

Elaborato Calcolo numerico 2020
Autori: Emanuele Brizzi, Massimo Hong

August 16, 2020

Contents

1	Esercizio1	3
2	Esercizio2	3
3	Esercizio3	4
4	Esercizio 4	4
5	Esercizio 6	5
6	Esercizio 7	6
7	Esercizio 10	7
8	Esercizio 13	8
9	Esercizio 14	9

1 Esercizio1

Verificare che, per h sufficientemente piccolo, si ha

$$\frac{f(x-h) - 2f(x) + f(x+h)}{h^2} = O(h^2)$$

Per la dimostrazione utilizziamo il polinomio di Taylor di grado n centrato in x_0 :

$$P_n(x; x_0) = \sum_{k=0}^n \frac{(x-x_0)^k}{k!} f^{(k)}(x_0)$$
$$R_n(x; x_0) = O(x-x_0)^{(n+1)}$$

Per cui possiamo calcolare $f(x+h)$ e $f(x-h)$ sviluppando il polinomio di Taylor centrato in x :

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{6}f'''(x) + O(h^4)$$
$$f(x-h) = f(x) - hf'(x) + \frac{h^2}{2}f''(x) - \frac{h^3}{6}f'''(x) + O(h^4)$$

Sostituendo all'equazione iniziale otteniamo:

$$\frac{f(x-h) - 2f(x) + f(x+h)}{h^2} = \frac{(h^2)f''(x) + O(h^4)}{h^2} = f''(x) + O(h^2)$$

2 Esercizio2

Spiegare cosa calcola il seguente script Matlab: `u = 1; while 1, if 1+u==1, break, end, u = u/2; end, u`

Teoricamente questo script non dovrebbe mai terminare poichè dividendo per 2 il valore u , non si dovrebbe mai raggiungere lo 0, tuttavia il seguente script termina. Più precisamente, alla fine dell'esecuzione risulta che $u = 1.1102 \cdot 10^{-16}$, che è minore del valore $d = eps = 2.2204 \cdot 10^{-16}$ (che rappresenta la distanza da 1.0 al valore in doppia precisione immediatamente successivo) e quando andiamo a sommare ad un numero n un valore u , non verrà effettuata alcuna modifica sul valore.

3 Esercizio3

1. $a = 1e20$; $b = 100$; $a-a+b$
2. $a = 1e20$; $b = 100$; $a+b-a$

Spiegare i risultati ottenuti.

1. $a = 1e20$; $b = 100$; $a-a+b$
Questo script restituisce il valore 100, in quanto $a - a = 0$ e $0 + 100 = 100$.
2. $a = 1e20$; $b = 100$; $a+b-a$ Matlab ha il valore $eps = 2.2204 \cdot 10^{-16}$, che corrisponde a circa 15-16 cifre di precisione (che $1e20$ supera facilmente), per cui quando andiamo ad effettuare $a + b = c$, otterremo un valore approssimato c con le prime 15-16 cifre equivalenti ad a , e di conseguenza $c - a = 0$.

4 Esercizio 4

A quanto pare si deve usare la formula di newton (Modificandola un po').

5 Esercizio 6

Tolleranza	Bisezione	Newton	Secanti
10^{-3}	7.392578125000000e-01	7.390851333852840e-01	7.390851121274639e-01
10^{-6}	7.390851974487305e-01	7.390851332151607e-01	7.390851332150012e-01
10^{-9}	7.390851331874728e-01	7.390851332151607e-01	7.390851332151607e-01
10^{-12}	7.390851332156672e-01	7.390851332151607e-01	7.390851332151607e-01

Tolleranza	Corde
10^{-3}	7.395672022122561e-01
10^{-6}	7.390845495752126e-01
10^{-9}	7.390851327392538e-01
10^{-12}	7.390851332157368e-01

6 Esercizio 7

La molteplicità m di $f(x) = x^2 \tan(x)$ è $m=3$. Sostituendo a x il valore zero otteniamo: $(0)^2 * \tan(0)$; 0 annulla due volte il primo termine del prodotto mentre annulla una volta il secondo termine, in quanto $\tan(0)=0$;

Tolleranza	Newton	Newton Modificato	Aitken
10^{-3}	1.994002961956096e-03	6.617444900424221e-24	-1.570796335324655e+00
10^{-6}	1.349222209381150e-06	6.617444900424221e-24	-1.570796356741072e+00
10^{-9}	1.369405530548002e-09	0	-1.570796314458764e+00
10^{-12}	1.389890778595252e-12	0	Il metodo non converge

7 Esercizio 10

Iterazione	Sigma	Norma
1	$0.1000 = 10^{-1}$	8.9839e-15
2	10	1.4865e-14
3	$1000 = 10^3$	1.3712e-12
4	$100000 = 10^5$	1.2948e-10
5	$10000000 = 10^7$	5.3084e-09
6	10^9	1.0058e-06
7	10^{11}	8.5643e-05
8	10^{13}	0.0107
9	10^{15}	0.9814
10	10^{17}	4.1004e+03

8 Esercizio 13

Eseguendo lo script:

```
A = [1 2 3 ; 1 2 4 ; 3 4 5 ; 3 4 6 ; 5 6 7]; b=[14 17 26 29 38];
```

```
QR = myqr(A); disp(QR); sol = qrsolve(QR,b); disp(sol);
```

Otteniamo:

La Matrice QR:

$$\begin{bmatrix} -6.7082 & -8.6461 & -11.1803 \\ 0.1297 & -1.1155 & -2.9881 \\ 0.3892 & -0.0827 & 1.0351 \\ 0.3892 & -0.0827 & -0.8037 \\ 0.6487 & -0.5222 & -0.4346 \end{bmatrix}$$

Soluzione del sistema lineare:

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

9 Esercizio 14

Dati i seguenti comandi:

```
A = rot90(vander(1:10)); A = A(:,1:8); x = (1:8)'; b = A*x;
```

Le operazioni:

- $A \setminus b$: da come risultato il vettore x , che rappresenta la soluzione del sistema lineare $Ax = b$; Questa operazione dà lo stesso risultato di $\text{inv}(A)*b$ se la matrice A ha rango massimo ed è non singolare.

$$\begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{bmatrix}$$

- $(A' * A) \setminus (A' * b)$: poichè stiamo lavorando con una matrice mal condizionata, il vettore x risultante presenta un errore di approssimazione;

$$\begin{bmatrix} 3.5759 \\ -3.4624 \\ 9.5151 \\ -1.2974 \\ 7.9574 \\ 4.9125 \\ 7.2378 \\ 7.9765 \end{bmatrix}$$

Questo è il warning che segnala Matlab :

Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND = 2.393980e-19.