Laborator 6. Reprezentari grafice 3-D

Consideratii teoretice

Traim intr-o lume cu trei dimensiuni si o mare parte din informatiile noastre, sunt cel mai bine evidentiate, cu ajutorul tehnicilor de interpretare 3-D. Din fericire, MatLab-ul ofera un pachet de functii, de reprezentare grafica, in sistem de coordonate 3-D, care permit vizualizarea rapida a datelor introduse.

Acest laborator este destinat sa prezinte aceste functii si sa duca la o buna intelegere a capacitatii de constructie si vizualizare a unor rezultate in trei dimensiuni.

In laboratoarele trecute am prezentat reprezentarea grafica 2-D, intr-un sistem de doua axe (x,y), o reprezentare grafica in trei dimensiuni necesita o a treia axa (x,y,z), fapt ce implica un set de trei parametri atribuiti unei functii de reprezentare grafica, deci vom avea un sistem de 3 axe X,Y,Z, ceea ce duce la de 3 parametri ai unei functii.

Una din functiile de reprezentare grafica 3D este functia **plot3.** Este similara cu functia plot din reprezentarea 2-D.

```
t=0:0.1:10*pi;          % definim vectorul timp
x=exp(-t/20).*cos(t);   % functia x
y=exp(-t/20).*sin(t);   % functia y
z=t;                    % functia z
plot3(x,y,z);           % reprezentare 3D
xlabel('x');
ylabel('y');
zlabel('z');
grid on                 % caroiaj
```

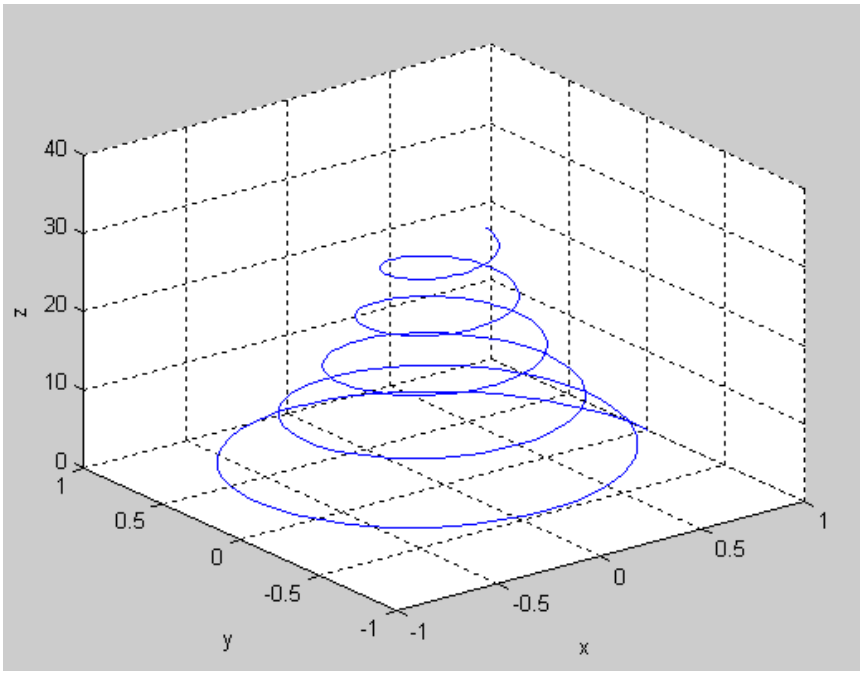

Fig. 1

Forma generala a acestei functii este plot3(x,y,z,'string'), unde x,y,z reprezinta coordonatele reprezentari grafice.

1. in cazul in care x,y,z sunt vectori de aceasi lungime, o linie de reprezentare 3-D este creata prin conectarea coordonatelor specificate de elementele vectorilor x,y,z.
2. in cazul in care x,y,z sunt matrici care au acelasi numar de lini si de coloane, mai multe lini de reprezentare vor fi create de la coloanele matricelor
3. daca unele din variabilele de intrare sunt matrici, iar altele sunt vectori si vectori au aceasi lungime fie ca numar de lini, fie ca numar de coloane din matrice, MatLab-ul v-a reproduce in aceasi maniera ca in cazurile precedente liniile de reprezentare grafica pentru ca graficul sa fie creat.

