PROVA PRATICA di CALCOLO NUMERICO

Dott. S. De Marchi

`Verona, 23 marzo 2005`

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

1. Si consideri l'equazione $x = e^{-x}$.

   - Individuato un intervallo che contiene la radice, usando l'iterazione

   $$x_{n+1} = \frac{e^{-x_n} + x_n}{2}, \quad n \geq 0 \tag{1}$$

   si determini la radice α dell'equazione data con $tol = 1.e-6$.

   - Si prenda ora l'iterazione

   $$x_{n+1} = \frac{\omega e^{-x_n} + x_n}{1+\omega}, \quad n \geq 0, \ \ \omega \neq 0, \ \omega \neq -1\ , \tag{2}$$

   Determinata α, graficamente si dica per quali valori di ω l'iterazione (2) converge più rapidamente di (1) (*Sugg.* si calcoli in α la derivata della funzione d'iterazione (2))

   - Qual è il valore ottimale di ω?

2. Data la matrice *simmetrica*.

$$A = \begin{pmatrix} 4 & 1 & 1 & 0 \\ 1 & 3 & 2 & 3 \\ 1 & 2 & -1 & -2 \\ 0 & 3 & -2 & 5 \end{pmatrix}$$

   (a) mediante il metodo delle potenze determinare l'autovalore di modulo massimo M, con precisione $tol = 1.e-6$;

   (b) mediante il metodo delle potenze inverse determinare l'autovalore di modulo minimo m, con precisione $tol = 1.e-6$;

   (c) si determinino infine gli altri due autovalori (interni a $[m, M]$.)

3. Calcolare numericamente

$$\int_{-1}^{1} (1+x^2)\sqrt{1-x^2}\, dx$$

   usando il metodo di Simpson composito. Quanti punti sono necessari affinchè l'errore assoluto sia $< 1.e-4$? Come valore *esatto*, considerare il valore dell'integrale ottenuto con `quadl` a meno di $1.e-6$.

`Tempo:` 3 ore.

```
%****************************************
% Soluzioni del Compito del 23 marzo 2005
%****************************************
%--------------------
% Esercizio 1
%--------------------
% PARTE (a)
format long
x0=0;
x1=(exp(-x0)+x0)/2;
tol=1.e-6;
kmax=1000;
k=1;

while abs(x1-x0)/abs(x1) > tol & k <=kmax
    x0=x1;
    x1=(exp(-x0)+x0)/2;
    k=k+1;
end
disp('La radice cercata a meno di tol e'' '); x0

% PARTE (b)
alpha=x0;
iter=[];s=1;
w=0.1:0.05:2;
for i=1:length(w)
x0=0;
x1=(w(i)*exp(-x0)+x0)/(1+w(i));
tol=1.e-6;
kmax=1000;
k=1;
while abs(x1-alpha) > tol & k <=kmax
    x0=x1;
    x1=(w(i)*exp(-x0)+x0)/(1+w(i));
    k=k+1;
end iter(s)=k; s=s+1; end
bar(w,iter);
[miniter,imin]=min(iter);
disp('il numero minimo di
iterazioni e'' '),miniter
```

```
disp('che corrisponde al valore di w  =  '), w(imin)

%--------------------------------------------------------------------
% PARTE (c)
%
% Derivando la funzione d'iterazione calcolata in alpha,
% si ottiene il valore ottimale di w.
% Il valore ottimale e' w=exp(alpha) (che implica
% che la derivata della funzione d'iterazione si annulla!). In pratica
% per quel valore di w, il metodo e' del secondo ordine come il metodo
% di Newton.
%
% Numericamente poi, si ottiene che w in un intorno di 1.7 e'
% il valore cercato.
%--------------------------------------------------------------------
-----> Esecuzione

>> esame23marzo2005esI
La radice cercata a meno di tol e'

x0 =

   0.56714298529873

il numero minimo di iterazioni e'

miniter =

     4

che corrisponde al valore di w  =

ans =

   1.70000000000000



%-----------------------------
% Esercizio 2
%-----------------------------

% PARTE (a)
```

