Laboratorio Calcolo Numerico

Esercizio 1

- Si vogliono calcolare le soluzioni approssimate (una è anche determinabile in modo analitico esatto!) dell'equazione non lineare,
$$f(x) = x^2 - 1 + e^{-x} = 0.$$
- Si determinino graficamente, utilizzando le capacità grafiche del Matlab, ed aiutandosi con gli script che disegnano una funzione in un intervallo, le soluzioni dell'equazione $f(x) = 0$ e, si determinino degli intervalli sufficientemente piccoli (di **ampiezza non maggiore di 0.2**) che contengono **una ed una sola soluzione**.
- Si vuole creare uno script di nome `scriptbis.m` ed una function Matlab, di nome `bisezfun.m` che verrà utilizzata all'interno dello script per chiedere l'esecuzione del metodo di bisezione e quindi la determinazione di un'approssimazione della radice reale contenuta nell'intervallo $[a, b]$.
- Si inizi creando lo script. Lo script dovrà prevedere la lettura da tastiera (comando `input`) della stringa (sequenza di caratteri tra apici) che descrive la funzione (che poi nello script dovrà essere trasformata in una *anonymous function*; si veda il testo degli esercizi del precedente laboratorio!), gli estremi dell'intervallo, la tolleranza ed il numero massimo di iterazioni.
  NOTA BENE: la costante e di Nepero si ottiene tramite la funzione Matlab `exp`. Quindi $e \rightarrow$ `exp(1)`, $e^{-x} \rightarrow$ `exp(-x)` ed in generale $e^{\text{esponente}} \rightarrow$ `exp(`esponente`)`.
  Lo script dovrà verificare preventivamente se i valori assunti dalla funzione negli estremi dell'intervallo sono discordi e, dopo l'esecuzione della function, se si è raggiunto il numero massimo di iterazioni.
  Lo script dovrà restituire su video (comando `disp`) il vettore delle iterate `xv` che colleziona tutti i punti medi degli intervalli generati, il vettore dei residui corrispondenti `fxv`, il numero `n` corrispondente all'ultimo valore della successione calcolata ed l'ultima iterata.
  Nello script, alla fine, si preveda anche un grafico che rappresenti (scala logaritmica sull'asse y) il valore assoluto del vettore che contiene i residui contenuti nel vettore (`fxv`).
  Un possibile algoritmo dello script è:

  leggi e assegna f funzione
  leggi e assegna a estremo sinistro
  leggi e assegna b estremo destro
  if $f(a) \times f(b) >= 0$ **then**
      print I valori agli estremi non sono discordi
  else
      leggi e assegna `toll` tolleranza
      leggi e assegna `nmax` numero massimo iterazioni
      esegui `[xv, fxv, n] = bisezfun (f, a, b, toll, nmax)`
      if `n` == `nmax` **then**
          print raggiunto il numero massimo di iterazioni
          print `xv(n)` ultima approssimazione calcolata
          print `fxv(n)` ultimo residuo
      else
          print `xv, fxv`
          print `n`
          print `xv(n)` ultima approssimazione calcolata
          definisci `x(i), i= 1:n` (ascisse SOLO per disegno)

disegna i residui per punti `(x(i),abs(fxv(i))), i= 1:n`
end if
end if

- Si crei poi la function di nome `bisezfun.m`. Tale function deve avere come parametri **in ingresso** la funzione (*anonymous function*), gli estremi dell'intervallo, la tolleranza ed il numero massimo di iterazioni. Come parametri **in uscita** ci dovranno essere il vettore delle iterate `xv` che collezioni tutti i punti medi degli intervalli generati, il vettore dei residui corrispondenti `fxv` ed il numero `n` corrispondente al numero di iterazioni effettuate.
  La function avrà quindi la seguente intestazione:

```
function [xv, fxv, n] = bisezfun (f, a, b, toll, nmax)
%BISEZFUN Metodo di Bisezione
%
% Uso:
% [xv, fxv, n] = bisezfun(f, a, b, toll, nmax)
%
% Dati di ingresso:
%   f:      funzione (anonymous function)
%   a:      estremo sinistro
%   b:      estremo destro
%   toll:   tolleranza richiesta per l'ampiezza
%           dell'intervallo
%   nmax:   massimo numero di iterazioni
%
% Dati di uscita:
%   xv:     vettore contenente le iterate
%   fxv:    vettore contenente i corrispondenti residui
%   n:      numero di iterazioni effettuate
```

