ESERCITAZIONE MATLAB 3: Equazioni non lineari

1. Si scriva un M-file di tipo FUNCTION che implementa il metodo di Newton per il calcolo di una radice di una equazione non lineare. L'intestazione della FUNCTION deve essere la seguente

```
 function [x,it] = newton(fun,dfun,x0,tol,itmax)

% [x,it] = newton(fun,dfun,x0,tol,itmax)
%
% Implementa il metodo di Newton per il
% calcolo di una radice della equazione
% non lineare
%
% fun(x) = 0
%
% Input: fun  --> funzione di cui si vuole calcolare la radice
%        dfun --> derivata prima di fun
%        x0   --> punto di innesco
%       tol   --> tolleranza per il criterio di arresto
%      itmax  --> numero massimo di iterazioni consentite
%
% Output: x   --> approssimazione della radice calcolata
%         it  --> numero di iterazioni applicate per il
%                 calcolo di x
```

   Si salvi l'M-file con il nome `newton.m`.

2. Al fine di verificare se il metodo di Newton è stato implementato correttamente lo si utilizzi per calcolare la soluzione di

$$f(x) \equiv x - 1 = 0.$$

   Per fare questo, si eseguano le seguenti istruzioni da prompt dei comandi

```
>> fun1 = @(x) x-1;
>> dfun1 = @(x) 1;
```

   Si lanci poi un certo numero di volte l'esecuzione della function `newton` come segue

```
>> [x,it] = newton(fun1,dfun1, x0, 1e-14, 100)
```

   sostituendo `x0` con vari valori. Se il metodo di Newton è stato implementato correttamente allora si deve ottenere `x` = `1` e `it` = `1` (o, al massimo, `it` = `2`) per ogni `x0`.

3. Si utilizzi il metodo di Newton implementato per calcolare la soluzione della equazione

$$f(x) \equiv \mathtt{e}^{x-1} - 1 = 0.$$

   Per fare questo, si eseguano i seguenti comandi

```
>> fun2 = @(x) exp(x-1)-1;
>> dfun2 = @(x) exp(x-1);
```

In modo analogo all'esercizio precedente, si esegua poi un certo numero di volte il seguente comando

```
>> [x,it] = newton(fun2,dfun2, x0, tol, 100)
```

specificando in input vari valori di `x0` e di `tol`.

4. È ben noto che la seguente equazione

$$f(x) \equiv \arctan(x) = 0$$

ammette come unica soluzione $x^* = 0$. Si applichi il metodo di Newton per determinarla e si verifichi che il metodo converge soltanto se x_0 è scelto "sufficientemente" vicino a x^*.

5. Si utilizzi il metodo di Newton per calcolare la soluzione della seguente equazione

$$f(x) \equiv (x-1)^2 \mathtt{e}^x = 0 \tag{1}$$

utilizzando `x0 = 2` come approssimazione iniziale della radice $x^* = 1$ che ha molteplicità $m = 2$. Si scriva poi un M-file di tipo FUNCTION, da salvare con il nome `newtonmod.m`, che implementi la versione modificata del metodo di Newton per la approssimazione di radici multiple con molteplicità nota a priori. L'intestazione della nuova function deve essere la seguente

```
function [x,it] = newtonmod(fun,dfun,x0,tol,itmax,m)
```

Si utilizzi quindi `newtmod` per risolvere (1) e si verifichi che il metodo di Newton modificato converge molto più velocemente. Si ripeta lo stesso esperimento per l'equazione

$$f(x) \equiv (x-1)^3 \mathtt{e}^x = 0. \tag{2}$$

SOLUZIONI:

1.
```
function [x,it] = newton(fun,dfun,x0,tol,itmax)

% [x,it] = newton(fun,dfun,x0,tol,itmax)
%
% Implementa il metodo di Newton per il calcolo di una radice
% della equazione non lineare
%
% fun(x) = 0
%
% Input: fun  --> funzione di cui si vuole la radice
%        dfun --> derivata prima di fun
%        x0   --> punto di innesco
%       tol   --> tolleranza per il criterio di arresto
%      itmax  --> numero massimo di iterazioni consentite
%
% Output: x   --> approssimazione della radice calcolata
%         it  --> numero di iterazioni applicate per il
%                 calcolo di x


f   = feval(fun,x0);
if (f==0),
    x=x0;
    it=0;
    return,
end

df  = feval(dfun,x0);
if (df==0),
   error('Metodo di Newton non applicabile'),
end

x = x0 - f/df;

it = 1;

while ((abs(x-x0)>tol)&(it<itmax))

    x0 = x;

    f   = feval(fun,x0);
    if (f==0),
        return
    end

    df  = feval(dfun,x0);
    if (df==0),
       error('Metodo di Newton non applicabile'),
    end
```

```
    x = x0 - f/df;

    it = it + 1;
end

if (abs(x-x0)>tol),
   error('Il metodo di Newton non converge'),
end
```

