Funzioni e Operazioni Matriciali

Informatica (ICA) AA 2020 / 2021

Giacomo Boracchi

17 Novembre 2020

giacomo.boracchi@polimi.it

Esempio

Scrivere una funzione *cerca* che controlla se un elemento **x** appartiene ad un vettore **vett** e, in caso affermativo, ne restituisce la posizione

0, i ← ([3 4 5 6], 8)

-1, 2 ← (, 4)

Esempi

```
function [pres, pos] = cerca(x, v)
    p=0; pos=[];
    for i=1:length(v)
        if v(i)==x
            p=p+1;
            pos(p)=i;
        end
    end
    pres=p>0;
```

```
>> A=[1, 2, 3, 4, 3, 4, 5, 4, 5, 6]
A = 1 2 3 4 3 4 5 4 5 6
>> [p, i]=cerca(4,A)
p = 1
i = 4 6 8
```

Esercizio: implementare usando find()

Esempi

```
function [pres, pos] = cerca2(x, v)
pres = 1;
pos = find(v == x);
if isempty(pos)
    pres = 0;
end
```

Esempi

```
function [pres, pos] = cerca2(x, v)
pres = 1;
pos = find(v == x);
if isempty(pos)
    pres = 0;
end
```

```
function [pres, pos] = cerca3(x, v)
pres = any(v == x);
pos = find(v == x);
```

Esercizio

Scrivere una funzione *closestVal* che prende in ingresso un vettore **vett** ed uno scalare **x** e restituisce il valore di **vett** più vicino ad **x**

Esercizio

Scrivere un programma che determina se una parola è palindroma

Esercizio TODO

Scrivere un programma che richiede in ingresso due parole e determina se l'una è l'anagramma dell'altra

- Uno script si occupa dell'acquisizione delle parole.
- Implementare una funzione per creare l'istogramma delle parole.
- Eseguire nello script il confronto tra istogrammi e visualizzare a schermo il risultato.

Funzioni Built in per Visualizzazione

Funzioni Grafiche

Funzione	Scopo
figure(figNumber)	apre una figura identificata dall'handle figNumber . Se non presente definisce l'handle in maniera incrementale
hold	+ on / off definisce se tenere o cancellare il grafico attualmente presente nella figura alla prossima operazione di visualizzazione sulla figura.
plot(x,y)	disegna in un riferimento cartesiano 2D le coppie di punti identificati da (x(1),y(1))… (x(end) , y(end)). x ed y devono avere la stessa lunghezza
plot3(x,y,z)	disegna in un riferimento cartesiano 3D le coppie di punti identificati da (x(1),y(1),z(1))… (x(end) , y(end), z(end). x,y e z devono avere la stessa lunghezza
plot(x,y, frmStr)	frmStr specifica il marker ed il colore usato nella visualizzazione dei punti
imagesc(A)	Visualizza la matrice A come un'immagine in colormap di default. Ogni pixel viene ridimensionato per migliorare la visualizzazione
imshow(A)	visualizza un'immagine A in scale di grigio (se A è di dimensione 2) o a colori nello spazio RGB (se A è di dimensione 3)
legend(titles)	Visualizza la legenda, usando le stringhe in titles

Plot

Diagrammi a due dimensioni

La funzione `plot(x,y)` disegna il **diagramma** cartesiano dei punti che hanno valori delle ascisse nel vettore x, delle ordinate nel vettore y

Il diagramma è l'insieme di **coppie di punti** [x(1), y(1)],…, [x(end),y(end)] rappresentanti le coordinate dei punti del piano cartesiano

- La funzione `plot` congiunge i punti con una linea, per dare continuità al grafico.

In `plot(x,y)`, `x` e `y` devono essere **due vettori aventi le stesse dimensioni**

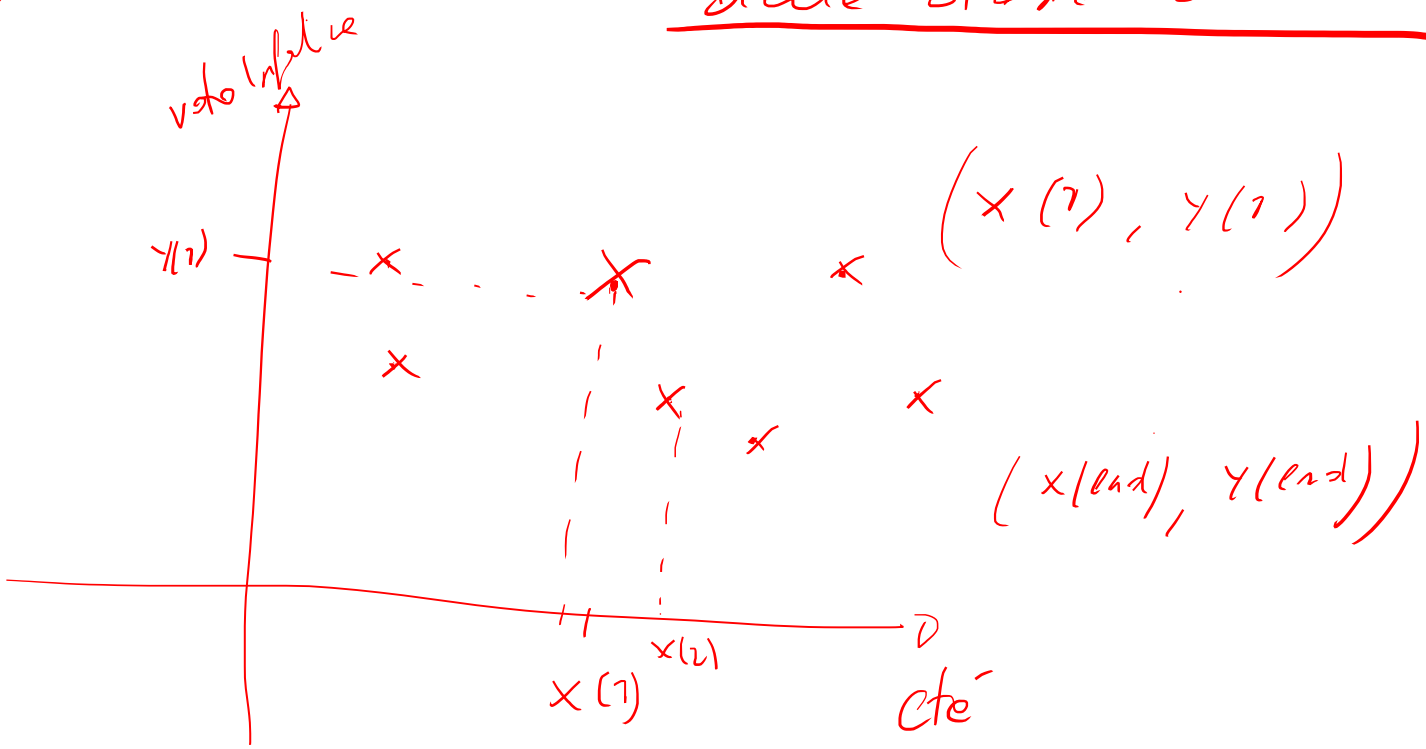
E' possibile specificare diversi elementi grafici (help plot per una lista delle opzioni)

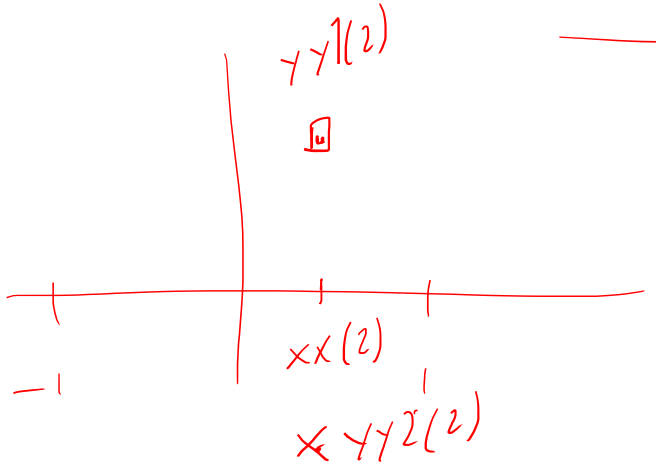
Le funzioni `xlabel` visualizzano una stringa come nome asse ascisse, `ylabel` per ordinate, `title` per il titolo

