Matematica Applicata 2012
- Stefano Vena -

Lavorare con funzioni matematiche

Funzioni

Le funzioni si definiscono con delle espressioni testuali:

```
fun ='1./(1+x.^2)';
```

Esse vanno valutate su intervalli.

```
xes =[ -5 ,5];
```

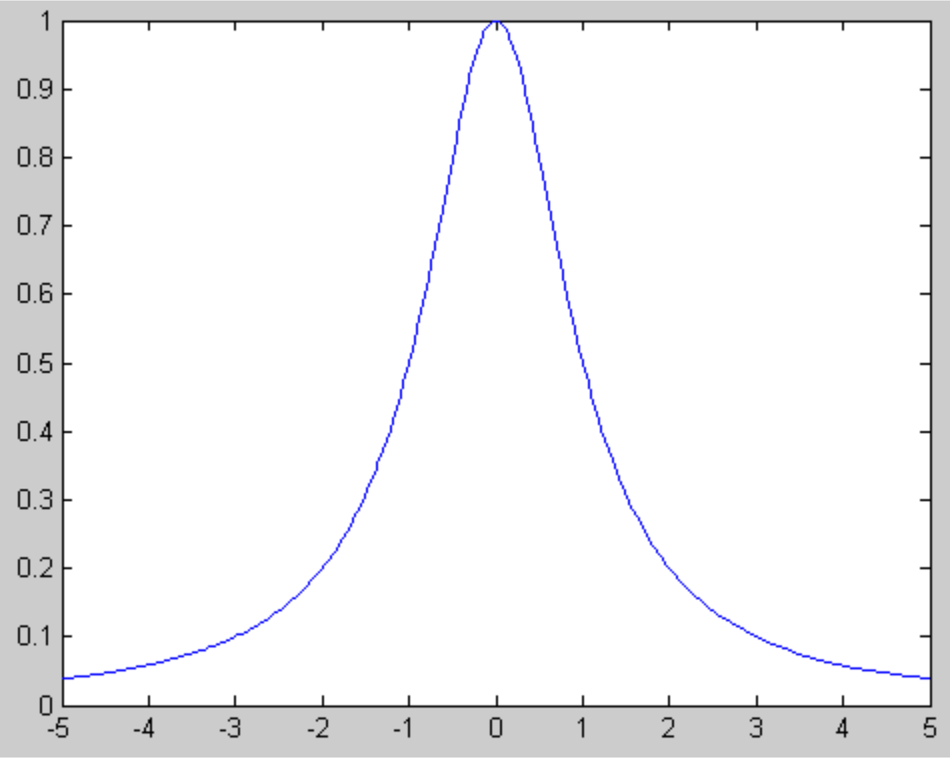
Quindi possiamo disegnarla

```
fplot(fun ,xes);
```

La notazione «element wise» è necessaria poiché la funzione viene valutata impiegando un vettore.

Altro uso: (doc eval)

```
 x=[-5:0.1:5];
 y = eval(fun);
plot(y);
```



Altri modi per inizializzare una funzione

```
>> fplot('x^2 - 1 + exp(x)' ,[-5 5]);

Inserisci griglia

>> grid on
```

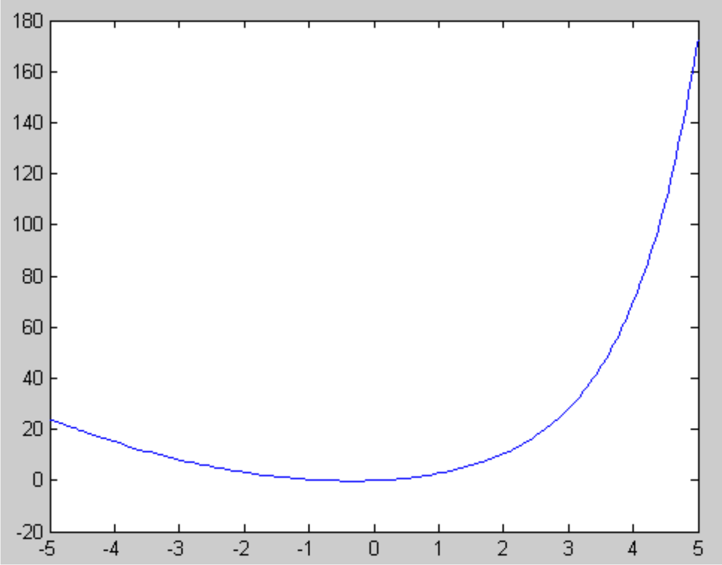
Un modo differente di inizializzare la funzione