2. ```
   >> fun1 = @(x) x - 1;
   >> dfun1 = @(x) 1;
   >> [x,it] = newton(fun1,dfun1,2,1e-14,100)
   x =

        1

   it =

        1

   >> [x,it] = newton(fun1,dfun1,10,1e-14,100)

   x =

        1

   it =

        1

   >> [x,it] = newton(fun1,dfun1,-5,1e-14,100)

   x =

        1

   it =

        1
   ```

3. ```
   >> format long
   >> fun2 = @(x) exp(x-1) - 1;
   >> dfun2 = @(x) exp(x-1);
   >> [x,it] = newton(fun2,dfun2,5,1e-3,100)

   x =

      1.000000000102263

   it =
   ```

```
     8

>> [x,it] = newton(fun2,dfun2,5,1e-10,100)

x =

     1

it =

     9

>> [x,it] = newton(fun2,dfun2,-1,1e-3,100)

x =

   1.000000028921676

it =

     9

>> [x,it] = newton(fun2,dfun2,-1,1e-10,100)

x =

     1

it =

    11
```

4.
```
>> fun3 = @(x) atan(x);
>> dfun3 = @(x) 1./(1+x.^2);
>> [x,it] = newton(fun3,dfun3,1,1e-10,100)

x =

   0


it =

     5

>> [x,it] = newton(fun3,dfun3,1.5,1e-10,100)
??? Error using ==> newton at 48
Metodo di Newton non applicabile
```

   Si osserva che scegliendo $x_0 = 1.5$, la successione di approssimazioni fornita dal metodo

di Newton ha segno alterno ed è divergente in modulo. Dato che $f'(x) \to 0$ per $x \to \pm\infty$, accade che la procedura iterativa viene arrestata poichè, in aritmetica finita, si trova $f'(x_n) = 0$ per un certo indice n;

5. Il codice per la versione modificata del metodo di Newton è il seguente

```
function [x,it] = newtonmod(fun,dfun,x0,tol,itmax,m)

% [x,it] = newtonmod(fun,dfun,x0,tol,itmax,m)
%
% Implementa il metodo di Newton modificato per il calcolo di una radice
% multipla della equazione non lineare
%
% fun(x) = 0
%
% Input: fun  --> funzione di cui si vuole la radice
%        dfun --> derivata prima di fun
%        x0   --> punto di innesco
%       tol   --> tolleranza per il criterio di arresto
%      itmax  --> numero massimo di iterazioni consentite
%          m  --> molteplicita' della radice
%
% Output: x   --> approssimazione della radice calcolata
%        it   --> numero di iterazioni applicate per il
%                 calcolo di x


f   = feval(fun,x0);
if (f==0),
    x=x0;
    it=0;
    return,
end

df  = feval(dfun,x0);
if (df==0),
   error('Metodo di Newton non applicabile'),
end

x = x0 - m*f/df;

it = 1;

while ((abs(x-x0)>tol)&(it<itmax))

    x0 = x;

    f   = feval(fun,x0);
    if (f==0),
        return
    end
```

```
    df  = feval(dfun,x0);
    if (df==0),
       error('Metodo di Newton non applicabile'),
    end

    x = x0 - m*f/df;

    it = it + 1;
end

if (abs(x-x0)>tol),
   error('Il metodo di Newton non converge'),
end
```

Esempi di applicazione:

```
>> format long
>> fun4 = @(x) (x-1).^2.*exp(x);
>> dfun4 = @(x) (x-1).*(x+1).*exp(x);
>> [x,it]=newton(fun4,dfun4,2,1e-6,100)

x =

   1.000000589869591


it =

    22

>> [x,it]=newtonmod(fun4,dfun4,2,1e-6,100,2)

x =

   1.000000000000188


it =

     5
```

Il vantaggio dell'utilizzo di Newton modificato rispetto a Newton classico è ancora più evidente nel caso della equazione (2) (chiaramente specificando tre come ultimo parametro di input per `newtonmod`).