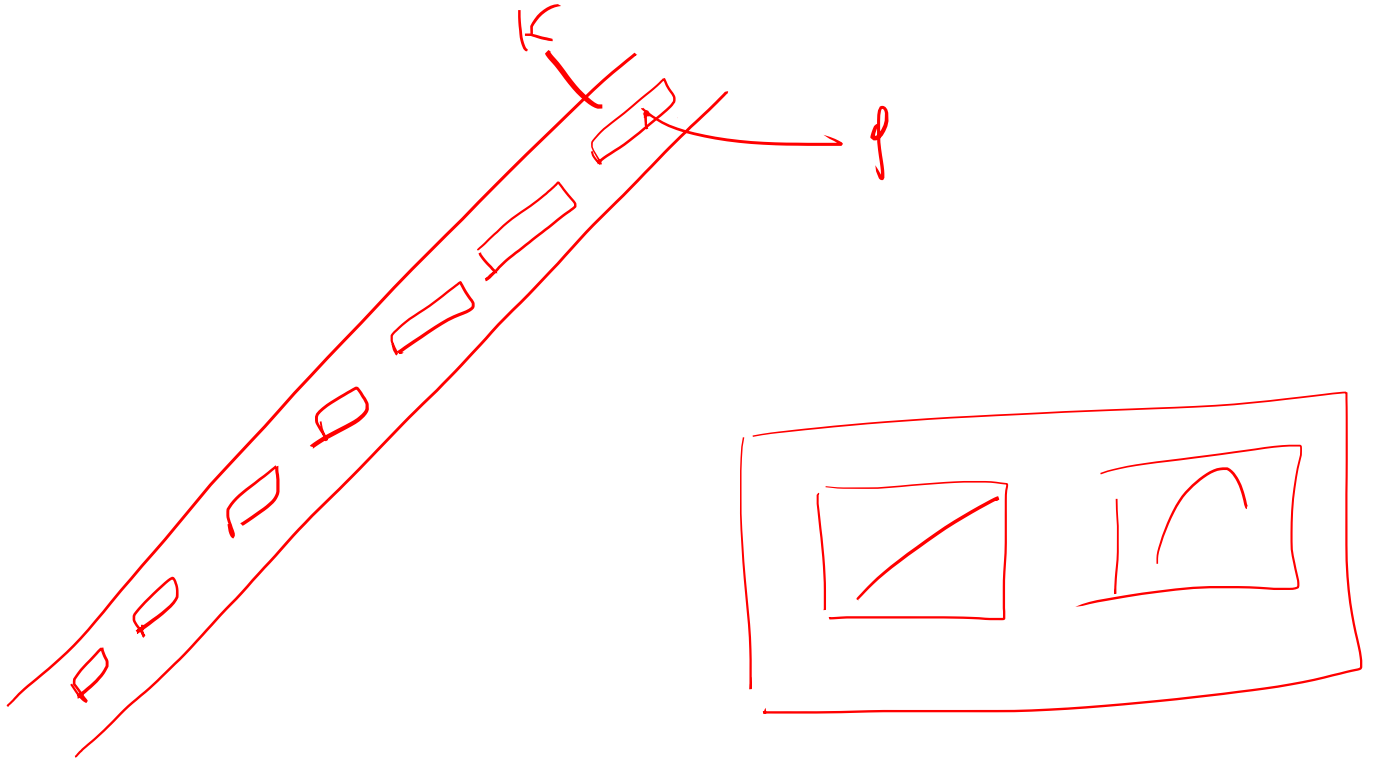
plot(x, y)

x, y devono essere vettori delle stessa dimensione

plot(x, y, 'x')






F
φ

Esercizio

Scrivere una funzione che prende in ingresso due coefficienti m, q ed un vettore di punti xx e restituisce il vettore yy dei punti che stanno sulla retta $y = mx + q$ in corrispondenza a xx

Si invochi la funzione e si disegni la retta nell'intervallo xx

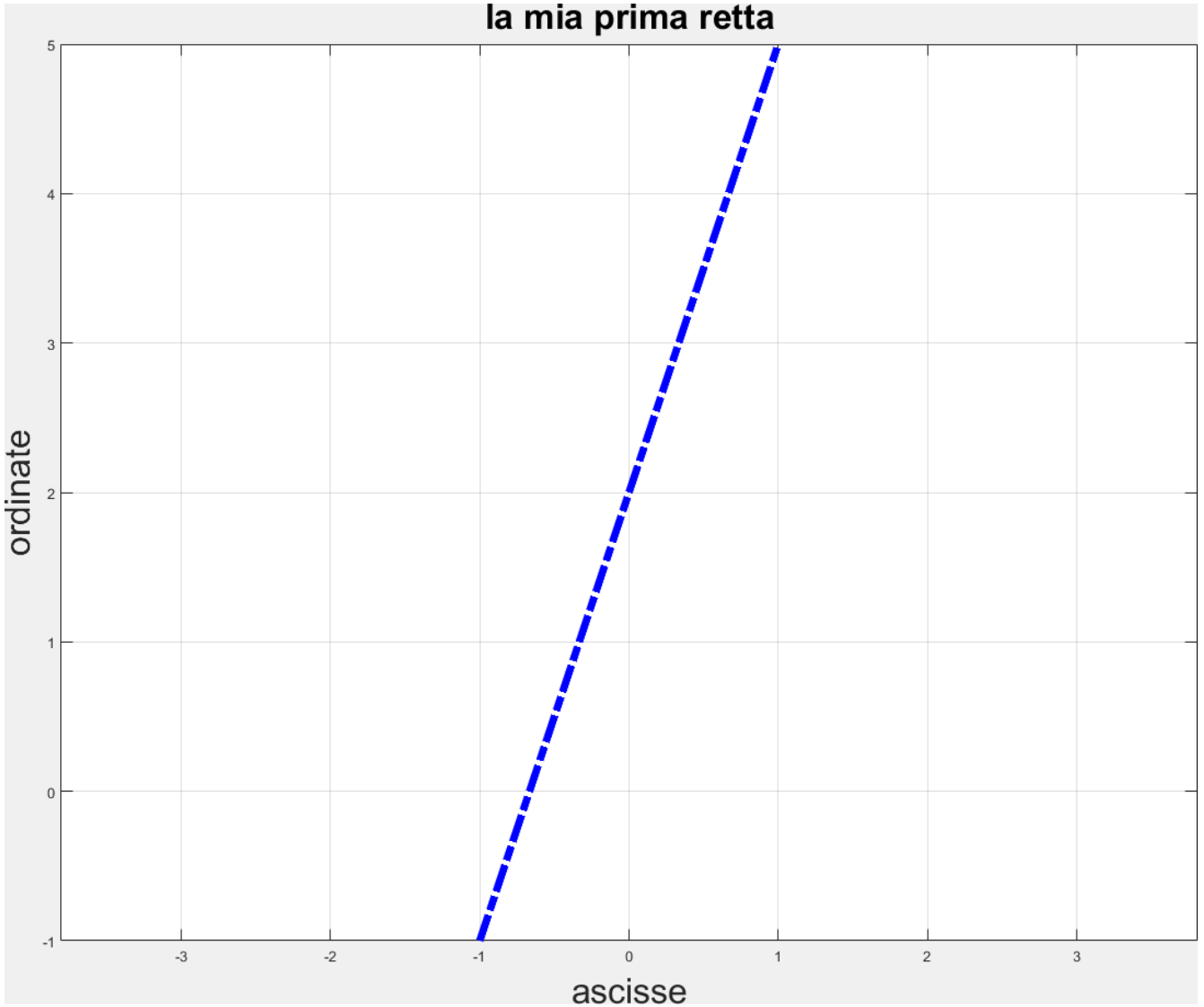
Esercizio

Scrivere una funzione che prende in ingresso due coefficienti m, q ed un vettore di punti xx e restituisce il vettore yy dei punti che stanno sulla retta $y = mx + q$ in corrispondenza a xx

```
function [yy] = retta(m, q, xx)
    yy = m * xx + q;

% for ii = 1 : length(xx)
%     yy(ii) = m * xx(ii) + q;
% end
```

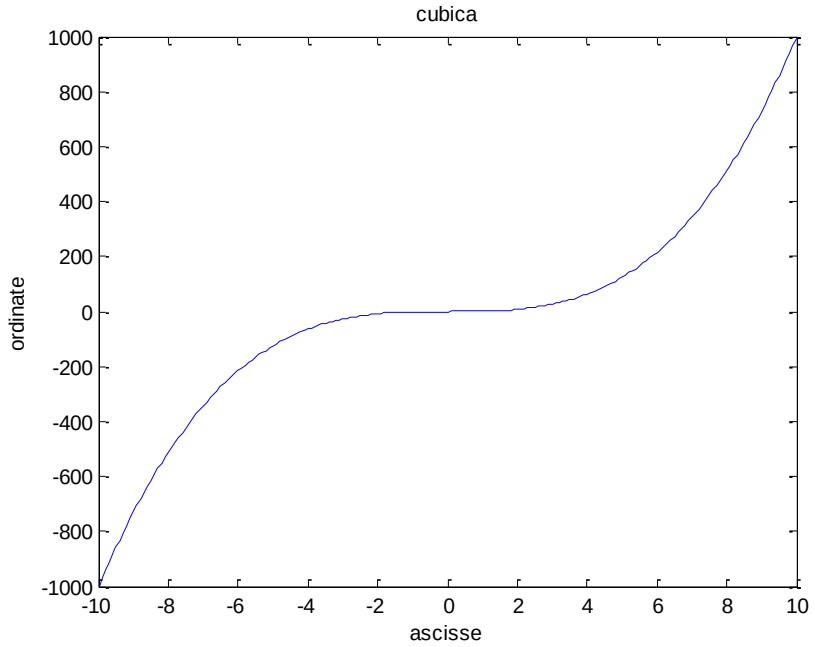
Esempi di script per invocare la funzione

```
x = [-1 : 0.1 : 1];
% invoco la funzione per plottare y = 3x +2
y = retta(3,2,x)
figure
plot(x,y, 'b*') % disegno con le stelline
axis equal % assi della stessa dimensione
plot(x,y, 'b-'), %disegno con una retta
grid on % aggingo aggiungo la griglia
plot(x,y, 'b-', 'LineWidth', 3), axis equal, grid on
plot(x,y, 'b--', 'LineWidth', 5), axis equal, grid on
plot(x,y, 'b-.', 'LineWidth', 5), axis equal, grid on
title('la mia prima retta', 'FontSize', 24)
xlabel('ascisse', 'FontSize', 24)
ylabel('ordinate', 'FontSize', 24)
```

