

# Metodi numerici per la risoluzione di equazioni non lineari

Autore: Claudio Marsan

Ultima revisione: 9 novembre 2003

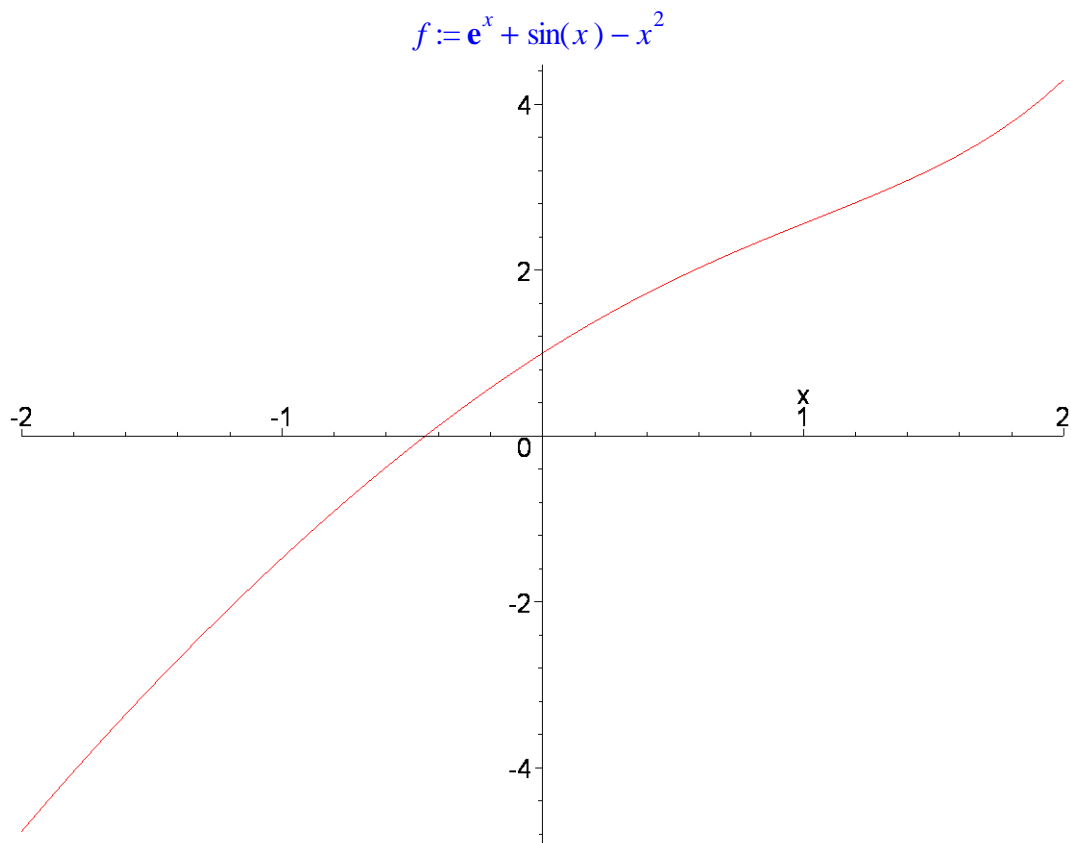
Versione: Maple V Release 6.02 for Windows 2000

```
> restart: Digits := 20:  
> read "equazioni_non_lineari.mpl6":
```

## Esempio 1

La funzione della quale cerchiamo uno zero e il suo grafico:

```
> f := exp(x) + sin(x) - x^2;  
> plot(f(x), x=-2..2);
```



Lo zero della funzione trovato mediante il comando "fsolve":

```
> fsolve(f(x) = 0, x);  
-0.45006669828891971032
```

Lo zero della funzione trovato mediante il metodo di bisezione (1a colonna: passi; 2a colonna: approssimazione per lo zero; 3a colonna: errore):

```
> metodo_di_bisezione(f, -1.0, 0.0, 10, 1e-10);  
1, -.50, -.12289487889156957667
```

```

2, -.250, .46889682381688193865
3, -.3750, .18039174970492463718
4, -.43750, .3056601922395402699e-1
5, -.468750, -.4571402438709265490e-1
6, -.4531250, -.746089467170315531e-2
7, -.44531250, .1158090196599418587e-1
8, -.449218750, .206708075988030521e-2
9, -.4511718750, -.269513865657840163e-2
10, -.45019531250, -.31358675104408663e-3