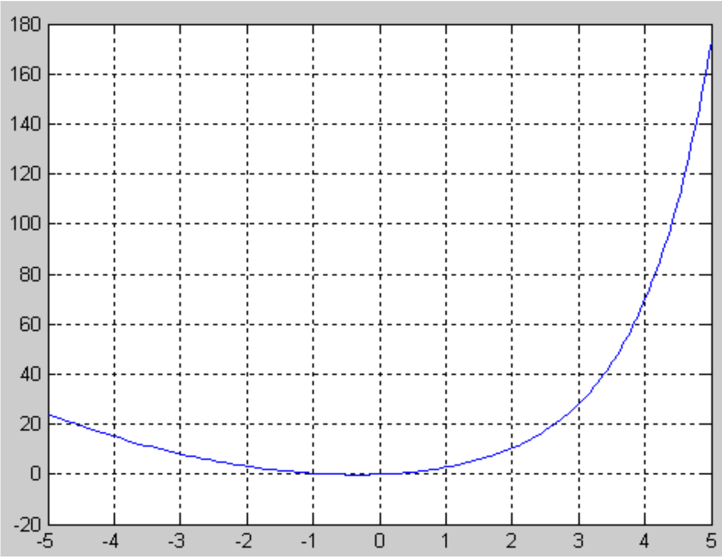
```
>> fun=inline('x^2 - 1 + exp(x)','x')

fun =
     Inline function:
     fun(x) = x^2 - 1 + exp(x)

>> fplot(fun ,[-5 5]);
>> fun(3)

ans =  28.0855
```





Esercizio

- Realizzare uno script Matlab che disegni i grafici delle funzioni

$$f(x) = \begin{cases} (m - \frac{x^2}{m})^m, & x \in [-m, 0], \\ (\frac{x^2}{m} + m)^m, & x \in [0, m], \end{cases}$$

- per m = 1..6; sovrapposti nella stessa finestra oppure tutti in una stessa finestra utilizzando sei sotto finestre della stessa finestra.
- La scelta deve essere fatta dall'utente.
- Si devono usare vettori con 101 elementi.

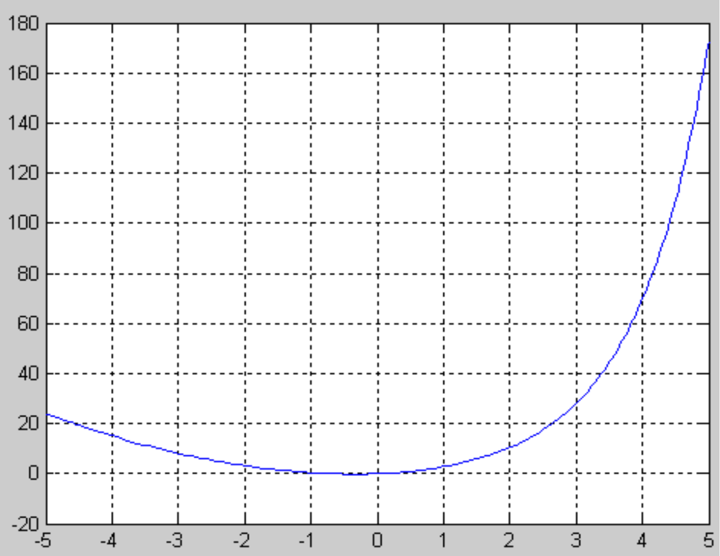
Esercizio

- Data la successione

$$x_{n+1} = \begin{cases} 2x_n + 2, & x_n \text{ dispari}, \\ x_n/2 + 1, & x_n \text{ pari} \end{cases}$$

- realizzare uno script Matlab che realizza il grafico e ne calcola il massimo ed il minimo sui primi 500 valori a partire da un valore iniziale intero x1 inserito in input dall'utente.