O proprietate a functiilor de reprezentare grafica 3-D este **view**(u,a), care permite vizualizarea rezultatelor sub diferite ungiuri. Primul parametru al proprietatii **view** este unghiul pe care il descrie un plan vertival fix, iar al doilea parametru reprezinta altitudinea cu care se doreste rotirea axelor.

Daca adaugam liniilor de program scrise anterior liniile de mai jos vom evidentia rotirea unui grafic 3-D :

```
subplot(221)
plot3(x,y,z);
xlabel('x');
ylabel('y');
zlabel('z');
view(-10,10);
title('Reprezentare 3-D')

subplot(222)
plot3(x,y,z);
xlabel('x');
ylabel('y');
zlabel('z');
view(-9,56);
title('Reprezentare 3-D cu view(-9,56)')

subplot(223)
plot3(x,y,z);
xlabel('x');
ylabel('y');
zlabel('z');
view(0,90);
title('Reprezentare 3-D cu view(0,90)')

subplot(224)
plot3(x,y,z);
xlabel('x');
ylabel('y');
zlabel('z');
view(90,0);
title('Reprezentare 3-D cu view(90,0)')
```

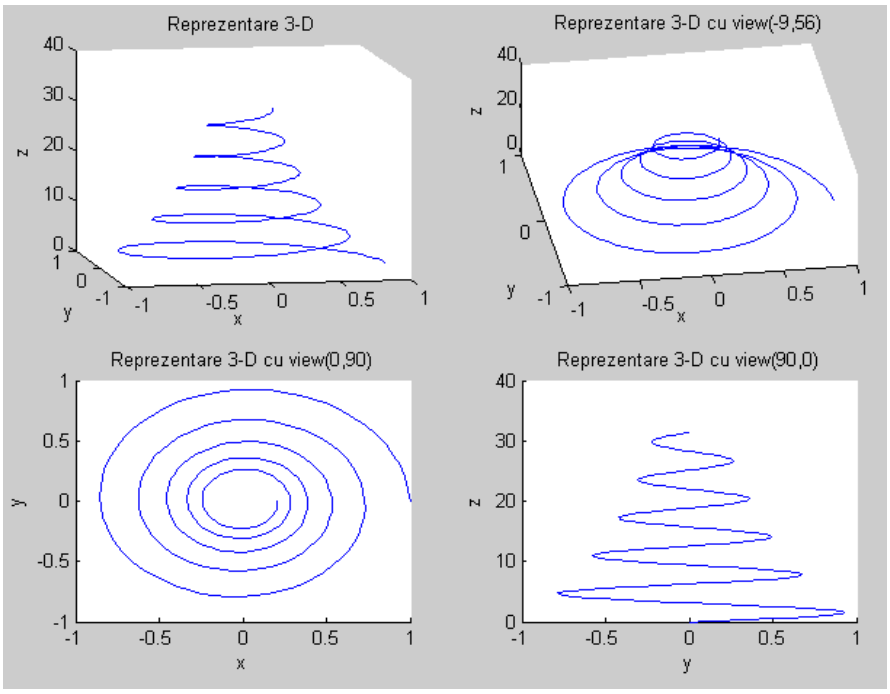

Fig. 2

In figura 2 se observa reprezentarea 3-D a funcitiilor x, y si z, vazute in diferite unghiuri.

De cele mai multe ori prin grafica 3-D se doreste reprezentarea grafica a unor suprafete, mai mult decat simple curbe. In acest scop, MatLab-ul pune la dispozitie doua functii speciale: **surf** si **mesh.** De cele mai multe functiile matematice sunt functii de doua variabile X,Y si pentru fiecare pereche X,Y exista un Z. Z = f(X,Y).

Folosind o bucla iterativa (for), aceasta problema de atribuire, a unei perechi de valori, unei necunoscute, ar putea fii usor rezolvata; dar Matlab-ul are avantajul ca lucreaza usor cu matrici si vectori, ceea ce duce la o simplificare a problemei noastre. Instructiunea care permite atribuirea unui pachet de valori dintr-un set, unei necunoscute este **meshgrid.**

Meshgrid realizeaza o transformare a vectorilor x, y in doua matrici, in functie de parametrii ce trebuie returnati.