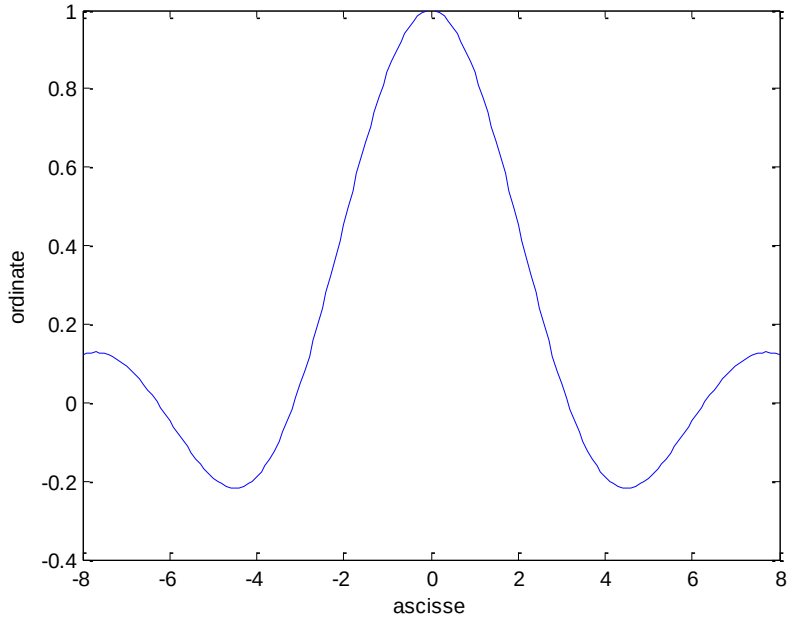

la mia prima retta
ordinate
ascisse
5
4
3
2
1
0
-1
-3
-2
-1
0
1
2
3

Diagrammi a due dimensioni: esempi

```
>> x = -10:0.1:10;
>> y=x.^3;
>> plot(x,y);
>> xlabel('ascisse');
>> ylabel('ordinate');
>> title('cubica');
```




```
>> x=[-8:0.1:8];
>> y= sin (x) ./ x;
>> plot(x, y);
>> xlabel('ascisse');
>> ylabel('ordinate');
```



Esempi

Definire una funzione *samplePolynomial* che prende in ingresso

- un vettore di coefficienti C
- un vettore che definisce un intervallo [a,b]

e restituisce un due vettori di 100 punti xx ed yy contenente i punti della del polinomio

$$y = C(1)x^{n-1} + C(2)x^{n-2} + \ldots + C(n-1)x^1 + C(n)$$

Utilizzare *samplePolynomial* per calcolare i punti delle seguenti curve (in un intervallo [-10 10]) e visualizzarlo:

$$y = x - 1;$$
$$y = 2x^2 + x - 12;$$
$$y = -0.1x^3 + 2x^2 - 10x - 12$$

visualizzare, per ogni valore di x, la curva maggiore

```
interval = [-10 , 10];
rettaCoeffs = [1 ,-1];
parabolaCoeffs = [ 2 ,  1 , -12] ;
cubicaCoeffs = [-0.1 , 2 ,-10 , -12];

% calcola i valori dei polinomi
[rx,ry] = samplePolynomial(rettaCoeffs , interval);
[px,py] = samplePolynomial(parabolaCoeffs , interval);
[cx,cy] = samplePolynomial(cubicaCoeffs, interval);
```

Disegno le tre curve

```
figure(1), plot(rx, ry, 'r-', 'LineWidth', 3)
hold on
plot(px, py, 'b--', 'LineWidth', 3)
plot(cx, cy, 'm:', 'LineWidth', 3)
hold off
legend('retta', 'parabola', 'cubica')
xlabel('x')
ylabel('y')
```

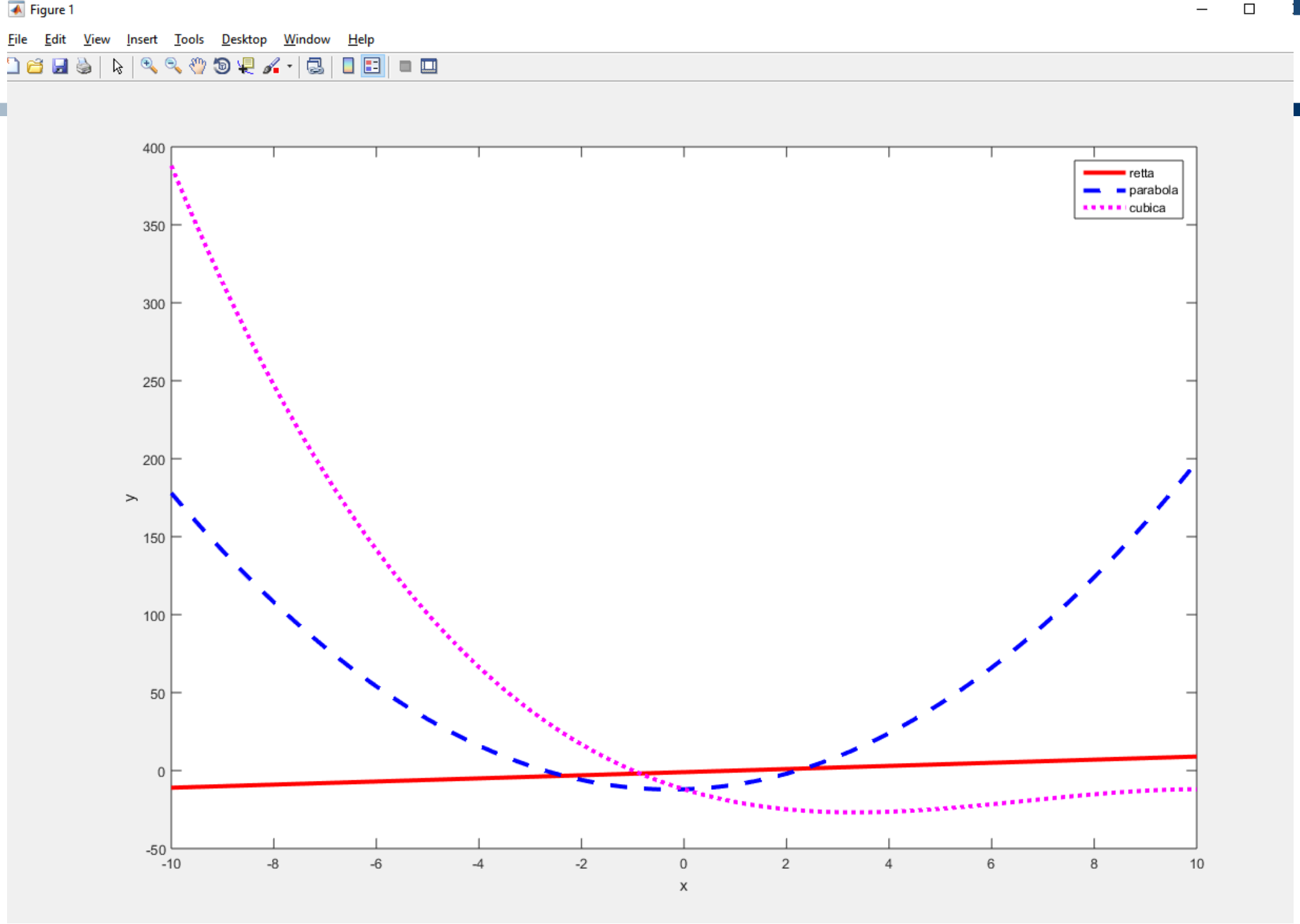


Figure 1
File Edit View Insert Tools Desktop Window Help
retta
parabola
cubica
400
350
300
250
200
150
100
50
0
-50
y
-10 -8 -6 -4 -2 0 2 4 6 8 10
x

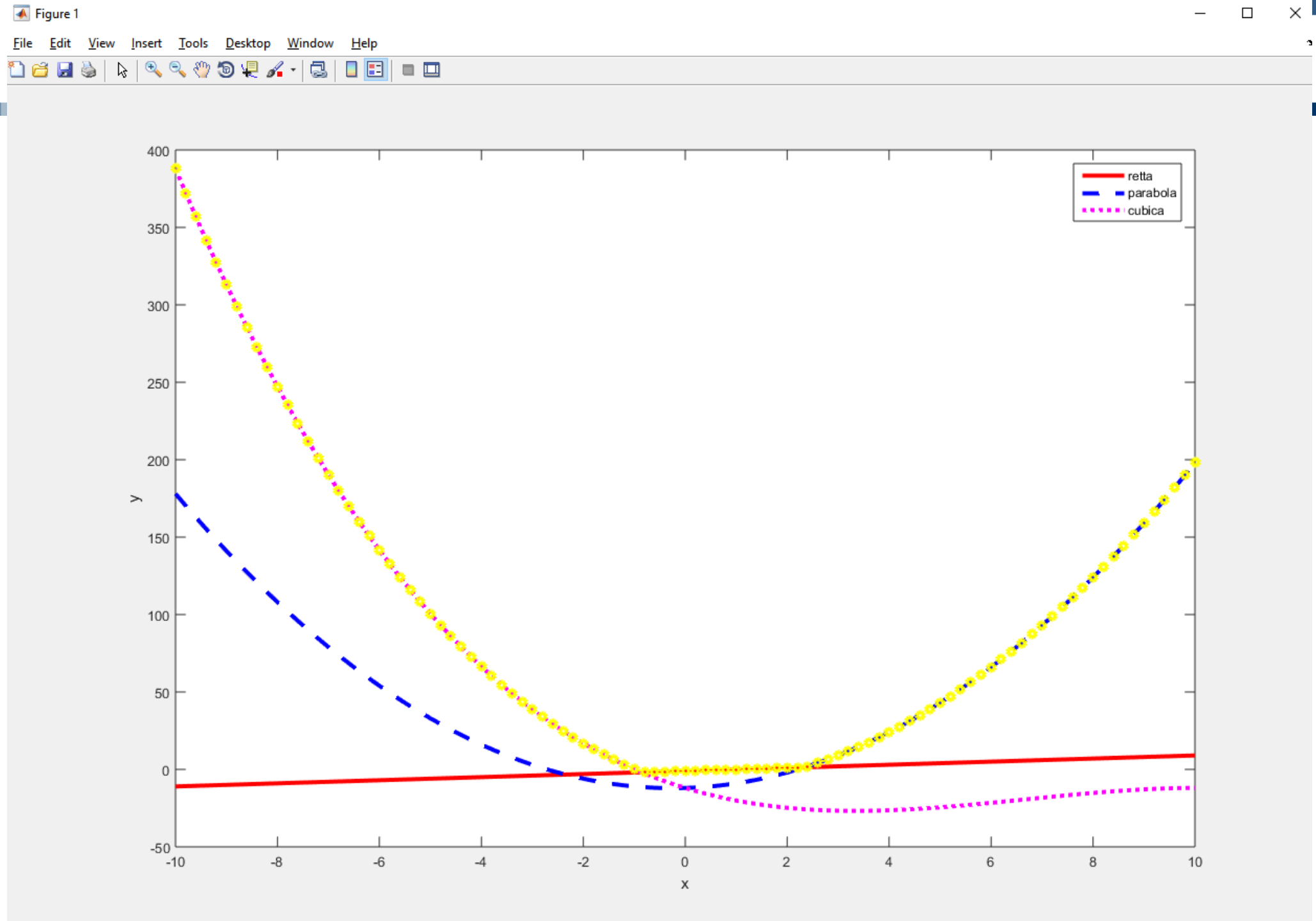


Figure 1
File Edit View Insert Tools Desktop Window Help
retta
parabola
cubica
y
x
400
350
300
250
200
150
100
50
0
-50
-10
-8
-6
-4
-2
0
2
4
6
8
10