Ricerca degli zeri



Osserviamo l' andamento della funzione appena definita:
Vediamo che in prossimità di -1 e 1 ce ne sono due. Allora
fzero(fun ,1)
ans = 5.4422e-018

fzero(fun ,-1)
ans = -0.7146

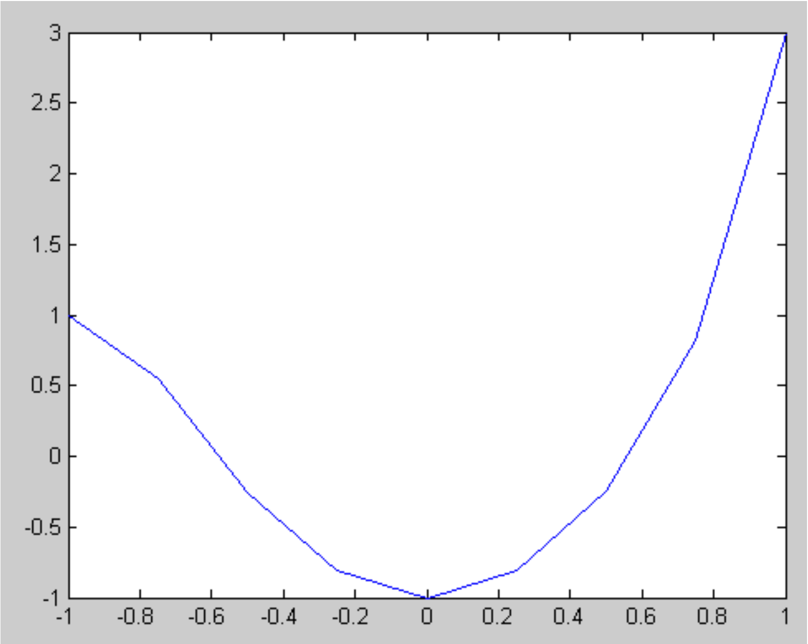
Polinomi

Sia dato il seguente polinomio

$$p(x) = x^7 + 3x^2 - 1$$

Possiamo definirlo come segue:

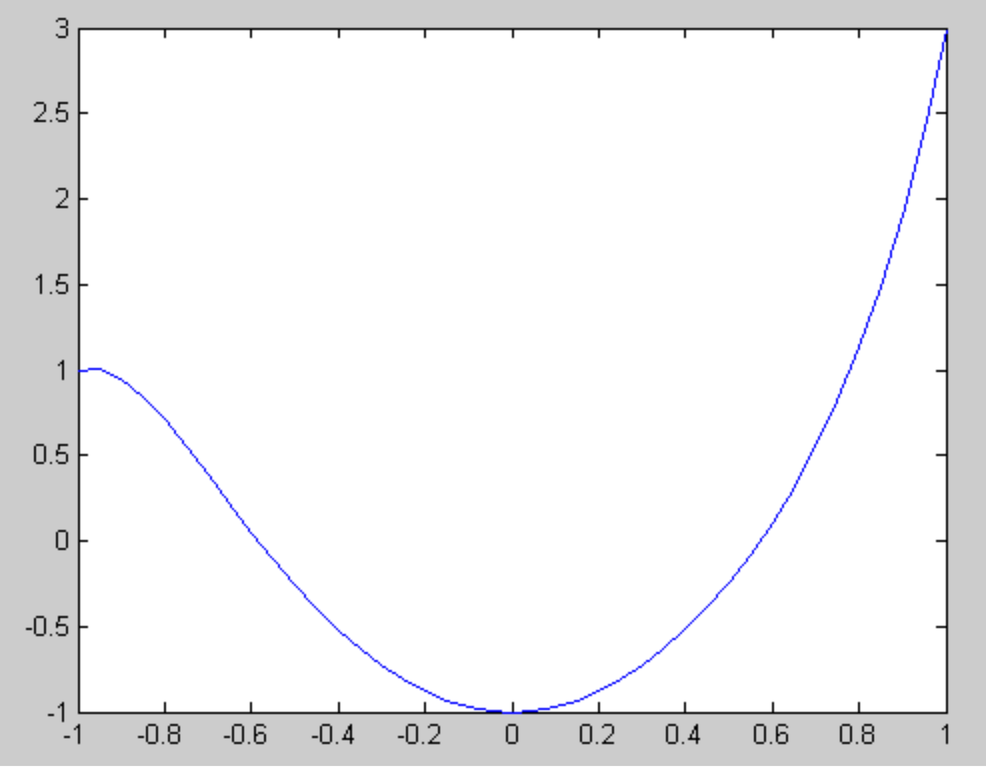
```
p = [1 0 0 0 0 3 0 -1];
x = [ -1:0.25:1];
y = polyval(p,x)
plot(x,y)
```



