PROVA PRATICA di CALCOLO NUMERICO

per Matematica Applicata e Informatica Multimediale

Prof. Stefano De Marchi, Dott. Marco Caliari

`Verona, 27 marzo 2008`

Il candidato dovrà scrivere su **ogni** foglio o file che intende consegnare cognome, nome e numero di matricola. Dovrà inoltre produrre uno o più files `.m` contenenti tutte le istruzioni, **dettagliatamente** commentate, necessarie al completo svolgimento degli esercizi. Tutti i files, opportunamente compressi, devono essere inviati via email a `marco.caliari@univr.it`.

1. Sia data l'equazione $1.5x - \tan x = 0.1$.
   (a) Dimostrare che l'equazione ha almeno uno zero, indicato con α, nell'intervallo $[0, \pi/6]$.
   (b) Calcolare due diverse approssimazioni di α, indicate con α_1 e α_2, usando il metodo di bisezione con un numero massimo di tre e quattro iterazioni, rispettivamente.
   (c) Dire quale punto iniziale, tra α_1 e α_2, assicura una successione convergente *monotonicamente* per il metodo di Newton e, a partire da quel punto iniziale, si approssimi lo zero con il metodo di Newton con tolleranza per l'errore assoluto pari a 10^{-8}.
   (d) Dire se la funzione di iterazione

$$g(x) = \arctan(1.5x - 0.1)$$

   rende convergente il corrispondente metodo di iterazione di punto fisso allo zero α e motivare la risposta. Proporre una funzione di iterazione che assicuri la convergenza a α e motivare il risultato.

2. Sia data la matrice

$$A = \begin{bmatrix} 2 & -1 & 1 \\ 2 & 2 & 2 \\ -1 & -1 & 2 \end{bmatrix}$$

   (a) Costruire il termine noto b tale che la soluzione del sistema lineare $Ax = b$ sia $x = [1, 2, 3]^T$.
   (b) Si dica, senza testarli, se i metodi iterativi di Jacobi e di Gauss–Seidel per la risoluzione del sistema lineare $Ax = b$ convergono e motivare la risposta.
   (c) Si implementi una function `[x,iter,stimaerr] = metiter(A,b,x0,tol,maxit,omega)` per la risoluzione di un sistema lineare con un metodo iterativo che accetti in input, tra l'altro, il parametro ω. Per $\omega = 0$, la function deve implementare il metodo di Jacobi, mentre, per $\omega > 0$, il metodo SOR. La stima dell'errore relativo (in norma 2) deve essere basata sullo scarto. Mostrare che il comportamento della function è in accordo con il punto (b) precedente.
   (d) Stimare, a meno di 0.1, il parametro ottimale (cioè che assicura il minor numero di iterazioni), $\omega^\star$. Per tale parametro, si calcoli la soluzione, a partire dal vettore iniziale $x_0 = [0, 0, 0]^T$ con una tolleranza relativa pari a 10^{-8} e la si confronti con l'errore relativo.

3. Sia data la funzione $f(x) = \mathrm{e}^{x+1}$, $x \in [0, 1]$.

   (a) Determinare a piori il numero di punti equispaziati necessari per interpolare $f(x)$, con interpolazione lineare composita, con un errore (in norma infinito) minore di 10^{-6}.

   (b) Far vedere poi che, usando punti di Chebyshev e interpolazione non composita, servono meno di un centesimo dei punti.

   (c) Produrre quindi il grafico di entrambe le interpolanti e calcolare numericamente l'errore assoluto in norma infinito su un insieme di 1000 nodi target equispaziati.

◊◊

Tempo: **3 ore**.