Diagrammi a due dimensioni: ancora esempi

Un diagramma è semplicemente una sequenza ordinata di punti, di coppie di coordinate cartesiane

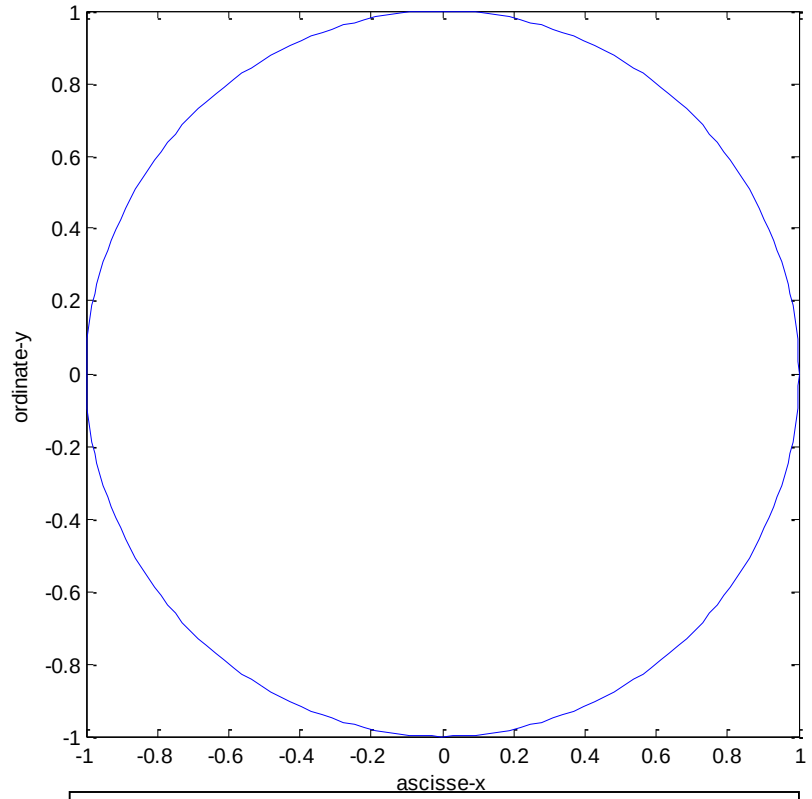
In `plot(x,y)` non necessariamente `x` contiene valori equispaziati e `y` non è necessariamente funzione di `x`. Sia `x` che `y` possono essere, ad esempio, funzioni di qualche altro parametro.

Che diagrammi disegnano i seguenti esempi?

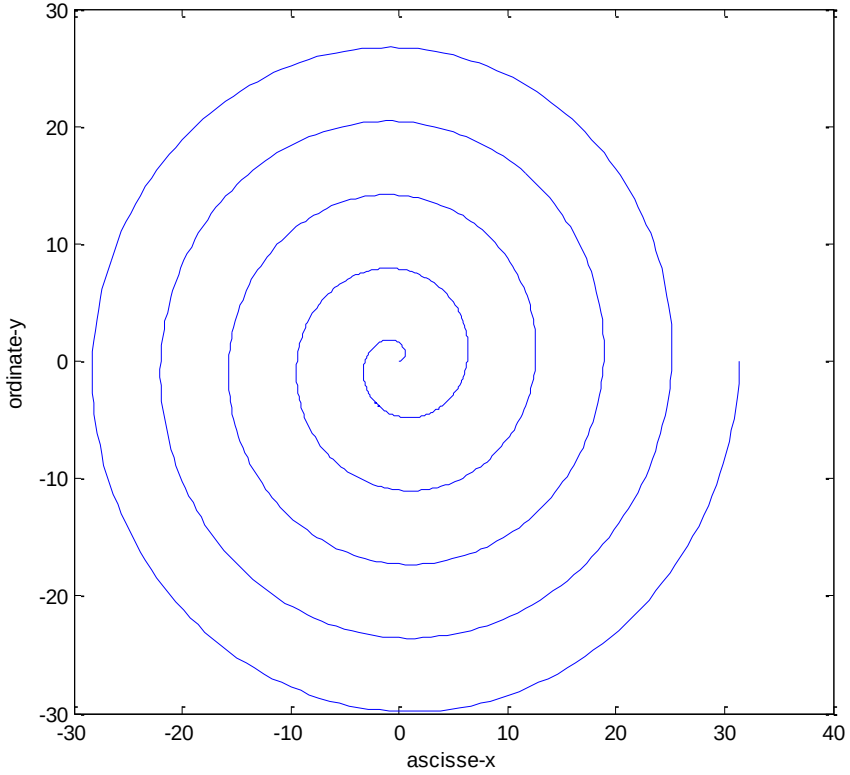
```
>> t=[0:pi/100:2*pi];
>> x=cos(t);
>> y=sin(t);
>> plot(x,y);
>> xlabel('ascisse-x');
>> ylabel('ordinate-y');
```

```
>> t=[0:pi/100:10*pi];
>> x=t .* cos(t);
>> y=t .* sin(t);
>> plot(x,y);
>> xlabel('ascisse-x');
>> ylabel('ordinate-y');
```

Esempi




```
>> t=[0:pi/100:2*pi];
>> x=cos(t);
>> y=sin(t);
>> plot(x,y);
>> xlabel('ascisse-x');
>> ylabel('ordinate-y');
```




```
>> t=[0:pi/100:10*pi];
>> x=t .* cos(t);
>> y=t .* sin(t);
>> plot(x,y);
>> xlabel('ascisse-x');
>> ylabel('ordinate-y');
```

Esempi

Definire una funzione *samplePolynomial* che prende in ingresso

- un vettore di coefficienti C
- un vettore che definisce un intervallo [a,b]

e restituisce due vettori di 100 punti xx ed yy contenenti i punti che stanno sulla curva (e le cui ascisse stanno in [a,b])

$$y = C(1)x^{n-1} + C(2)x^{n-2} + \ldots + C(n-1)x^1 + C(n)$$

Utilizzare samplePolynomial per calcolare i punti delle seguenti curve (in un intervallo [-10 10]) e visualizzarlo:

$$y = x - 1;$$
$$y = 2x^2 + x - 12;$$
$$y = -0.1x^3 + 2x^2 - 10x - 12$$

visualizzare, per ogni valore di x, la curva avente y maggiore

Plot 3D

Diagrammi lineari a tre dimensioni

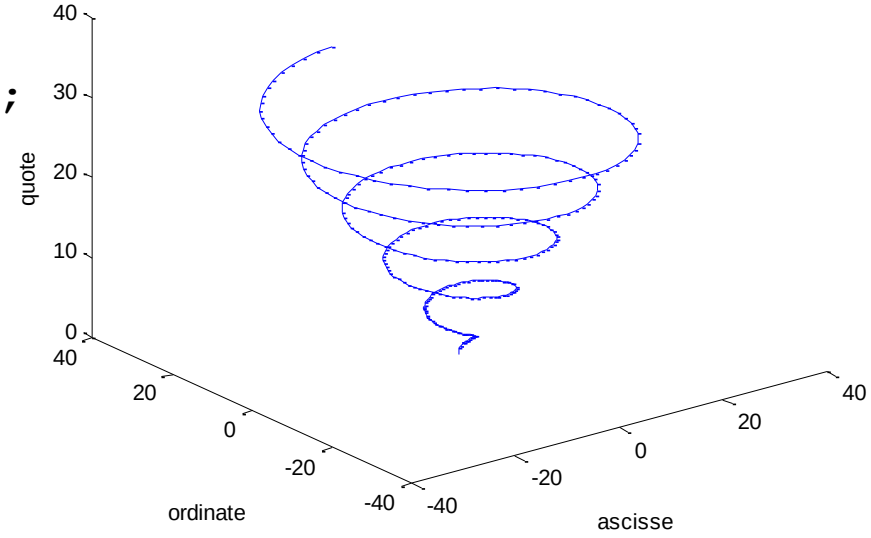
Generalizzazione del diagramma a due dimensione: insieme di terne di coordinate

