril(a,b',x0,nmax,toll,1);
disp('Soluzione con GS');
xv
iter

[xn,iter,maxe]=jacobi1(a,b',x0,nmax,toll);
disp('Soluzione con Jacobi');
xn
iter

%%% Funzione: jacobi1

function [xn,iter,me]=jacobi(a,f,xv,nmax,toll)

%----------------------------------------
% Metodo di Jacobi per sistemi lineari.
%----------------------------------------
% Parametri in ingresso
%
%  a     Matrice del sistema
%  f     Termine noto (vettore riga)
%  xv    Vettore iniziale (vettore riga)
%  nmax  Numero massimo di iterazioni
%  toll  Tolleranza sul test d'arresto (fra iterate)
%
% Parametri in uscita
%
%  x     Vettore soluzione
%  iter  Iterazioni effettuate
%  me    raggio spettrale matrice di Jacobi
%

[n,m]=size(a);
d=diag(diag(a));
b=eye(n)-d\a;
me=max(abs(eig(b)));
g=d\f;
i=0;
while i<nmax,
```

```
  xn=b*xv+g;
  if abs(xn-xv)<toll,
     iter=i; i=nmax+1;
  else
     xv=xn; i=i+1;
  end,
end
%
if i==nmax,
  disp('** Non converge nel numero di iterazioni fissato')
end
%
return



%%% La funzione 'gaussril' e' la stessa usata nel
%%% appello del 20.6.00 a cui si rimanda.



%%%%%%%%%%  RISULTATI
n=7


Soluzione con GS

xv =

    0.2917
    0.2857
    0.4286
    0.5714
    0.7143
    0.8571
    1.0417


iter =

     6
```






```
Soluzione con Jacobi

xn =

    0.2917
    0.2857
    0.4286
    0.5714
    0.7143
    0.8571
    1.0417


iter =

    10
```

```
%----------------------------
% II compito del 13.7.2000
%----------------------------
clear;
a=0;
b=2;
n=3;

step=(b-a)/3;
x=a:step:b;

y=sin(exp(x)-2);
y1=cos(exp(x)-2).*exp(x);

%--------------------------------------------------
% Calcolo i coeff. del polinomio interp. di Hermite
%--------------------------------------------------
m=length(x);
m1=2*n+2;

for i=1:m,
  z(2*i-1)=x(i);
  z(2*i)=x(i);
  q(2*i-1,1)=y(i);
  q(2*i,1)=y(i);
  q(2*i,2)=y1(i);

  if (i~=1),
    q(2*i-1,2)=(q(2*i-1,1)-q(2*i-2,1))/(z(2*i-1)-z(2*i-2));
  end;
end;

  for i=3:m1,
    for j=3:i,
     q(i,j)=(q(i,j-1)-q(i-1,j-1))/(z(i)-z(i-j+1));
    end;
  end;


for i=1:m1,
   qq(i)=q(i,i);
end;
```

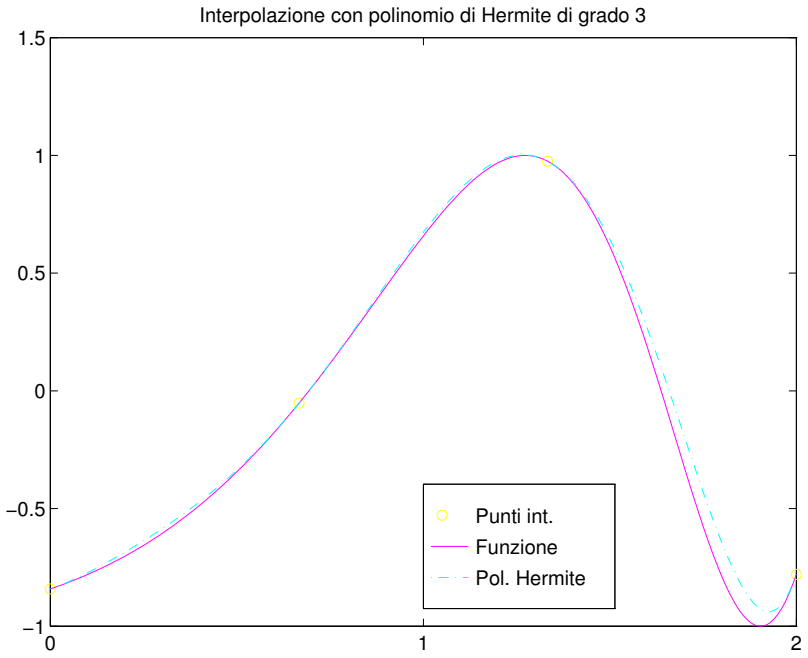
```
xx=0:.01:2;
yy=sin(exp(xx)-2);

%-------------------------------------
% Valutazione del polinomio di Hermite
% -------------------------------------

for r=1:length(xx),
pp(r)=qq(1);
pow=xx(r)-z(1);
  for i=2:m1,
    pp(r)=pp(r)+qq(i)*pow;
    pow=pow*(xx(r)-z(i));
  end;
end;


%---------------------------------------
% Plot dell'interpolante e della funzione
%---------------------------------------

plot(x,y,'o',xx,yy,'-',xx,pp,'-.')
legend('Punti int.','Funzione','Pol. Hermite');
titlestring=['Interpolazione con polinomio di Hermite di grado ', num2str(n)];
title(titlestring);
% -------------------
% Stima dell'errore
% -------------------
disp('Massimo errore tra funzione e poli. di Hermite ');
MaxErr=max(abs(sin(exp(xx)-2) - pp))

disp('Errore teorico ... ');
% La derivata 8^a e' (fatti i calcoli)


% La derivata m1-esima e' (fatti i calcoli)
yy1=sin(exp(xx)-2).*exp(8*xx)-28*cos(exp(xx)-2).*exp(7*xx)- 266*sin(exp(xx) - 2).*exp(6*xx);
yy2=1050*cos(exp(xx) - 2).*exp(5*xx) + 1701*sin(exp(xx) - 2).*exp(4*xx);
yy3=966*cos(exp(xx) - 2).*exp(3*xx) - 127*sin(exp(xx) - 2).*exp(2*xx) + cos(exp(xx) - 2).*exp(
yym1=yy1+yy2+yy3;
disp('Modulo derivata 8^a ');
```

```
M=max(abs(yym1))

err=step^m1/prod(1:m1)*M


%%% Risultati
Massimo errore tra funzione e poli. di Hermite

MaxErr =

    0.1473

Errore teorico ...

Modulo derivata 8^a

M =

   1.9239e+07


err =

   19.5507
```

L'errore che si ha con il polinomio di Hermite è

$$\frac{f^{2n+2}(\xi)}{(2n+2)!}\prod_{i=0}^{n}(x-x_i)^2$$

con $\xi \in \min_{x,x_i}$, ovvero il più piccolo intervallo che contiene x e i punti x_i. Il risultato di sopra è infatti consistente poichè esso è di tipo $\mathcal{O}(h^{2n+2})$.


Interpolazione con polinomio di Hermite di grado 3
1.5
1
0.5
0
−0.5
−1
0
1
2
Punti int.
Funzione
Pol. Hermite