Se creaza o figura noua si apoi o interfata ca in laboratoarele trecute, urmand a se adauga in fisierele programului creat liniile de cod din acest laborator.

Ex:

```
x = [-1 0 1];           % vectorul x
y = [9 10 11 12];       % vectorul y
[X,Y] = meshgrid(x,y)   % mashgrid pe vectorul x si y
```

Rezultul acestor instructiuni este :

X = -1 0 1
-1 0 1
-1 0 1
-1 0 1

Y = 9 9 9
10 10 10
11 11 11
12 12 12

Liniile matricii de iesire X vor fi copii ale vectorului x, iar coloanele matricii Y vor fii copii ale vectorului y.

Mesh - aceasta functie realizeaza o incrucisare a liniilor de reprezentare grafica sub forma unei plase de paianjen pe suprafata definita prin coordonatele x,y,z. Pentru a intelege principiul de reprezentare grafica a functiei **mesh** vom considera 3 matrici de dimensiune 3X3 si le vom reprezenta grafic.

Ex:

```
x=[1,2,3]                  % vectorul x
y=[1,2,3]                  % vectorul y
[X,Y]=meshgrid(x,y);       % mashgrid pe vectori x si y, matricea X, Y
Z=X+Y;                     % calculam Z
mesh(X,Y,Z)                % reprezentam 3-D
xlabel('x')
ylabel('y')
zlabel('Z')
title('Reprezentare 3-D mesh')
```

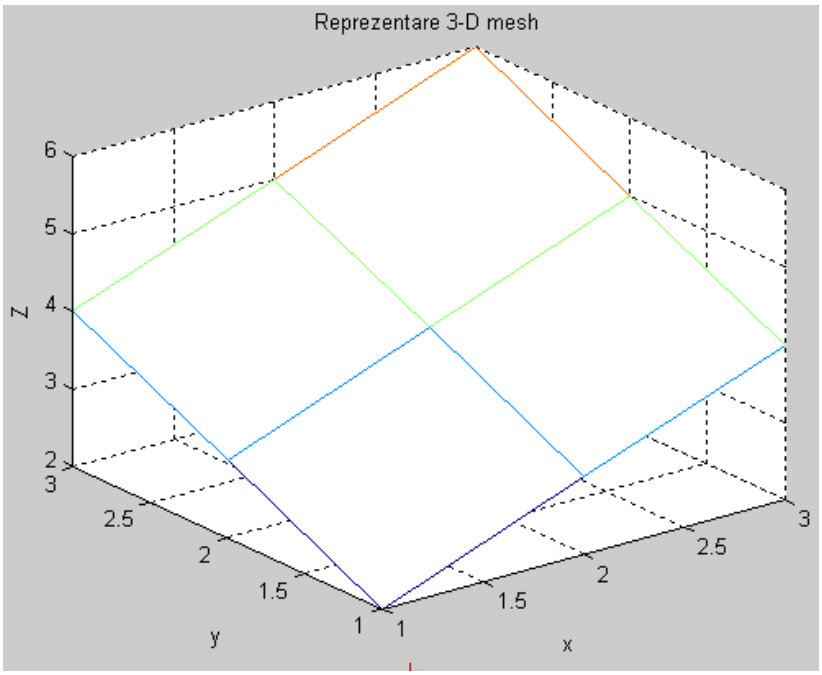

Fig. 3

X = 1 2 3 Y = 1 1 1 Z = 2 3 4
1 2 3 2 2 2 3 4 5
1 2 3 3 3 3 4 5 6

Dupa cum se poate observa si in exemplul anterior, sintaxa functiei mesh este : **mesh(X,Y,Z).**

Fiecare triplet (x(i,j),y(i,j),z(i,j)) corespunde unei linii i si unei coloane j a matricelor X,Y,Z. Functia mesh creaza un schelet al unei suprafete date de coordonate x,y,z. Pentru a vizualiza punctele de coordonate ce corespund graficului nostru se poate folosi functia **meshz(X,Y,Z).**

Surf – se foloseste pentru vizualizarea functiilor matematice intr-o suprafata rectangulara. Aceasta functie este identica cu functia **mesh,** doar ca, prin folosirea acestei functii, reprezentarea va aparea sub forma unei suprafete umbrite, in loc de un ecran grila, in cazul functiei mesh.