```

Lo zero della funzione trovato mediante il metodo di Newton con valore iniziale  $x_0 = -0.43750$  (1a colonna: passi; 2a colonna: approssimazione per lo zero; 3a colonna: errore):

```

> metodo_di_Newton(f, -0.43750, 10, 1e-10);
1, -.45009694838840920682, -.7375431445153680e-4
2, -.45006669846294800091, -.42430483346e-9
3, -.45006669828891971032, -.1e-19

```

Lo zero della funzione trovato mediante il metodo di Newton con valore iniziale  $x_0 = 2$  (1a colonna: passi; 2a colonna: approssimazione per lo zero; 3a colonna: errore):

```

> metodo_di_Newton(f, 2, 10, 1e-10);
1, .5541591934393049640, 1.9596130654160440735
2, -.7676692847144076390, -.81968275615885407026
3, -.46620042396996573095, -.3945674908852486509e-1
4, -.45011575068695539904, -.11959758768310881e-3
5, -.45006669874651556148, -.111568142611e-8
6, -.45006669828891971036, -.10e-18

```

Lo zero della funzione trovato mediante il metodo della "regula falsi" (1a colonna: passi; 2a colonna: approssimazione per lo zero; 3a colonna: errore):

```

> regula_falsi(f, -1.0, 0.0, 10, 1e-10);
1, -.40427046355838077491, .11068082769056720685
2, -.44588946756704566531, .1017656692645608521e-1
3, -.44968988818386218337, .91864886010160794e-3
4, -.45003274221057211693, .8278904007426060e-4
5, -.45006363863072077013, .745986185727241e-5
6, -.45006642259655975800, .67217573844554e-6
7, -.45006667344751059817, .6056676113001e-7
8, -.45006669605057126419, .545740040885e-8
9, -.45006669808723212836, .49174197813e-9
10, -.45006669827074654306, .4430867350e-10