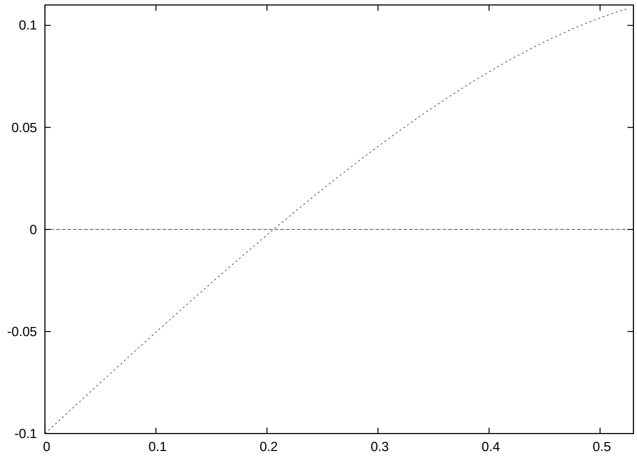
SOLUZIONI

Es. 1. (a) Si ha $1.5 \cdot 0 - \tan(0) - 0.1 = -0.1 < 0$ e $1.5 \cdot \pi/6 - \tan(\pi/6) - 0.1 \approx 0.11 > 0$ e la funzione è continua.

(b) `alpha1 = bisezione(@fesame,0,pi/6,1e-8,3)` e
`alpha2 = bisezione(@fesame,0,pi/6,1e-8,4)` e si trova $\alpha_1 \approx 0.20$ e $\alpha_2 \approx 0.23$.

```
function y = fesame(x)
y = 1.5*x-tan(x)-0.1;
```

(c) Dal grafico, si vede chiaramente che α_1 sta a sinistra di α.



Per il teorema di convergenza globale del metodo di Newton (oppure, provando), si trova che α_1 come punto iniziale assicura una successione monotona.
Poi, `alpha=newton(@fesame,@fesameder,alpha1,1e-8,5)`.

```
function y = fesameder(x)
y = 1.5-(1+tan(x).^2);
```

(d) Si ha

$$g'(\alpha) = \frac{1.5}{1 + (1.5\alpha - 0.1)^2} \approx 1.44 > 1$$

e dunque non convergenza. Prendendo invece $g(x) = (0.1 + \tan x)/1.5$, si ha

$$g'(\alpha) = \frac{1 + \tan^2 \alpha}{1.5} \approx 0.7 < 1$$

e dunque convergenza.

Es. 2. (a) `b = A*[1,2,3]'`.

(b) `D=diag(diag(A))`, `E=-(tril(A)-D)`, `F=-(triu(A)-D)`. Jacobi

```
> max(abs(eig(D\(E+F))))
ans =  1.1180
```

non converge. Gauss–Seidel

```
> max(abs(eig((D-E)\F)))
ans =  0.50000
```

converge.

(c)
```
function [x,iter,errest] = metiter(A,b,x0,tol,maxit,omega)
n = length(x0);
errest = tol+1;
iter = 0;
x = x0;
while (errest > tol) & (iter <= maxit)
  iter = iter+1;
  for i = 1:n
    if (omega > 0)
      xtilde(i) = (b(i)-A(i,1:i-1)*x(1:i-1)-A(i,i+1:n)*x0(i+1:n))/A(i,i);
      x(i) = omega*xtilde(i)+(1-omega)*x0(i);
    else
      x(i) = (b(i)-A(i,1:i-1)*x0(1:i-1)-A(i,i+1:n)*x0(i+1:n))/A(i,i);
    end
  end
  errest = norm(x-x0)/norm(x);
  x0 = x;
end
if (iter > maxit)
  warning('Raggiunto il numero massimo di iterazioni.')
end

octave:67> [x,iter,errest]=metiter(A,b,[0,0,0]',1e-8,100,0)
warning: Raggiunto il numero massimo di iterazioni.
x =

   3.5033e+04
   2.8026e+05
  -1.0509e+05

iter =  101
errest =  1.2823

octave:68> [x,iter,errest]=metiter(A,b,[0,0,0]',1e-8,100,1)
x =

   1.0000
   2.0000
   3.0000
```

```
iter =  34
errest =  6.5386e-09
```