Ex:

```
[X, Y] = meshgrid(-pi:pi/10:pi);        % meshgrid pe intervalul -pi,pi
Z = sin (X) .* sin (Y);                 % calculam Z
subplot('position',[0.1 0.53 0.5 0.4])
mesh (X,Y,Z);                           % reprezentam 3-D cu mesh
grid on
xlabel('x')
ylabel('y')
zlabel('z')
title('Reprezentare mesh')
subplot('position',[0.4 0.1 0.5 0.4])
surf (X,Y,Z);                           % reprezentam 3-D cu surf
grid on
xlabel('x')
ylabel('y')
zlabel('z')
title('Reprezentare surf')
```

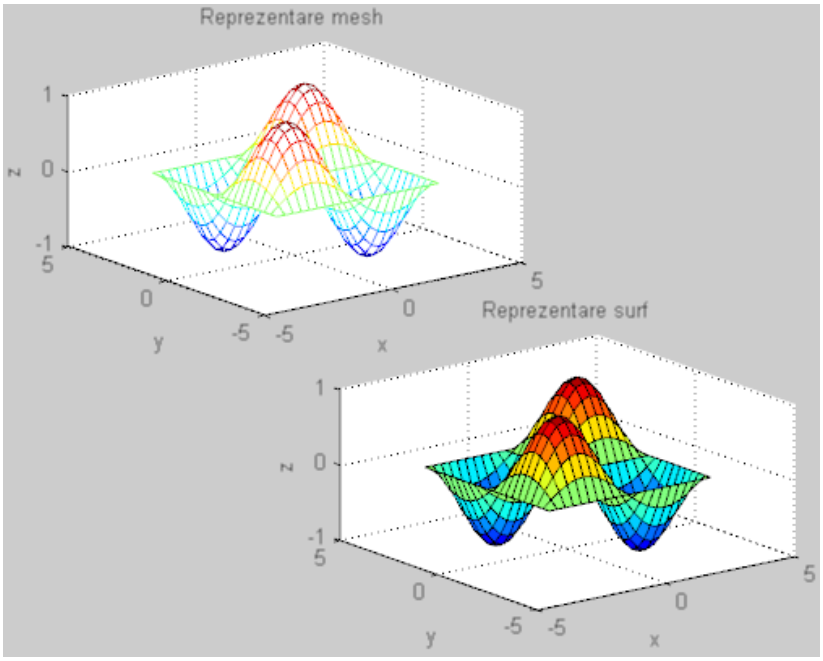

Fig. 4

In figura 4 se observa diferenta dintre cele doua functii de reprezentare grafica 3-D mesh , surf. Daca mesh creaza un schelet al suprafetei, functia surf are proprietatea de a umple aceste suprafete.

O proprietate a functiei surf este **shading**. Aceasta proprietate permite vizualizarea reprezentarii grafice in diferite intensitati ale coloritului.

```
[X, Y] = meshgrid(-pi:pi/10:pi);        % meshgrid pe intervalul -pi,pi
Z = sin (X) .* cos (Y);                 % calculam Z
subplot(221)
surf (X,Y,Z);                           % reprezentam 3-D cu mesh
grid on
xlabel('x')
ylabel('y')
zlabel('z')
title('fara shading')
subplot(222)
surf (X,Y,Z);                           % reprezentam 3-D cu surf
shading('interp');
xlabel('x')
ylabel('y')
zlabel('z')
title('shading interp')
subplot(223)
surf (X,Y,Z);                           % reprezentam 3-D cu surf
shading('flat');
xlabel('x')
ylabel('y')
zlabel('z')
title('shading flat')
subplot(224)
surf (X,Y,Z);                           % reprezentam 3-D cu surf
shading('faceted');
xlabel('x')
ylabel('y')
zlabel('z')
title('shading faceted')
```