Polinomi/2

Per migliorare la risoluzione della curva possiamo aumentare la risoluzione dei punti nel vettore x:

```
p = [1 0 0 0 0 3 0 -1];
x = [ -1:0.05:1];
y = polyval(p,x);
 plot(x,y)
```



Radici di un polinomio

Consideriamo un nuovo polinomio

```
p = [1 -6 11 -6];
```

Le radici saranno

```
roots(p)

ans =
   3.00000000000000
   2.00000000000000
   1.00000000000000
```

VERIFICA

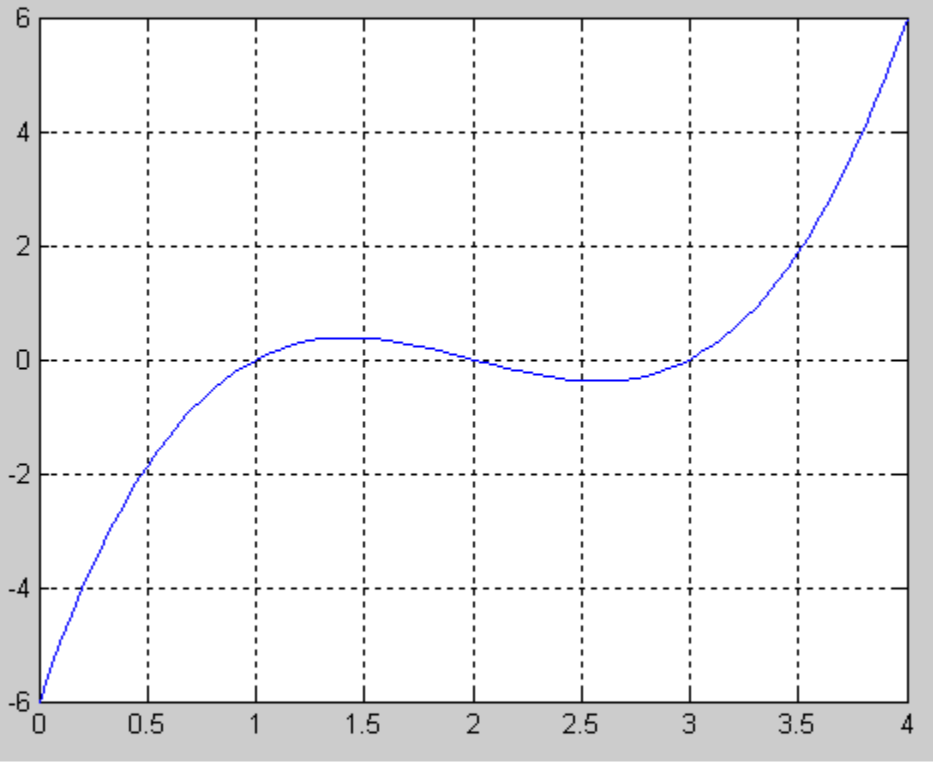
```
polyval(p,3)
ans =  0

polyval(p,2)
ans = 0

polyval(p,1)
ans = 0
```

Verifichiamolo graficamente

```
 x = [ 0:0.05:4];
 y = polyval(p,x);
 plot(x,y)
 grid on
```



Errori

Purtroppo questo metodo è soggetto ad errori

Torniamo infatti al vecchio polinomio

```
p = [1 0 0 0 0 3 0 -1];
x = [ -1:0.05:1];
y = polyval(p,x);
roots(p)

ans =
  -0.3739 + 1.2305i
  -0.3739 - 1.2305i
   0.9698 + 0.7716i
   0.9698 - 0.7716i
  -1.1793
  -0.5840
   0.5716
```

7 radici di cui 4 cc a coppie.