`plot3(x,y,z)` disegna un diagramma cartesiano con `x` come ascisse, `y` come ordinate e `z` come quote

funzioni `xlabel, ylabel, zlabel, title`

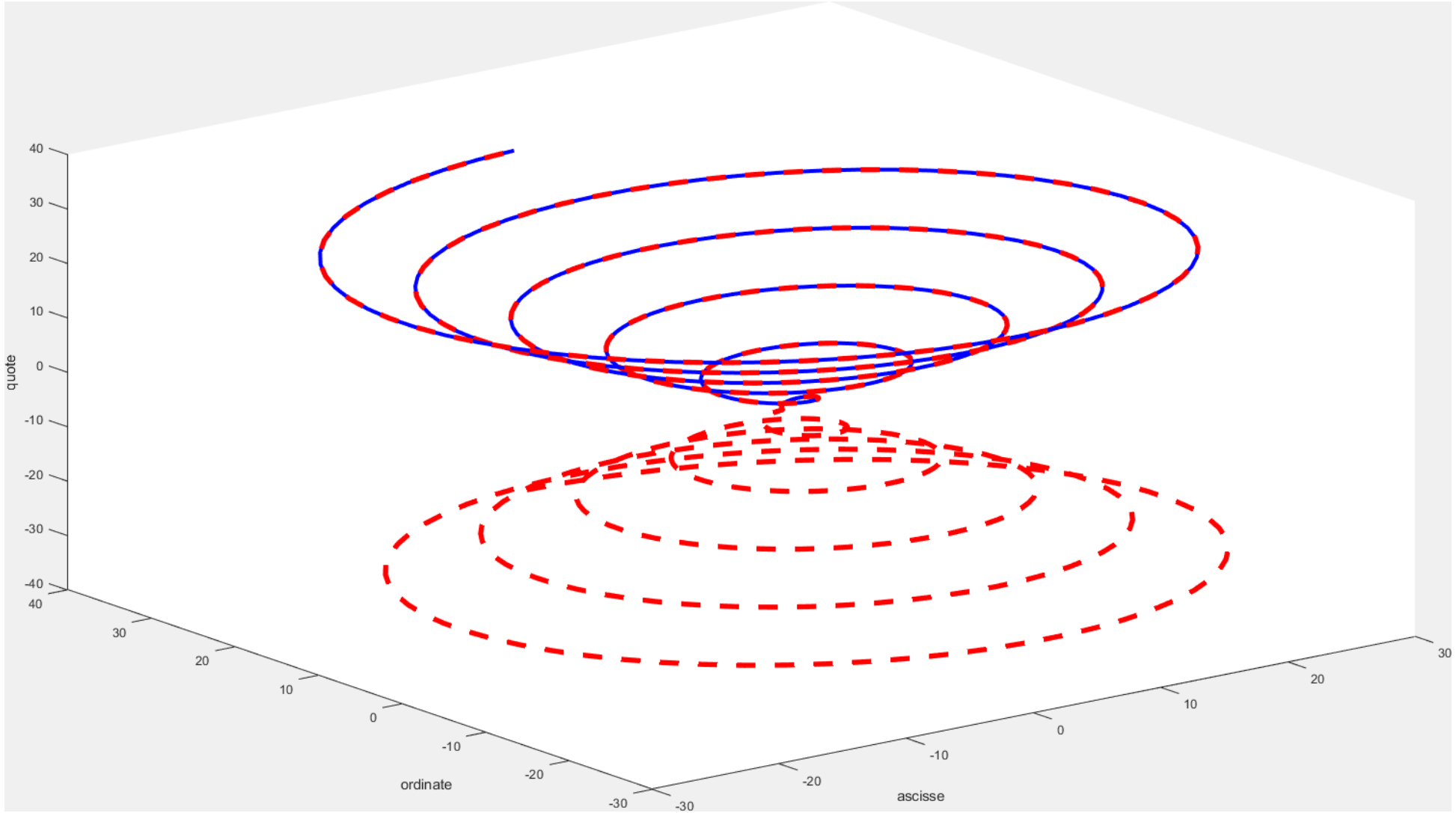
Esempio

```
>> t = 0:0.1:10*pi;
>> plot3 (t.*sin(t), t.*cos(t), t);
>> xlabel('ascisse');
>> ylabel('ordinate');
>> zlabel('quote');
```



Cosa fa?

```
figure(2),
t = 0: 0.1 : 10*pi;
plot3(abs(t).*sin(t), abs(t).*cos(t), t, 'b-','LineWidth',
3);
hold on
t = [-t(end : -1 : 1), t];
plot3(abs(t).*sin(t), abs(t).*cos(t), t, 'r--','LineWidth',
4);
xlabel('ascisse');
ylabel('ordinate');
zlabel('quote');
hold off
```


quote
40
30
20
10
0
-10
-20
-30
-40
ordinate
40
30
20
10
0
-10
-20
-30
ascisse
-30
-20
-10
0
10
20
30