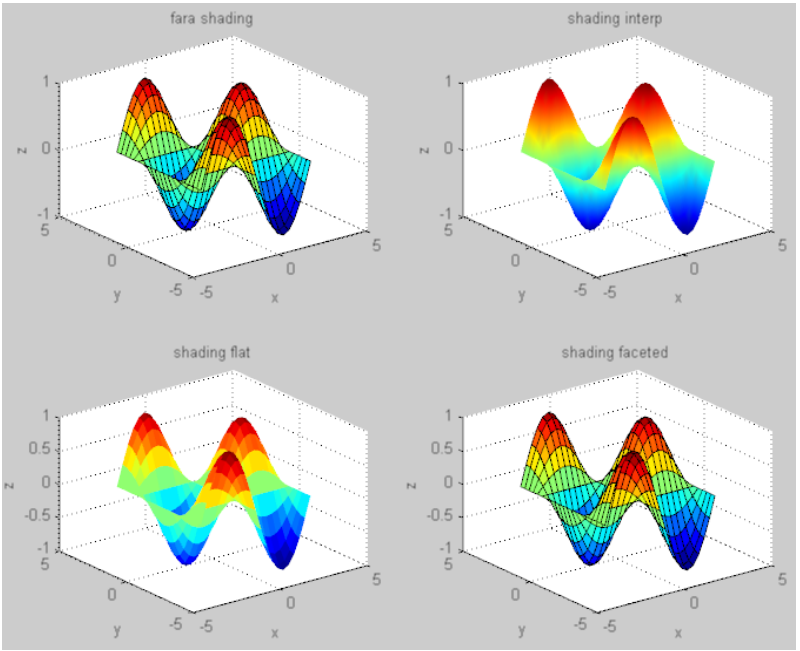

Fig. 5

Dupa cum se observa, proprietatea shading este caracterizata de parametri **flat, interp, faceted**, care este si setarea implicita a functiei surf.

Functia **sphere(n)** – genereaza reprezentarea grafica a unei sfere, sfera definita de (n+1)2 puncte. In cazul in care nu se seteaza o valoarea pentru parametrul n, acesta v-a fi setat la valoarea 20.

Ex:

```
subplot(221)
        sphere          % sfera cu valoare implicita
        axis equal
        title('Sfera')
subplot(222)
        sphere(8)       % sfera cu n=8
        axis equal
        title('Sfera cu n=8')
subplot(223)
        sphere(50)      % sfera cu n=50
        axis equal
        title('Sfera cu n=50')
subplot(224)
[x,y,z] = sphere(25);   % dam valori matricilor x,y,z
surf(x-3,y-2,z);        % reprezentam grafic cu translatie
hold on
surf(x*2,y*2,z*2);      % reprezentam grafic la alta scala
title('Sfera')
```