(d)
```
octave:69> [x,iter,errest]=metiter(A,b,[0,0,0]',1e-8,100,0.7)
x =

   1.00000
   2.00000
   3.00000

iter =  20
errest =  8.1799e-09
octave:70> [x,iter,errest]=metiter(A,b,[0,0,0]',1e-8,100,0.8)
x =

   1.00000
   2.00000
   3.00000

iter =  18
errest =  5.4193e-09
octave:71> [x,iter,errest]=metiter(A,b,[0,0,0]',1e-8,100,0.9)
x =

   1.00000
   2.00000
   3.00000

iter =  20
errest =  3.0851e-09
octave:72> [x,iter,errest]=metiter(A,b,[0,0,0]',1e-8,100,0.8)
x =

   1.00000
   2.00000
   3.00000

iter =  18
errest =  5.4193e-09
octave:73> norm(x-[1,2,3]')/norm([1,2,3]')
ans =  2.0788e-09
```

Es. 3. Per rispondere alla prima parte, basta far uso della Proposizione 11 e della formula dell'errore (5.38) ovvero

$$\max_{x\in[0,1]} |f(x) - p_1^c(x)| \leq \max_{x\in[0,1]} |f^{(2)}(x)| \, \frac{h^2}{8} \, .$$

Sapendo che la $f^{(2)}(x) = \mathtt{e}^{x+1}$ e pertanto il suo massimo vale e^2, si deduce $h^2 \leq \dfrac{8 \cdot 10^{-6}}{e^2}$. Ora, sapendo che $h = 1/n$, ricaviamo che

$$n \geq \sqrt{\frac{e^2}{8 \cdot 10^{-6}}} = \frac{\mathtt{e} \cdot 10^3}{\sqrt{8}} \approx 961.06$$

I punti richiesti sono almeno 962.

Nel caso di punti di Chebsyshev, basta usare la disuguaglianza (5.39)

$$\|f - p_n\|_\infty \leq \frac{\|f^{(n+1)}\|_\infty}{(n+1)!} 2^{-n}$$

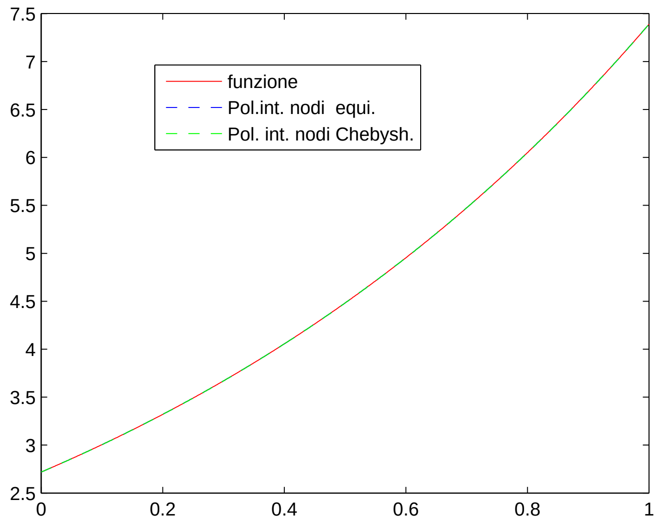
che risulta verificata per $n = 8$. Infatti, sapendo che $f^{(n)}(x) = \mathtt{e}^{x+1}$ e quindi che $\|f^{(n)}\|_\infty = e^2$ si ottiene

```
n=8; exp(2)/(factorial(n+1)*2^n)

ans =
  7.9540e-008
```

Infine, per fare il plot dei polinomi interpolatori, basta usare `interp1` di Matlab/Octave per costruire l'interpolante lineare a tratti, mentre per determinare il polinomio d'interpolazione di Chebyshev, potremo usare la forma di Newton. Gli errori calcolati sono come segue:

```
>> esame27marzo2008esIII
Errore d'interpolazione in norma infinito: nodi equispaziati

err =

  9.8380e-007

Errore d'interpolazione in norma infinito: nodi Chebyshev

err =

  1.1642e-007
```

Si noti come le stime siano appunto delle sovrastime. Infatti, nel caso di Chebyshev bastano solo 7 punti e non 9.


funzione
Pol.int. nodi equi.
Pol. int. nodi Chebysh.
7.5
7
6.5
6
5.5
5
4.5
4
3.5
3
2.5
0
0.2
0.4
0.6
0.8
1