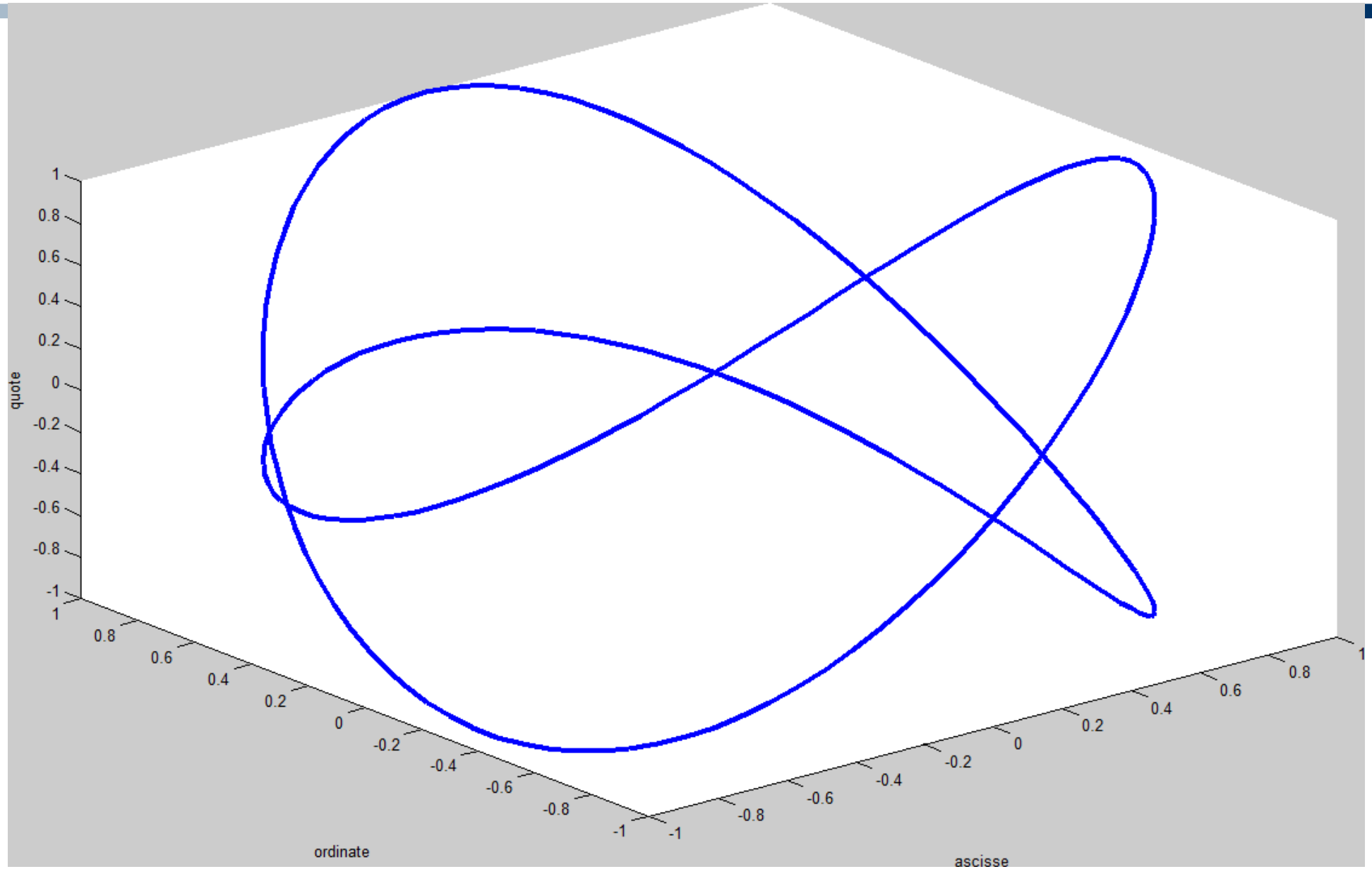
La funzione linspace

Linspace(a,b,n):
crea un vettore di n punti equispaziati tra a e b

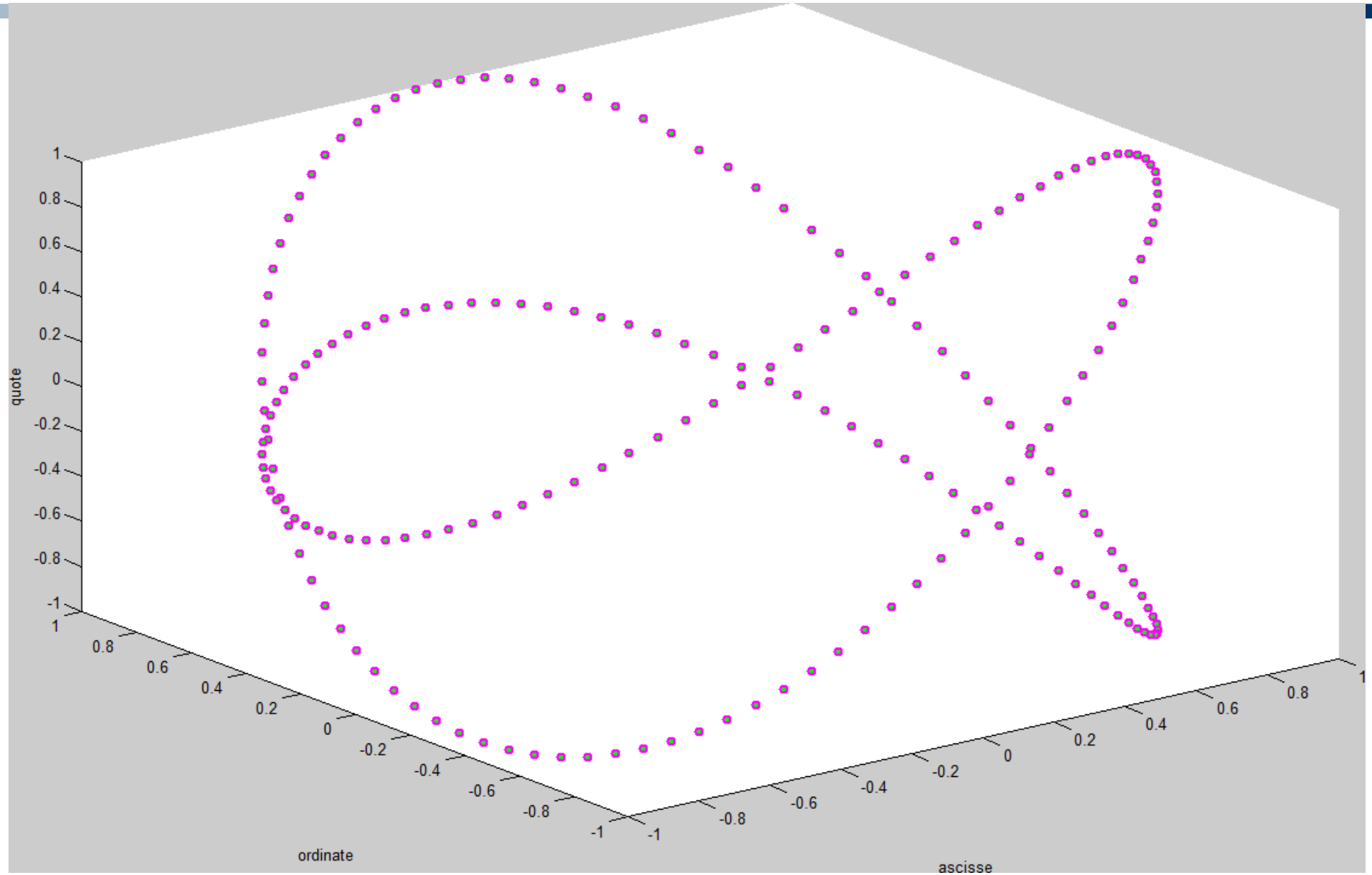
plot restituisce un hanlde, una variabile di riferimento per poter accedere nuovamente all'insieme di punti disegnato

Set(plot_handle, 'Property Name', PropertyVal)
permette di modificare

```
t = linspace(0,4*pi ,200);
plot_hnd = plot3(sin(t),cos(t),cos(3/2 *t))
set(plot_hnd, 'LineWidth', 3)
xlabel('ascisse');
ylabel('ordinate');
zlabel('quote');
```


quote
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
ordinate
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
ascisse
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

```
t = linspace(0,4*pi ,200);
plot_hnd = plot3(sin(t),cos(t),cos(3/2 *t))
set(plot_hnd, 'LineWidth', 3)
xlabel('ascisse');
ylabel('ordinate');
zlabel('quote');

set(plot_hnd ,'LineStyle','none','Marker','o',
'MarkerFaceColor', [0 1 0],...
'MarkerEdgeColor',[1 0 1],...
'MarkerSize',5,'LineWidth' ,1.5)
```


quote
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
ordinate
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
ascisse
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

Superfici

Superfici

Come si disegna una superficie che rappresenta una funzione a due variabili $z = f(x,y)$?

Occorre definire il dominio che non è più un intervallo in una retta ma una porzione del piano

Superfici

Come si disegna una superficie che rappresenta una funzione a due variabili $z = f(x,y)$?

Occorre definire il dominio che non è più un intervallo in una retta ma una porzione del piano

La funzione `mesh(xx,yy,zz)` genera superficie, a partire da tre argomenti

- `xx` contiene le ascisse
- `yy` contiene le ordinate
- `zz` contiene le quote

`xx` e `yy` sono matrici che identificano una griglia in corrispondenza del quale per `zz` rappresenta il valore della funzione

Funzione meshgrid

Le due matrici, xx, e yy, si possono costruire, mediante la funzione `meshgrid(x,y)`

`[xx,yy]=meshgrid(x,y)`

- `x` e `y` sono due vettori
- `xx` e `yy` sono due matrici entrambe di legth(y) righe e length(x) colonne
- la prima, `xx`, contiene, ripetuti in ogni riga, i valori di `x`
- la seconda, `yy` , contiene, ripetuti in ogni colonna, i valori di `y` trasposto

```
[xx, yy] = meshgrid([-3 : 3], [-4 : 4]);
```

xx =

-3	-2	-1	0	1	2	3
-3	-2	-1	0	1	2	3
-3	-2	-1	0	1	2	3
-3	-2	-1	0	1	2	3
-3	-2	-1	0	1	2	3
-3	-2	-1	0	1	2	3
-3	-2	-1	0	1	2	3
-3	-2	-1	0	1	2	3
-3	-2	-1	0	1	2	3

yy =

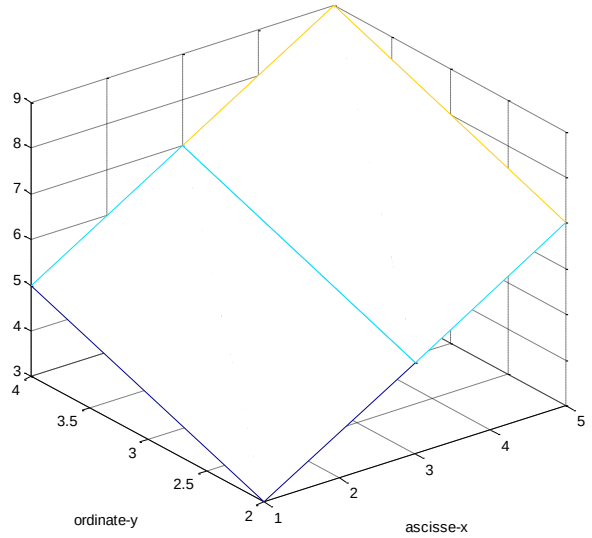
-4	-4	-4	-4	-4	-4	-4
-3	-3	-3	-3	-3	-3	-3
-2	-2	-2	-2	-2	-2	-2
-1	-1	-1	-1	-1	-1	-1
0	0	0	0	0	0	0
1	1	1	1	1	1	1
2	2	2	2	2	2	2
3	3	3	3	3	3	3
4	4	4	4	4	4	4

È possibile quindi valutare una funzione di queste due matrici, e.g., zz = xx + yy , e disegnarla mediante mesh

Superfici: esempi

Disegnamo z=x+y

```
›› x=[1, 3, 5];
›› y=[2, 4];
›› [xx,yy]=meshgrid(x,y);
›› zz=xx+yy;
›› mesh(xx,yy,zz);
›› xlabel('ascisse-x');
›› ylabel('ordinate-y');
```




```
›› xx
xx =
     1     3     5
     1     3     5
```

```
›› yy
yy =
     2     2     2
     4     4     4
```

Punti di coordinate (x,y)…

(1,2) (3,2) (5,2)
(1,4) (3,4) (5,4)

```
›› zz
zz =
     3     5     7
     5     7     9
```

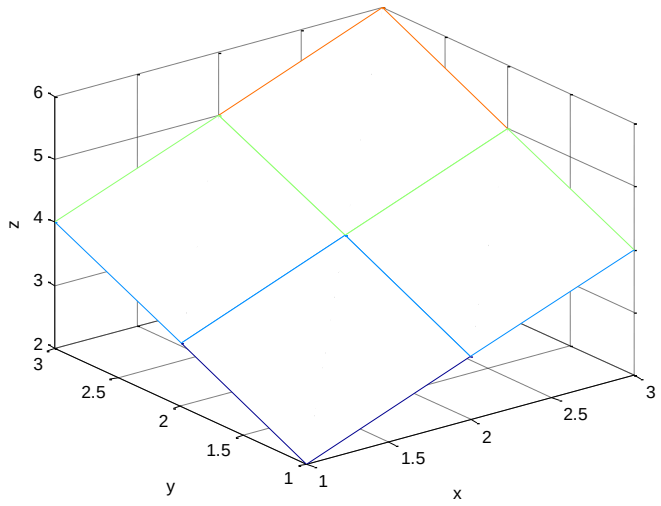
…hanno coordinate (x,y,z)

(1,2,3) (3,2,5) (5,2,7)
(1,4,5) (3,4,7) (5,4,9)

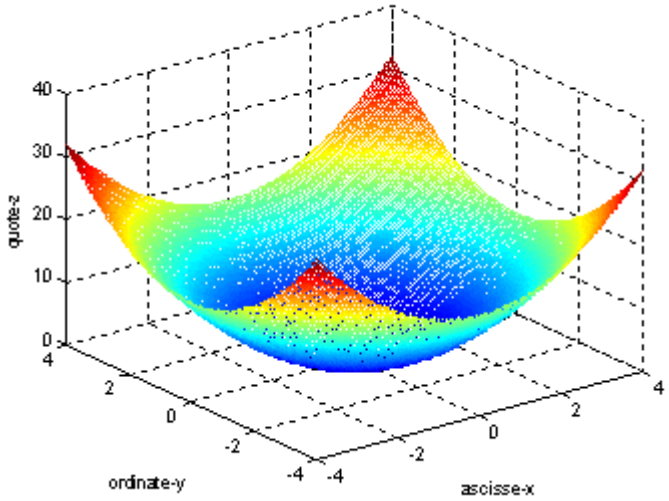
(NB: z=x+y)

Superfici: esempi (2)

```
>> x=[1:1:3];
>> y=x;
>> [xx,yy]=meshgrid(x,y);
>> zz=xx+yy;
>> mesh(xx,yy,zz);
>> xlabel('x');
>> ylabel('y');
>> zlabel('z');
```




```
>> x=[-4:0.05:4];
>> y=x;
>> [xx,yy]=meshgrid(x,y);
>> zz=xx .^ 2 + yy .^ 2;
>> mesh(xx,yy,zz);
>> xlabel('ascisse-x');
>> ylabel('ordinate-y');
>> zlabel('quote-z');
```



Hold on

Le superfici vengono visualizzate su un grafico 3D.