```

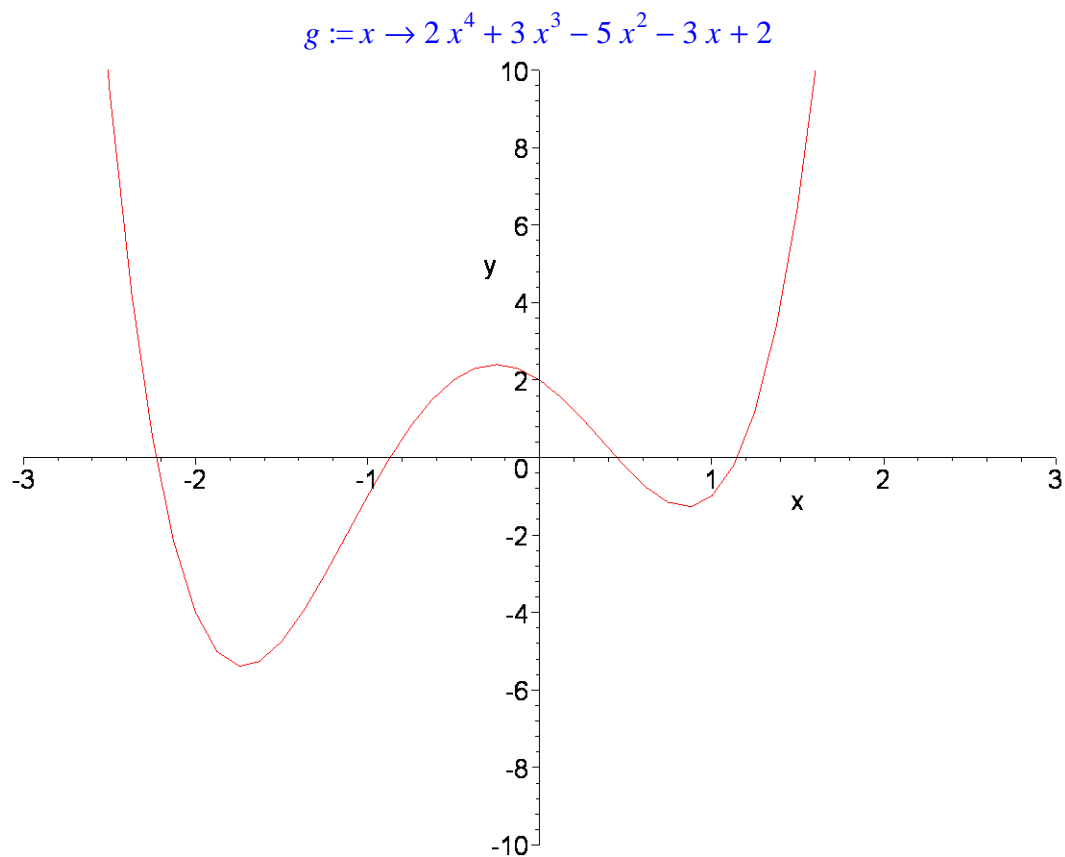
## Esempio 2

La funzione della quale cerchiamo uno zero e il suo grafico:

```

> g := x -> 2*x^4 + 3*x^3 - 5*x^2 - 3*x + 2;
> plot(g(x), x=-3..3, y=-10..10);

```



Lo zero della funzione trovato mediante il comando "fsolve":

```
> fsolve(g(x) = 0, x);
-2.2293397155437320841, -.86941813842887187211, .44856330913931694662,
1.1501945448332870096
```

Lo zero della funzione trovato mediante il metodo di bisezione (1a colonna: passi; 2a colonna: approssimazione per lo zero; 3a colonna: errore):

```
> metodo_di_bisezione(g(x), -0.5, 0.5, 10, 1.0e-15);
1, 0., 2.
2, .25, .99218750
3, .375, .369628906250
4, .4375, .549621582031250e-1
5, .46875, -.993328094482421875e-1
6, .453125, -.225619077682495117e-1
7, .4453125, .161154344677925110e-1
8, .44921875, -.32455842010676861e-2
9, .447265625, .64294855401385576e-2
10, .4482421875, .15905723867035704e-2
```

Lo zero della funzione trovato mediante il metodo della "regula falsi" (1a colonna: passi; 2a colonna: approssimazione per lo zero; 3a colonna: errore):

```
> regula_falsi(g(x), -0.5, 0.5, 10, 1.0e-15);
1, .38888888888888888889, .2993446121018137479
2, .44943469515155718943, -.43146171866283332e-2
```

```
3, .44857441510378351530, -.550044617680825e-4
4, .44856344992526232681, -.6972721785382e-6
5, .44856331092388206747, -.88384366556e-8
6, .44856330916193759872, -.1120335696e-9
7, .44856330913960367974, -.14201064e-11
8, .44856330913932058119, -.180010e-13
9, .44856330913931699269, -.2282e-15
```

Lo zero della funzione trovato mediante il metodo delle secanti (1a colonna: passi; 2a colonna: approssimazione per lo zero; 3a colonna: errore):

```
> metodo_delle_secanti(g(x), -0.5, 0.5, 10, 1.0e-15);
1, .38888888888888888889, .2993446121018137479
2, .44943469515155718943, -.43146171866283332e-2
3, .44857441510378351530, -.550044617680825e-4
4, .44856330629083405551, .141077147244e-7
5, .44856330913932621899, -.459234e-13
6, .44856330913931694662, 0.
```

Lo zero della funzione trovato mediante il metodo di Newton con valore iniziale  $x_0=0$  (1a colonna: passi; 2a colonna: approssimazione per lo zero; 3a colonna: errore):

