

**Corso di Laurea Magistrale in
Ingegneria Biomedica
e Ingegneria elettronica e telecomunicazioni per l'energia
Laboratorio di Analisi Numerica
A.A. 2019/2020 – I Ciclo**

Esercitazione 2

Creare una cartella <cognome> in C: dove verranno salvati i file creati nella sessione di lavoro.
Appena entrati in MATLAB posizionarsi in <cognome>.
Risolvere in ambiente MATLAB i seguenti esercizi.

RISOLUZIONE EQUAZIONI/SISTEMI NON LINEARI

L'ambiente MATLAB offre una built-in function per il calcolo degli zeri di una funzione *funz* in una singola variabile vicino ad x_0 :

root=fzero('funz',x0)

che realizza l' Algoritmo di Dekker-Bren (ibrido tra secanti e bisezione)

Mentre la funzione built-in

root=roots(c)

calcola le radici del polinomio i cui coefficienti sono le componenti del vettore *c*.

Se *c* ha $n+1$ componenti allora il polinomio è dato da

$$c(1)*X^n + \dots + c(n)*X + c(n+1).$$

1. Creare due function *fa.m* e *fb.m* per realizzare le seguenti funzioni:

a) $f(x) = 1 - 2xe^{-x/2}$; $[a,b]=[0,1]$

b) $f(x) = 2 + x^{-1} \ln x$; $[a,b]=[0.1,1]$

Nello script *test1.m*, disegnare per punti le due funzioni. Localizzare graficamente gli zeri delle due funzioni per determinare gli iterati iniziali con cui innescare il metodo per determinarne le radici. Determinare gli zeri delle funzioni test (a) e (b), utilizzando la funzione **fzero()** di MATLAB, e visualizzare le radici trovate sul grafico.

2. Si consideri il seguente polinomio:

$$p(x) = 30x^4 - 28x^3 - 110x^2 - 31x + 11$$

- a) Nello script *test2.m* determinare le radici di $p(x)$ mediante la funzione **roots()** di MATLAB.
b) Visualizzare $p(x)$ nell'intervallo $[(x_{\min}-0.1), (x_{\max}+0.1)]$ dove x_{\min} e x_{\max} sono il minimo e massimo tra le radici trovate, disegnare inoltre un marker nella posizione delle radici.

3. Il programma *newton.m* realizza il metodo di Newton per la risoluzione di sistemi non lineari. Si determinino le soluzioni del seguente sistema non lineare:

$$\begin{cases} y = \frac{1}{(x-3)^2} \\ y = -x^3 + 4x \end{cases} \Rightarrow \begin{cases} f_1(x, y) = y - \frac{1}{(x-3)^2} = 0 \\ f_2(x, y) = y + x^3 - 4x = 0 \end{cases}$$

Si memorizzi il sistema nella funzione *funz3.m* e la matrice Jacobiana, determinata in forma analitica, nella funzione *jacf3.m*.

Si consideri come vettore iniziale $x_0=(0,0)$ ' e successivamente $x_0=(2, 1)$ '.

Si confronti il numero di passi richiesti se si considera il vettore iniziale $x_0=(-10, 10)$ '.

4. Scrivere una funzione *bisection.m* che realizzi il metodo di Bisezione.

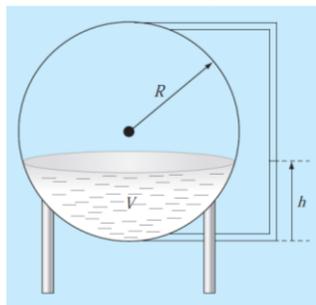
In uno script *test4.m* stampare gli iterati x_k nella risoluzione dell'equazione:

$$f(x) = 2 - e^x \quad \text{in} \quad [0.5, 1]$$

ottenuti con i metodi Newton con $x_0=0$ (*newton.m*), secanti (*secanti.m*) e regola falsi (*regula_falsi.m*) e con quello di bisezione. Confrontare i metodi in termini di numero di iterazioni per raggiungere una data accuratezza (radice esatta $x^* = \ln 2 = 0.6931471806$)

5. You are designing a spherical tank to hold water for a small village in a developing country. The volume of liquid it can hold can be computed as

$$V = \pi h^2 \frac{(3R - h)}{3}$$



where V = volume [m^3], h = depth of water in tank [m], and R = the tank radius [m].

If $R = 3m$, to what depth must the tank be filled so that it holds $30m^3$? In *ex5.m* use bisection method to determine your answer. Determine the number of iterations n required to obtain a solution under the given tolerance $1e-10$. Employ initial guesses of 0 and R .

6. **Calcolo della radice n-esima di un numero.**

Def: la radice n-esima o radicale n-esimo (con n non nullo) di un numero reale k , scritto come $\sqrt[n]{k}$, è un numero reale b tale che $b^n = k$.

L'equazione:

$$(*) \quad f(x) = x^n - k = 0 \quad k > 0$$

ha per n intero e per $x > 0$ una sola soluzione reale $b = \sqrt[n]{k}$. Scrivere in un commento % la iterazione k-esima del metodo di Newton per (*). La formula di Erone, (matematico alessandrino del II secolo. In realtà il metodo era già noto ai babilonesi) per il calcolo della radice quadrata è la stessa che si ottiene applicando il metodo di Newton a (*) con $n=2$:

$$x_{i+1} = \frac{1}{2} \left[x_i + \frac{k}{x_i} \right]$$

Calcolare con *newton.m* il valore $\sqrt{2}$ scegliendo $x_0 = 2$ richiedendo 5 cifre decimali significative.

7. Minimo di una funzione senza vincoli

In *ex7.m* usare la function *newton()* per trovare il minimo delle seguenti funzioni di due variabili:

$$\min_{(x,y) \in \mathbb{R}^2} f(x,y)$$

$$f_1(x,y) = 10x^2 + y^2 \quad (x^*, y^*) = (0,0) \quad (x_0, y_0) = [1;2]$$

$$f_2(x,y) = (x-2)^4 + (x-2)^2 y^2 + (y+1)^2 \quad (x^*, y^*) = (2,-1) \quad (x_0, y_0) = [1;1]$$

Rappresentare in due figure distinte le superfici corrispondenti alle funzioni date, il punto di minimo trovato e le curve di livello.

Risolvere i precedenti problemi di minimo per confrontare la soluzione ottenuta usando la function di Matlab *fminsearch()* mediante il seguente comando

$$x = \mathbf{fminsearch}(\text{fun}, x_0)$$

dove *fun* è il nome della function che contiene la funzione e x_0 il punto iniziale.