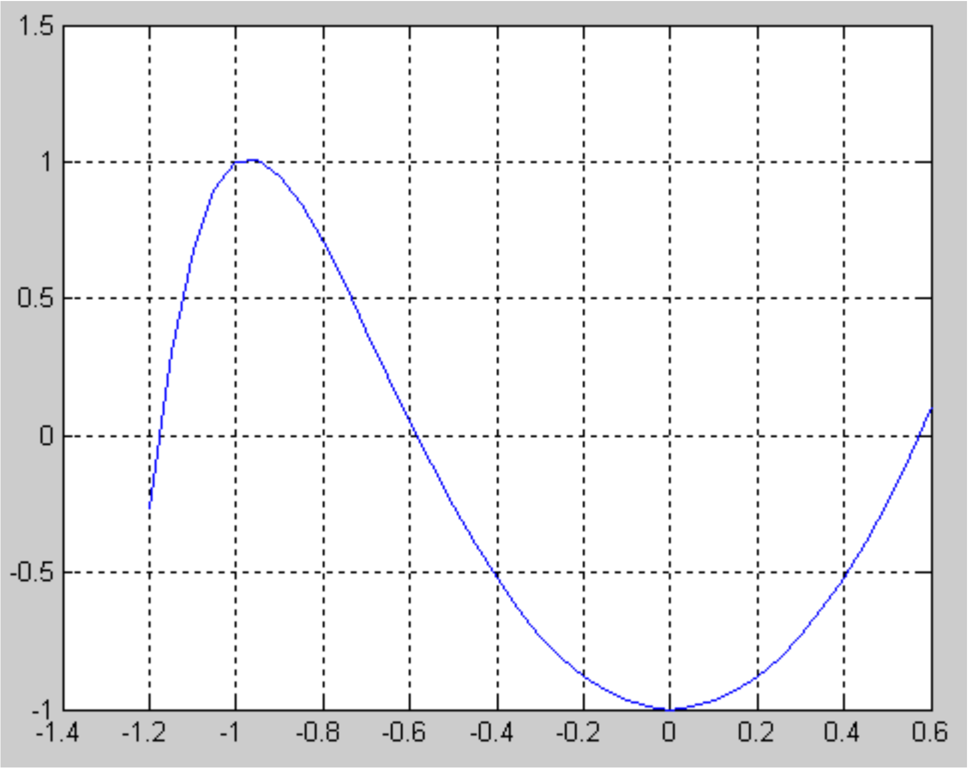
Verifichiamo:

```
polyval(p,-0.3739 +i*1.2305)
ans =
 -5.8083e-004 -8.0563e-004i
```

Le risposte approssimano le radici, ma non le centrano perfettamente.

L'esistenza delle tre radici reali è del resto evidente dal plot

```
p = [1 0 0 0 0 3 0 -1];
x = [ -1.2:0.05:0.6];
plot(x,y)
grid on
```



Come migliorare i risultati

```
p = [1 0 0 0 0 3 0 -1];
x = [ -1.2:0.05:0.6];
format long

roots(p)
ans =
 -0.37392903093853 + 1.23052948304675i
 -0.37392903093853 - 1.23052948304675i
  0.96979380747135 + 0.77159912197128i
  0.96979380747135 - 0.77159912197128i
 -1.17929794679761
 -0.58400002290522
  0.57156841663718

polyval(p,-0.58400002290522)
ans = 1.998401444325282e-015

polyval(p,-0.37392903093853 +i*1.23052948304675)
ans = 4.662936703425658e-015 +7.588374373312945e-014i
```