È quindi possibile aggiungere degli elementi in sovraimpressione utilizzando la funzione

- plot3(), mesh(),altre funzioni grafiche quali surf() etc..

- Per sovrascrivere ad un grafico usare la funzione hold on e hold off quando si ha terminato

Esempio, disegnare in sovraimpressione al paraboloide precedente la curva $\begin{cases} z = x^2 \\ y = 0 \end{cases}$

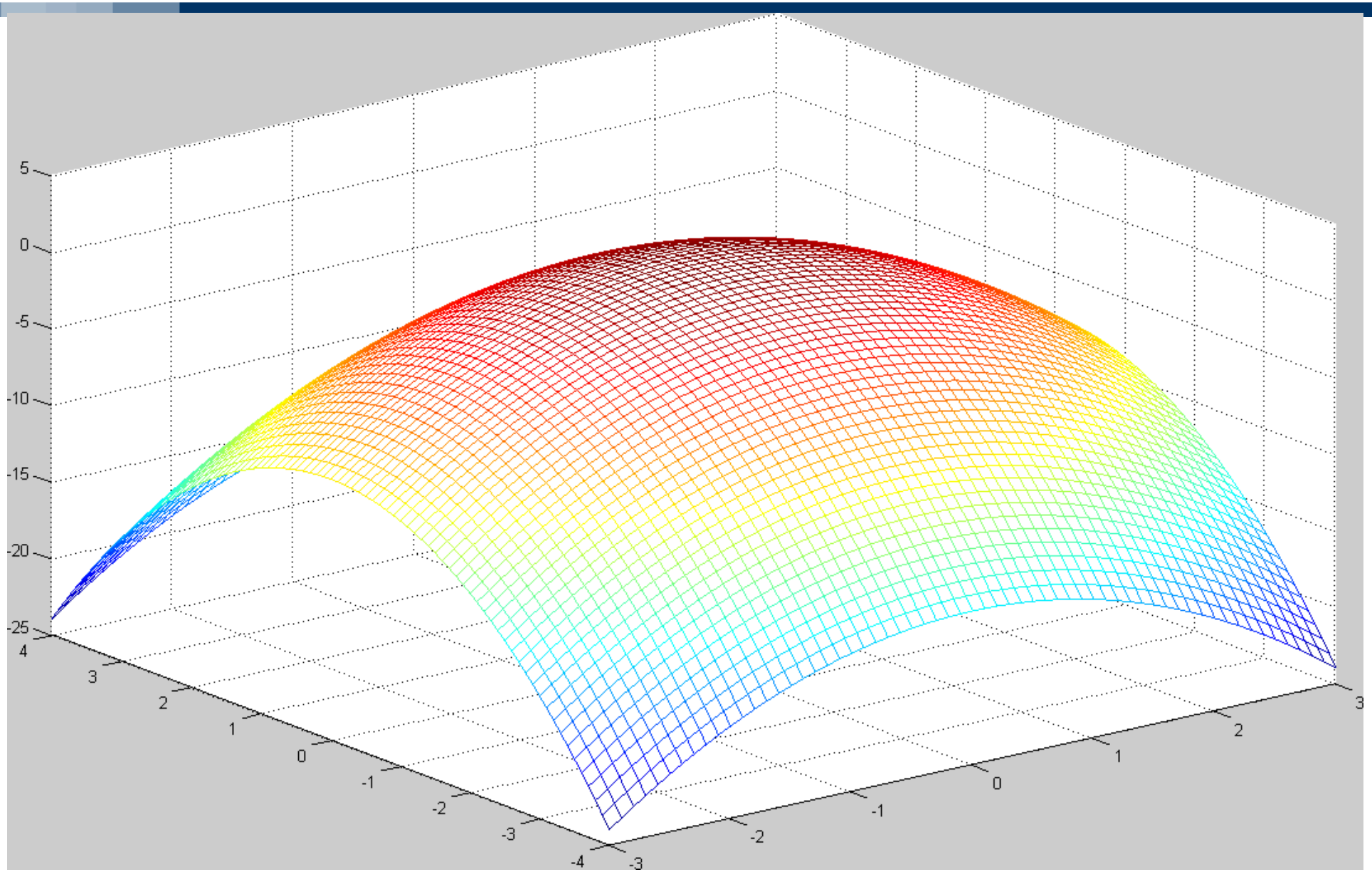
dove `x=[-4:0.05:4];`

```
figure, mesh(xx, yy, zz)
hold on
% aggiunge una linea rossa con uno spessore di 5
plot3(x, zeros(size(x)), x.^2, 'r-', 'LineWidth', 5);
hold off
```

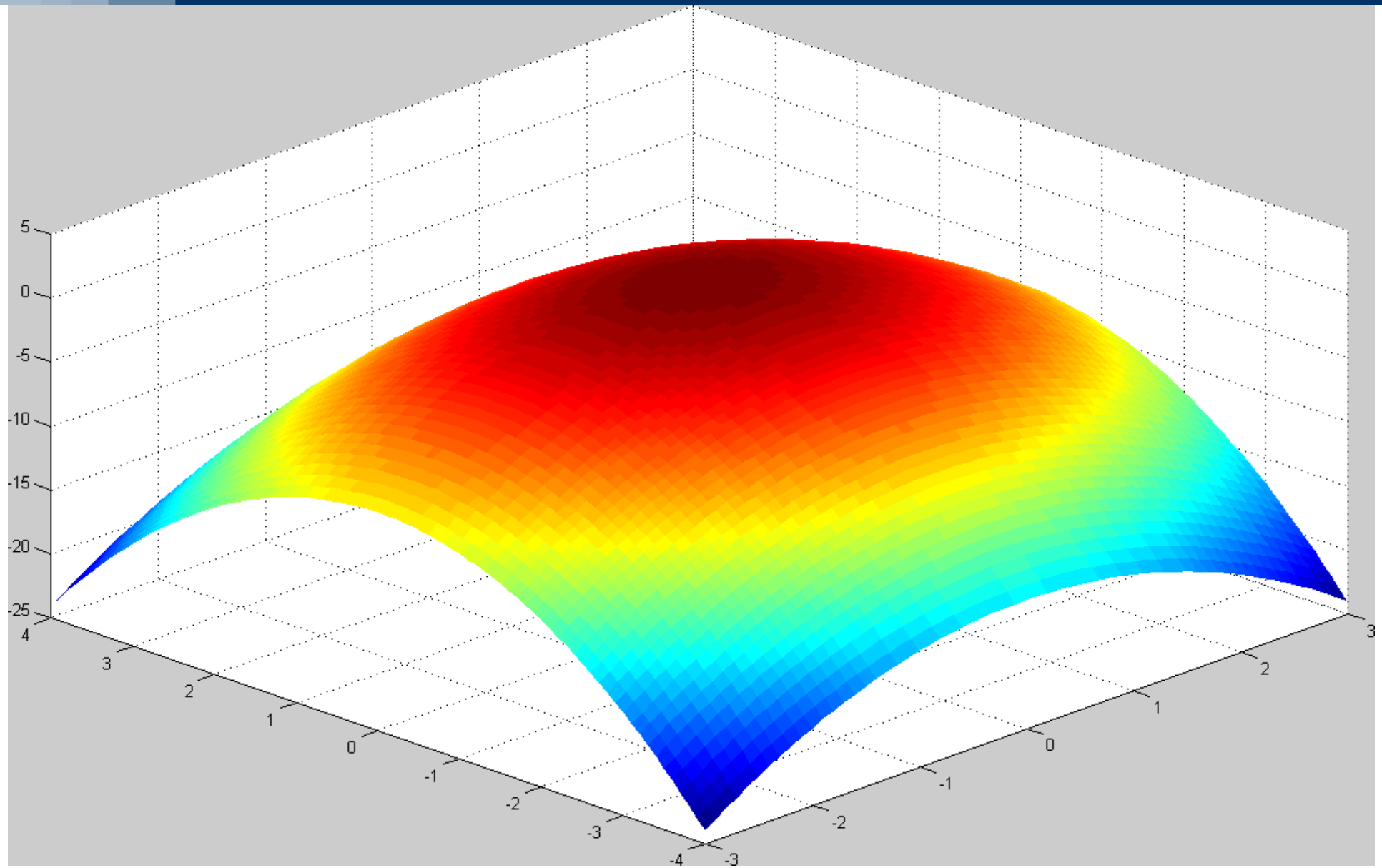
Mesh

```
[xx, yy] = meshgrid([-3 : 0.1 :3], [-4 : 0.1 :4]);
f = @(x, y)(1 - x.^2 - y.^2);


figure,
aa = mesh(xx, yy, f(xx, yy))
```

Mesh unisce i punti con delle linee colorate. Di default il colore indica il valore della quota