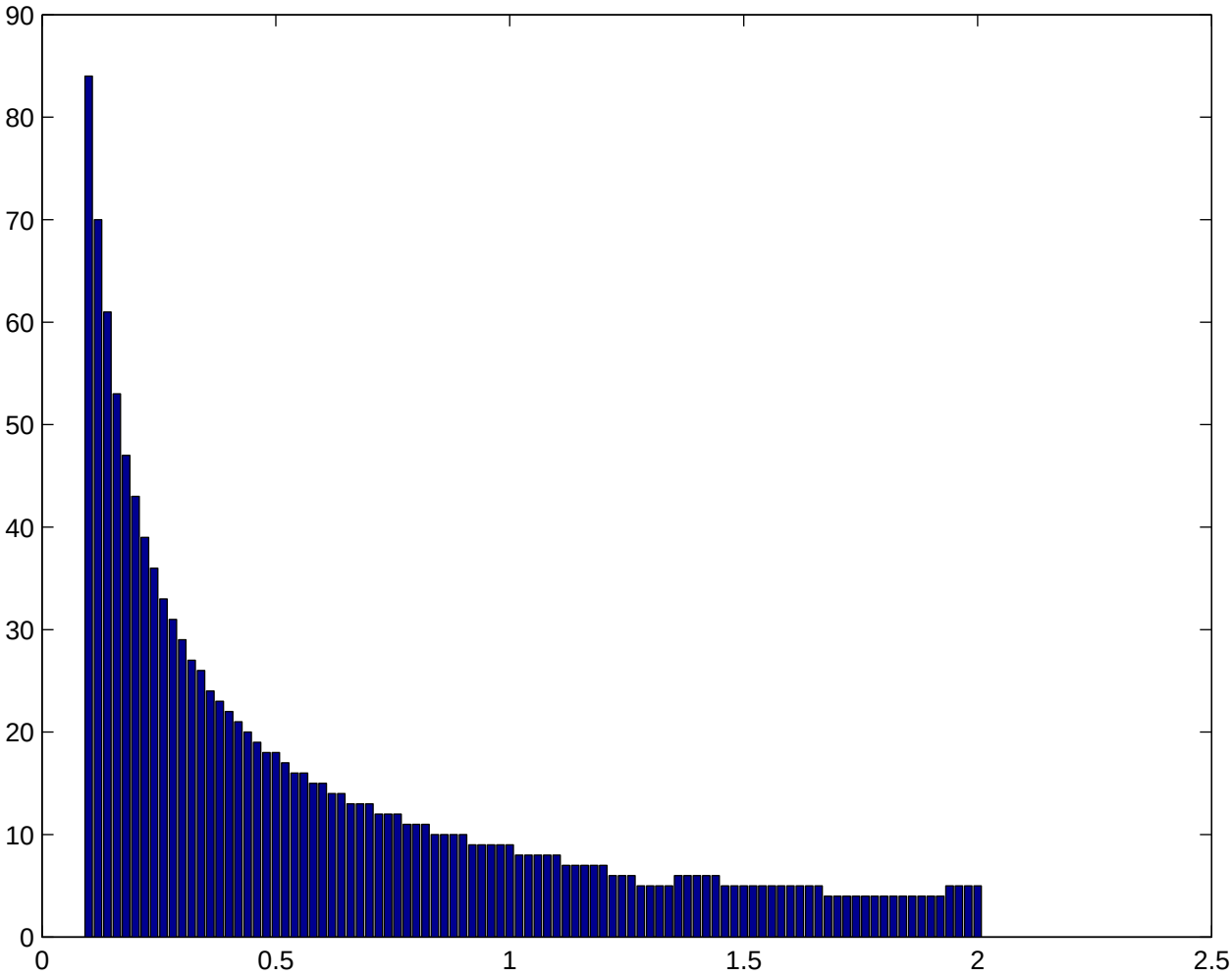

Figure 1: Esercizio 1 (b): iterazioni al variare di $\omega \in [0, 2]$, con passo di 0.05. Il minimo si ottiene in un intorno di $\omega \approx 1.7 \approx e^{\alpha}$.

```
clear
%-------------------------------------------
% Metodo delle potenze per il
% calcolo dell'autovalore di modulo massimo
%-------------------------------------------
tol=1.e-6;
% -----> Matrice  <-----
mat=[4 1 1 0; 1 3 2 3 ; 1 2 -1 -2; 0 3 -2 5];
%-----------------------------------------------
[n,m]=size(mat);
kmax=2000;
x0=ones(n,1);
y0=x0/norm(x0);
x1=mat*y0; lambda0=y0'*x1;
y1=x1/norm(x1);
lambda1=y1'*mat*y1;
k=0;
while( k <= kmax & abs(lambda1-lambda0)>tol*abs(lambda1) & abs(lambda1)>0 )
    lambda0=lambda1;
    x1=mat*y1;
    y1=x1/norm(x1);
    lambda1=y1'*x1;
    k=k+1;
end
disp('iterazioni fatte = '); k
disp ('autovalore di modulo massimo = '); lambda1

% PARTE (b)
%-------------------------------------------
% Metodo delle potenze inverse per il
% calcolo dell'autovalore di modulo minimo
%-------------------------------------------
a=inv(mat);
[n,m]=size(a);
x0=ones(n,1);
y0=x0/norm(x0);
x1=a*y0; lambda0=y0'*x1;
y1=x1/norm(x1);
lambda1=y1'*a*y1;
k=0;
while( k <= kmax & abs(lambda1-lambda0)>tol*abs(lambda1) & abs(lambda1)>0 )
    lambda0=lambda1;
```

```
    x1=a*y1;
    y1=x1/norm(x1);
    lambda1=y1'*x1;
    k=k+1;
end
disp('iterazioni fatte = '); k
disp ('autovalore di modulo minimo = '); 1/lambda1

% PARTE (c)

for i=1:2 % calcolo gli altri autovalori rimasti col metodo
          % delle potenze inverse con shift
eta=input('Dammi eta = ');
% ------------------------------------------------------------
% Volendo il valore dello shift si pu individuare
% dapprima costruendo i cerchi di Gerschgorin per dare
% uno "shift" pi preciso
%-------------------------------------------------------------
n=max(size(mat));
a=inv(mat-eta*eye(n));
x0=ones(n,1);
y0=x0/norm(x0);
x1=a*y0; lambda0=y0'*x1;
y0=x1/norm(x1);
x1=a*y0;
lambda1=y0'*x1;
k=0;
while( k <= kmax & abs(lambda1-lambda0)>tol*abs(lambda1) & abs(lambda1)>0 )
    lambda0=lambda1;
    y0=x1/norm(x1);
    x1=a*y0;
    lambda1=y0'*x1;
    k=k+1;
end
disp('iterazioni fatte = '); k
disp ('autovalore pi vicino a eta  = '); 1/lambda1+eta