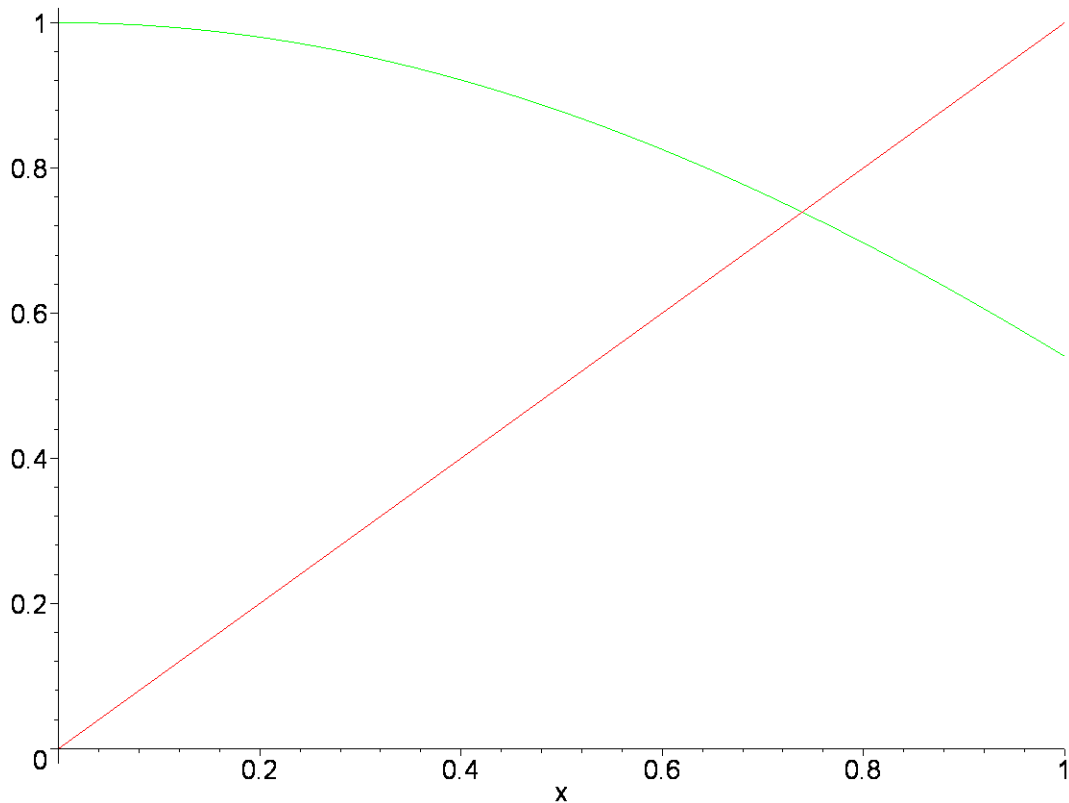
```
> metodo_di_Newton(g(x), 0, 10, 1.0e-15);
1, .66666666666666666667, -.9382716049382716049
2, .38202247191011235957, .3340828071474208260
3, .44803719542262017276, .26060904229569794e-2
4, .44856322842610962279, .3997492605464e-6
5, .44856330913931503726, .94565e-14
6, .44856330913931694662, 0.
```

### Esempio 3

Vogliamo risolvere l'equazione non lineare  $\cos(x) = x$  nell'intervallo  $[0,1]$

Risoluzione grafica:

```
> plot({x, cos(x)}, x=0..1);
```



La soluzione trovata mediante il comando "fsolve":

```
> fsolve(cos(x) = x, x);
                .73908513321516064166
```

La soluzione trovata mediante il metodo di iterazione con valore iniziale  $x_0=0.5$  (1a colonna: passi; 2a colonna: approssimazione  $z$  per lo zero; 3a colonna:  $\cos(z)$ ):

```
> metodo_di_iterazione(cos(x), 0.5, 25, 1.0e-10);
1, .87758256189037271612, .63901249416525922865
2, .63901249416525922865, .80268510068233485563
3, .80268510068233485563, .69477802678800619881
4, .69477802678800619881, .76819583128201604852
5, .76819583128201604852, .71916544594241901794
6, .71916544594241901794, .75235575942152703886
7, .75235575942152703886, .73008106313782326118
8, .73008106313782326118, .74512034135144008667
9, .74512034135144008667, .73500630901484315561
10, .73500630901484315561, .74182652264324587170
11, .74182652264324587170, .73723572544223138691
12, .73723572544223138691, .74032965187826312029
13, .74032965187826312029, .73824623833223344393
14, .73824623833223344393, .73964996276966128447
15, .73964996276966128447, .73870453935698324133
16, .73870453935698324133, .73934145228121009242
17, .73934145228121009242, .73891244933210305379
18, .73891244933210305379, .73920144413579906725
19, .73920144413579906725, .73900677978081295033
20, .73900677978081295033, .73913791076229291356
```

21, .73913791076229291356, .73904958059520852278  
22, .73904958059520852278, .73910908142052666307  
23, .73910908142052666307, .73906900120401150733  
24, .73906900120401150733, .73909599983575472825  
25, .73909599983575472825, .73907781328517518028