Calcolo Numerico: L.T. Ing. Aerospaziale
(Prof. G. Zilli)

1a. Esercitazione (A.A. 2009/2010)
Sulla Soluzione di Equazioni non Lineari

Dopo aver dimostrato esistenza ed unicità della soluzione ξ in I, si risolva utilizzando il linguaggio MatLab, l'equazione $f(x) = 0$, con

$$f(x) = 3x^2 - 1 + \log x, \qquad I = [0.2,\ 1].$$

Si applichino, a scelta (almeno 3), i metodi:
1. di Newton-Raphson (e della tangente fissa)
2. della secante variabile (e della secante fissa)
3. della bisezione
4. del punto fisso.
Si prenda $x_0 = 1$ come punto iniziale. Per i 2 metodi delle secanti si prenda x_1 ottenuto con il metodo di Newton-Raphson.
Con tutti i metodi, calcolare l'approssimazione x_k della radice ξ, usando il test di arresto sullo scarto: $|s_k| = |x_{k+1} - x_k| < TOLL$, (ad esempio con una tolleranza $TOLL = 10^{-6}$). Porre a 50 il numero massimo di iterazioni consentite.

Tutti i metodi vengano implementati nel seguente modo:

- per il metodo di Newton-Raphson, per ogni iterata k-esima si riporti: k, x_k, $f(x_k)$, $f'(x_k)$, lo scarto s_k, e si verifichi se l'ordine di convergenza è quadratico considerando il rapporto $|s_{k+1}|/|s_k|^2$ (stima della costante asintotica M).

- per il metodo della secante variabile, per ogni iterata k-esima si riporti: $k,\ x_k,\ f(x_k),\ \dfrac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$, lo scarto s_k, e si verifichi se l'ordine di convergenza è $p = 1.6181$, considerando il rapporto $|s_{k+1}|/|s_k|^p$ (stima della costante asintotica M).

- Si proceda analogamente per i 2 metodi della tangente fissa e della secante fissa (e della bisezione). Considerando il rapporto fra gli scarti $|s_{k+1}|/|s_k|$ si verifichi se l'ordine di convergenza è $p = 1$, stimando la costante asintotica M e quindi la velocità di convergenza R. Di conseguenza, si stimi il numero k di iterazioni necessarie per avere $|e_k|/|e_0| < TOLL$, confrontandolo con il risultato sperimentale.

- Per il metodo del punto fisso, occore considerare un'opportuna funzione di punto fisso $g(x)$. Si suggeriscono le seguenti 2 funzioni:

$$\begin{cases} g(x) = & \sqrt{\dfrac{1-\log x}{3}} \\ g(x) = & \exp\left(-3x^2+1\right) \end{cases}$$

  Prendendo come punto iniziale $x_0 = 1$, per ogni l'iterata k-esima si riporti:
  k, x_k, $g(x_k)$, lo scarto $|s_k| = |x_{k+1} - x_k|$. Considerando il rapporto $|s_{k+1}|/|s_k|$ si verifichi se l'ordine di convergenza è $p = 1$, stimando la costante asintotica M e quindi la velocità di convergenza R. Di conseguenza, stimare il numero k di iterazioni necessarie per avere $|e_k|/|e_0| < TOLL$, confrontandolo con il risultato sperimentale.
  Commentare la convergenza o divergenza di ciascuno dei due schemi alla luce della teoria.

Si riporti in grafico semilogaritmico lo scarto s_k in funzione del numero delle iterate (un unico grafico in cui vi sono le curve di convergenza per i vari metodi impiegati), commentando brevemente i risultati.
Si costruisca anche il grafico della funzione f in I, per esempio discretizzando l'intervallo con punti equidistanti (si scelga un numero di punti compreso tra 20 e 30) e riportando su due colonne i valori dei punti e della funzione nei punti stessi.

Si scriva una breve relazione in un documento di testo, in cui si descrive il problema, i risultati ottenuti con i metodi utilizzati e il loro confronto, il numero delle iterazioni richieste per soddisfare il test di arresto, e l'ordine di convergenza (includendo e descrivendo i grafici). Si alleghino inoltre: i programmi Matlab che implementano i metodi sopra elencati, i file di input e i file di output generati.
Non sono ammesse fotocopie; portare tutto il materiale prodotto in originale.

Altre **funzioni** suggerite su cui ripetere, a scelta, l'**Esercitazione**:

$$f(x) = e^x - \ln(1-x) = 0, \qquad I = [-1, 0] \quad x_0 = -0.9.$$

$$f(x) = e^{-3x} - 2x^3 = 0, \qquad I = [0, 1.5] \quad x_0 = 1.$$

$$f(x) = x - \frac{1}{2}e^x + \cos x = 0, \qquad I = [-1, 0] \quad x_0 = -0.6.$$

$$f(x) = e^{-x} - \sin x = 0, \qquad I = [0, 1] \quad x_0 = 0.$$

Segue un semplice programma MatLab (*newton.m*, di tipo script) che implementa il metodo di Newton-Raphson applicato all'equazione

$$f(x) = e^{-x} - x^2 = 0, \qquad I = [0,1], \quad x_0 = 1.$$

La funzione e la derivata prima $f'(x) = (-e^{-x} - 2x)$ sono dati nel programma. Si consiglia di ricavarne un programma di tipo function, valido per ogni funzione: *function* [*output1,...*] = *newton(input1,...)*

```
% newton.m
% Newton-Raphson applicato a f(x)= e^{-x} - x^2 = 0,
% valori di input
nmax = 50;
tol = 1.e-8;
scarto = 10;
scarto_old = 10;
xold = 1.;
i = 0;
M = 1;
% fine valori di input
disp('iter            xnew                  scarto                   M')
x1 = sprintf('%3d %22.10e %22.10e %22.10e',i,xnew,scarto,M);
disp(x1);
while i < nmax & abs(scarto) > tol
   fx = exp(-xold) - xold^2;
   dfx = -exp(-xold) - 2*xold;
    if abs(dfx) < tol*abs(fx)
       dfx=dfx+tol*100
    disp('La derivata f''(x0) e'' nulla')
    end
   xnew = xold - fx/dfx;
   i = i + 1;
   scarto = abs(xnew - xold);
   if i>1
      M = scarto/scarto_old^2;
   end;
   x1 = sprintf('%3d %22.10e %22.10e %22.10e',i,xnew,scarto,M);
   disp(x1);
   scarto_old = scarto;
   xold = xnew;
end;
```

RISULTATO

iter	xnew	scarto	M
0	7.0346742250e-001	1.0000000000e+001	1.0000000000e+000
1	7.3304360525e-001	2.6695639475e-001	1.0000000000e+000
2	7.0380778632e-001	2.9235818921e-002	4.1023679091e-001
3	7.0346746833e-001	3.4031799234e-004	3.9815702604e-001
4	7.0346742250e-001	4.5833405071e-008	3.9574214391e-001
5	7.0346742250e-001	7.7715611724e-016	3.6995084483e-001