end


------> Esecuzione

>> esame23marzo2005esII
```

```
iterazioni fatte =

k =

    15

autovalore di modulo massimo =

lambda1 =

   7.27639843947276

iterazioni fatte =

k =

    19

autovalore di modulo minimo =

ans =

   2.01357922900579

Dammi eta = -3
iterazioni fatte =

k =

     2

autovalore pi vicino a eta  =

ans =

  -3.05250821148432

Dammi eta = 4
iterazioni fatte =

k =
```

```
     6

autovalore pi vicino a eta  =

ans =

   4.76252819710185

%-----------------------
% Esercizio 3
%-----------------------
format long
%-------------------------------------------------------
% Applico il metodo di Simpson composito finch l'errore
% assoluto non scende sotto la tolleranza richiesta.
%-------------------------------------------------------
tol=1.e-4;
n=2; % Punti della suddivisione all'inizio.
a=-1;
b=1;
h=(b-a)/n;
x=linspace(a,b,n+1); %punti equispaziati.
realValue=quadl(@funQ,a,b,1.e-6);

fSc=funQ(x);
fSc(2:end-1)=2*fSc(2:end-1);
ValSc=h*sum(fSc)/6;
x=linspace(a+h/2,b-h/2,n);
fSc=funQ(x);
ValSc=ValSc+2*h/3*sum(fSc);

while abs(ValSc-realValue)> tol
    h=h/2; n=2*n;
    x=linspace(a,b,n+1); %punti equispaziati.
    fSc=funQ(x);
    fSc(2:end-1)=2*fSc(2:end-1);
    ValSc=h*sum(fSc)/6;
    x=linspace(a+h/2,b-h/2,n);
    fSc=funQ(x);
    ValSc=ValSc+2*h/3*sum(fSc);
end
plot(x,fSc,'o')
```

```
title('Quadratura composita di Simpson e relativi nodi');
disp('Errore assoluto')
erroreT=abs(realValue-ValSc)
disp(' I punti necessari sono '),
n

function y=funQ(x)
y=(1+x.^2).*sqrt(1-x.^2);
end

------> Esecuzione

>> esame23marzo2005esIII

Errore assoluto

erroreT =

    3.907033526773240e-005

 I punti necessari sono

n =

   512
```

Nota. Relativamente all'esercizio 3, ad ogni passo si è raddoppiato il numero di punti perchè in questo modo il programma è più efficiente. Per completezza, c'è da dire che se si incrementasse n di 1 ad ogni step, ci si arresterebbe con $n = 276$, ma il numero di passi da eseguire sarebbe di un ordine di grandezza più grande. Con $n = 276$ l'errore è leggermente più grande anche se soddisfacente la tolleranza richiesta. Poiché il nostro scopo è di produrre programmi efficienti, l'implementazione qui proposta sarà quella preferibile anche in futuro per situazioni simili.