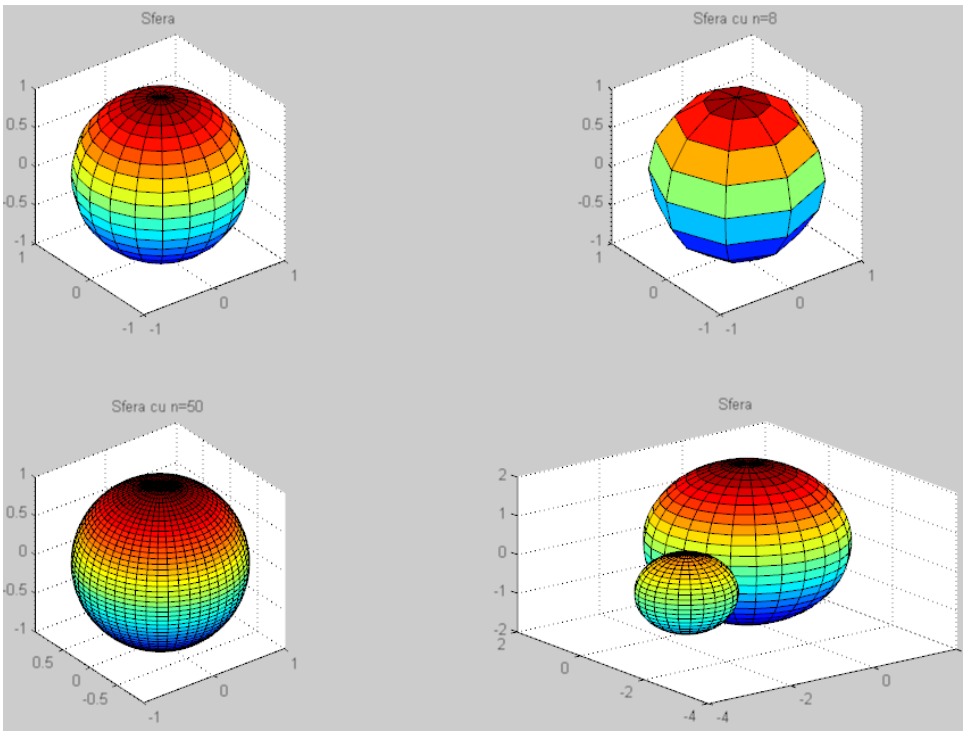

Fig. 6

In figura 6 se pot vizualiza reprezentarile grafice ale functiei **sphere(n).**

O alta functie de reprezentare 3-D este **ellipsoid.** Aceasta functie este bazata pe acelasi principiu ca functia sphere si produce un elipsoid de coordonate x,y,z descris de ecuatia:

$$\left(\frac{x-x_c}{r_c}\right)^2+\left(\frac{y-y_c}{r_y}\right)^2+\left(\frac{z-z_c}{r_z}\right)^2=1$$

Unde x_c, y_c si z_c sunt centrele razelor, iar r_x, r_y si r_z sunt razele corespunzatoare axelor.

Forma generala a functiei ellipsoid este : [x,y,z] = ellipsoid(xc,yc,zc,rx,ry,rz,n). Parametrul n reprezinta numarul de puncte in care se calculeaza elipsoidul, daca nu se selecteaza o valoare, acesta va fi setat asemeni functiei sphere la valoarea 20.

Ex:

```
[x,y,z]=ellipsoid(2,0,2,2,1,1);
surf(x,y,z);
axis([0 4 -2 2 0 4]);
hold on
contour(x,y,z);
title('Elipsoid')
```

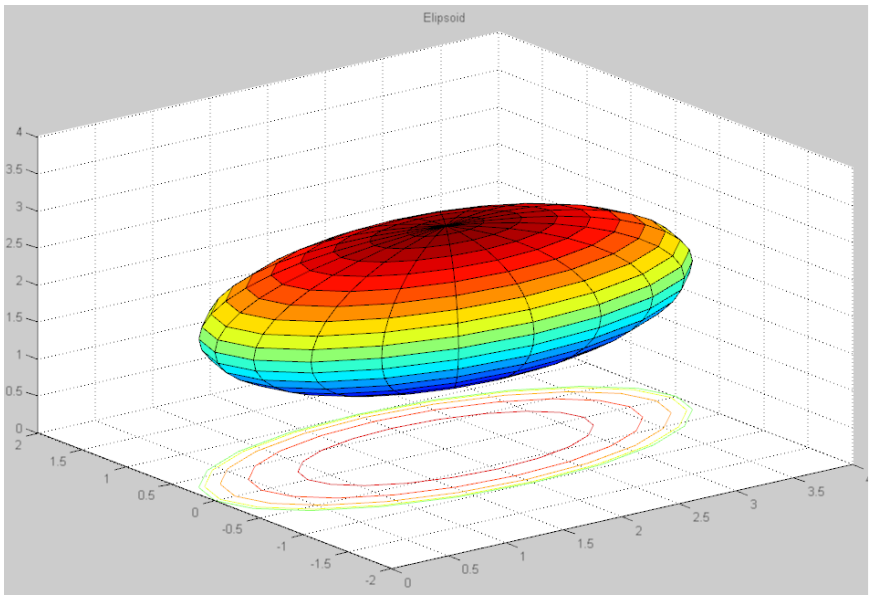

Fig. 7

In liniile de cod de mai sus se observa descrierea functiei ellipsoid si a comenzi **contour**, comanda care traseaza conturul unei reprezentari grafice in functie de coordonatele acesteia.

O alta functie de reprezentare 3-D este **cylinder.** Atunci cand o astfel de functie se apeleaza cu sau fara argumente de intrare si iesire, se genereaza automat o perspectiva in reprezentarea 3-D a unui cilindru.