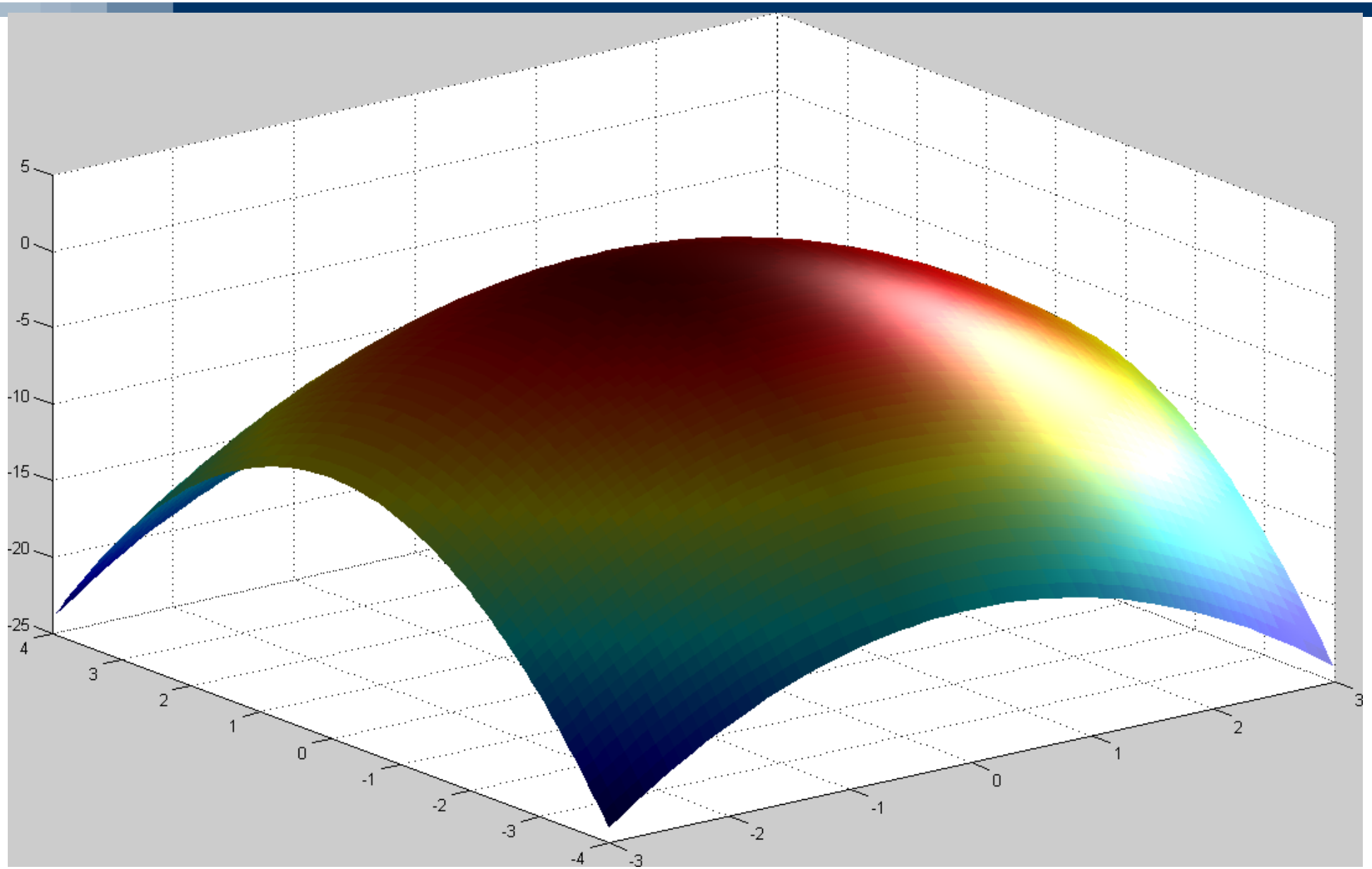

5
0
-5
-10
-15
-20
-25
4
3
2
1
0
-1
-2
-3
-4
-3
-2
-1
0
1
2
3

```
[xx, yy] = meshgrid([-3 : 0.1 :3], [-4 : 0.1 :4]);
f = @(x, y)(1 - x.^2 - y.^2);


figure,
aa = surf(xx, yy, f(xx, yy))
set(aa, 'EdgeColor', 'none')
```


5
0
-5
-10
-15
-20
-25
4
3
2
1
0
-1
-2
-3
-4
-3
-2
-1
0
1
2
3

```
[xx, yy] = meshgrid([-3 : 0.1 :3], [-4 : 0.1 :4]);
f = @(x, y)(1 - x.^2 - y.^2);


figure,
aa = surf(xx, yy, f(xx, yy))
set(aa, 'EdgeColor', 'none')
light % aggiunge una sorgente luminosa per il rendering
```

Surf riempie le regioni tra le linee con del colore che, di default, dipende dal valore della quota


5
0
-5
-10
-15
-20
-25
4
3
2
1
0
-1
-2
-3
-4
-3
-2
-1
0
1
2
3

Hold on

Le superfici vengono visualizzate su un grafico 3D.

È quindi possibile aggiungere degli elementi in sovraimpressione utilizzando la funzione

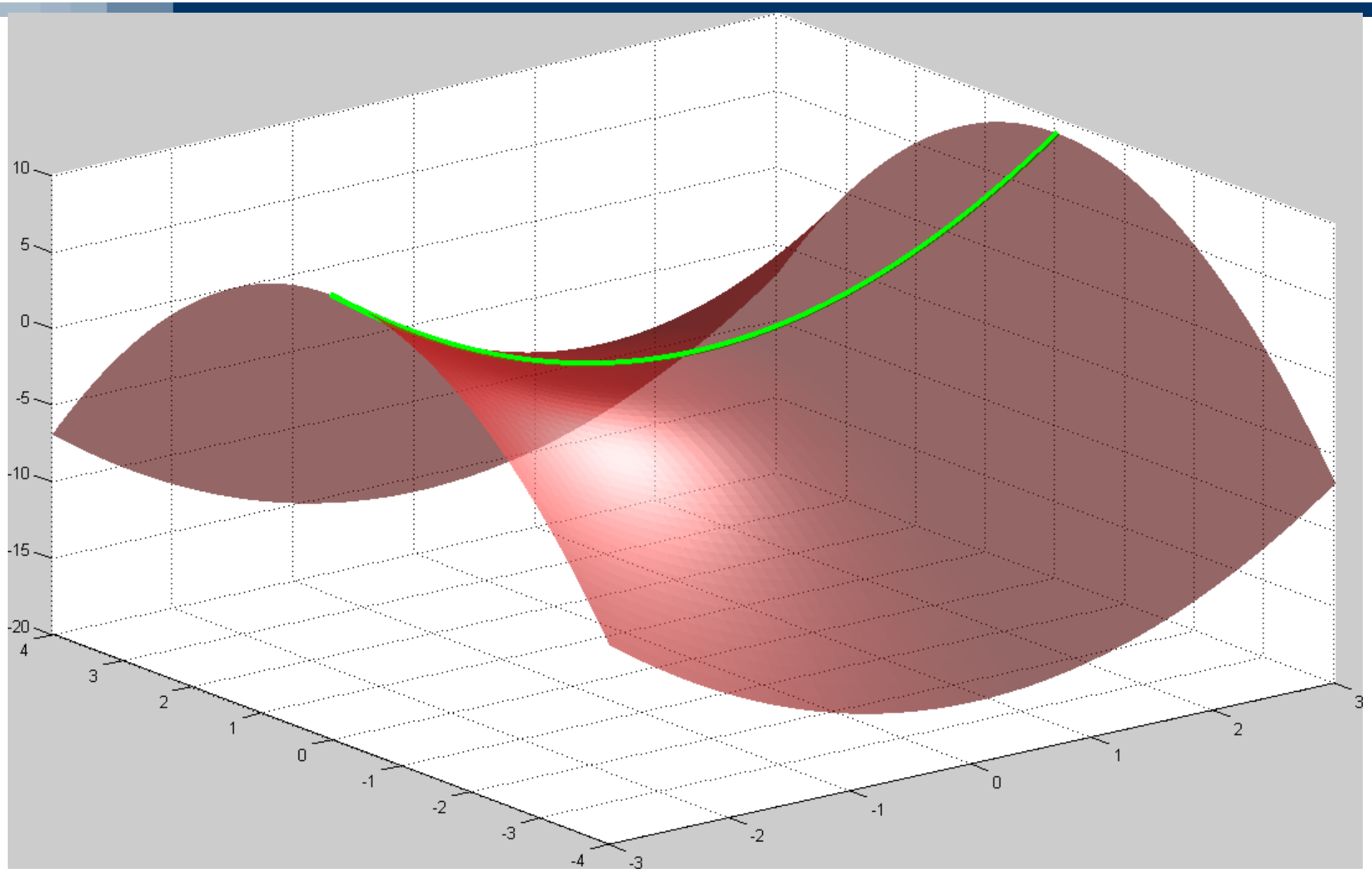
- plot3(), mesh(),altre funzioni grafiche quali surf() etc..

- Per sovrascrivere ad un grafico usare la funzione hold on e hold off quando si ha terminato

Esempio, disegnare in sovraimpressione alla quadrica

$z = x^2 - y^2$ la curva $\begin{cases} z = x^2 \\ y = 0 \end{cases}$

```
[xx, yy] = meshgrid([-3 : 0.1 :3], [-4 : 0.1 :4]);
f = @(x, y)(x.^2 - y.^2);
figure(),
aa = surf(xx, yy, f(xx, yy))
hold on
x = xx(1, :);
y = zeros(size(x));
bb = plot3(x, y, f(x,y), 'g-')
set(aa, 'EdgeColor', 'none', 'FaceColor', 'red', 'FaceAlpha', 0.6)
set(bb, 'Linewidth', 3)
light
hold off
```


10
5
0
-5
-10
-15
-20
4
3
2
1
0
-1
-2
-3
-4
-3
-2
-1
0
1
2
3

Disegnare la funzione

$$z = \frac{\sin\left(\sqrt{x^2 + y^2}\right)}{\sqrt{x^2 + y^2}}$$

e una curva su questa funzione passante per l'origine

```
tx=[-8:0.1:8];
ty=tx;
[xx, yy] = meshgrid (tx, ty);
f = @(x,y)(sin(sqrt(x.^2 + y.^2)) ./ sqrt(x.^2 + y.^2));
figure,
aa = surf(xx, yy, f(xx, yy));
hold on
bb = plot3(tx, tx, f(tx, tx), 'r-' ,'LineWidth', 3)
set(aa, 'EdgeColor', 'none')
```

Superfici: esempi (3)