  Un possibile algoritmo relativo alla function è:

$$[\mathtt{xv}, \mathtt{fxv}, n] = \textbf{Bisezione}\ (f, a, b, \mathtt{toll}, n_{\max})$$

$n = 0$
$\mathtt{amp} = \mathtt{toll} + 1$
$\mathtt{fa} = f(a)$
while $\mathtt{amp} \geq \mathtt{toll}$ and $n < n_{\max}$ **do**
  $n = n + 1$
  $\mathtt{amp} = |b - a|$
  $\mathtt{xv(n)} = a + \mathtt{amp} \times 0.5$
  $\mathtt{fxv(n)} = f(\mathtt{xv(n)})$
  if $\mathtt{fa} \times \mathtt{fxv(n)} < 0$ **then**
    $b = \mathtt{xv(n)}$
  else if $\mathtt{fa} \times \mathtt{fxv(n)} > 0$ **then**
    $a = \mathtt{xv(n)}$
    $\mathtt{fa} = \mathtt{fxv(n)}$
  else
    $\mathtt{amp} = 0$

end if
end while

- Considerata la soluzione **positiva** della funzione ed utilizzando l'intervallo determinato graficamente, si applichi il metodo di bisezione eseguendo lo script. Si utilizzi come test di arresto il valore $\varepsilon \to \texttt{toll} = \texttt{1e} - 8$ e $n_{\max} = 100$.
- Si provi a ripetere l'esecuzione inserendo $n_{\max} = 20$ e la stessa tolleranza, e si analizzino i risultati ottenuti paragonandoli ai precedenti, in particolare, **quali considerazioni possono essere effettuate relativamente al valore x_n ottenuto?**

Esercizio 2

Si modifichi opportunamente lo script in modo che

- stampi a video quante iterazioni sono necessarie al metodo per ottenere una radice approssimata con un'accuratezza di $\tau = \varepsilon/2$. Suggerimento: si veda la formula a pagina 72 del libro di Calcolo Numerico e la funzione Matlab `ceil` (`ceil(nu)` $\to \lceil \nu \rceil$). Si noti anche che la formula calcola il valore n che corrisponde all'indice dell'iterata, considerando che la prima iterata (punto medio di $[a, b]$) sia indicata con x_0, mentre nel Matlab la prima iterata è `xv(1)` e quindi in Matlab tale formula deve essere modificata.

Esercizio 3 (solo per esperti)

Per una visualizzazione dei dati più accurata, anzichè dare dei comandi Matlab per implementare la visualizzazione
print `xv, fxv`
si crei la seguente funzione, e se ne chieda l'esecuzione nello script al posto delle istruzioni di visualizzazione.

```
function [] = risultati_bis(a,b,f,xv,fxv)
%RISULTATI_BIS function per visualizzare risultati provenienti dal metodo
%   di bisezione per la ricerca degli zeri di equazioni non lineari
% Uso:
% risultati_bis(a, b, f, xv, fxv)
%
% Dati di ingresso:
%   a:          estremo sinistro dell'intervallo
%   b:          estremo destro dell'intervallo
%   f:          funzione di cui cercare lo zero (anonymous function)
%   xv:         vettore contenente le iterate
%   fxv:        vettore contenente i corrispondenti residui
%

xv=xv(:);
fxv=fxv(:);
n=length(xv);
ampv=[(abs(b-a))./(2.^[0:n-1])];
fprintf('\nf: %s \tIntervallo:  a=%g  b=%g  Bisezione \n\n', ...
    func2str(f),a,b);
fprintf('n \t    x_n \t\t\t\t f(x_n) \t\t b_n-a_n\n');
fprintf('%d\t %20.15f \t %10.2e \t %10.4e \n', ...
    [1:n;xv';fxv';ampv]);
```