Forma acestei functii este **cylinder(R,N),** unde primul parametru R, reprezinta un vector raza, care defineste raza cilindrului, distante egale ale punctelor ce urmeaza sa fie reprezentate in inaltime.Daca acest parametru nu se seteaza, implicit v-a primi valoarea [1 1]. Vectorul R este un vector de 2 elemente, ce reprezinta razele cilindrului. Primul

element al vectorului reprezinta raza bazei cilindrului, iar al doilea element reprezinta raza varfului cilindrului.

Parametru N, reprezinta un intreg si specifica numarul de puncte ce vor fi folosite pentru a defini circumferinta cilindrului. Daca aceasta valoare nu se seteaza, la fel ca in cazul functiilor prezentate anterior, valoarea acestestuia este 20.

Ex:

```
r=[2,2];          % definim r
n=40;             % definim n
subplot(121)
cylinder;         % reprezentam un cilindru
title('Cilindru cu cylinder')
subplot(122)
[X,Y,Z]=cylinder(r,n);
Z=Z.*3;
h=surf(X,Y,Z);   % reprezentam cilindru
set(h,'EdgeColor','y');
set(h,'FaceColor','c');
title('Cilindru cu cylinder(r,n)')
```

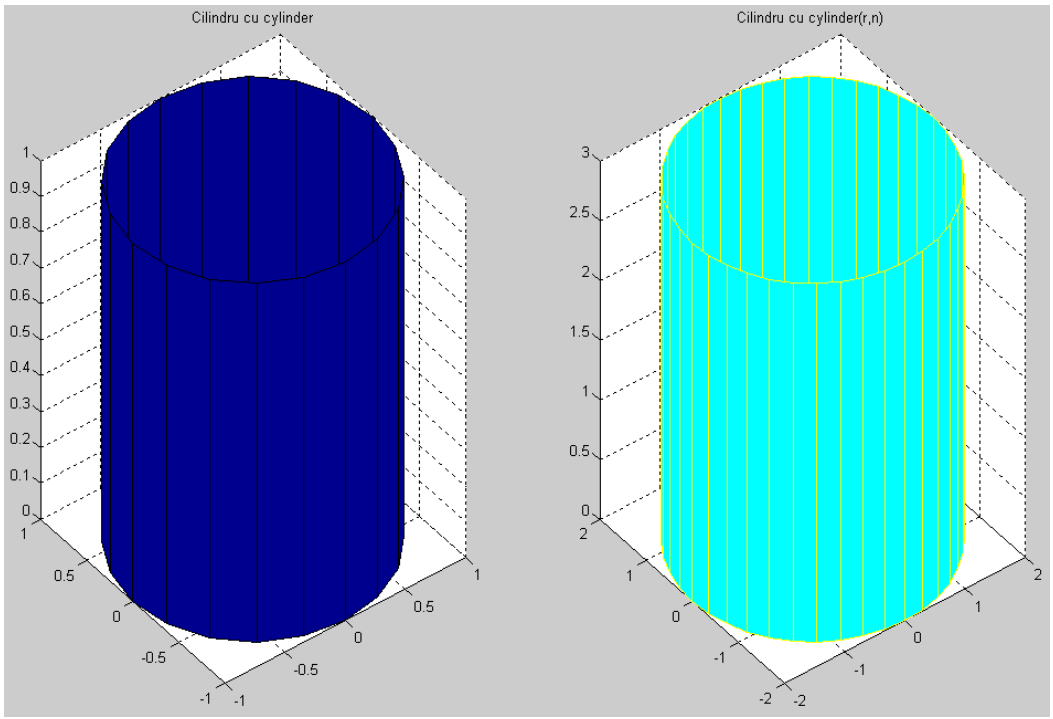

Fig. 8

Cu ajutorul acestei functii se pot crea obiecte geometrice cunoscute :

- con R =[0 r] sau R = [r 0]
- trunchi de con R = [r r/2] sau R = [r/2 r]
- piramida R= [0 r] sau R = [r 0] si N = 3 sau N = 4
- trunchi de piramida R = [0 r] sau R = [r 0] si N = 3 sau N = 4

Crearea unui con :

```
r=[2,0];
n=30;
subplot(121)
[X,Y,Z]=cylinder(r,n);
h=surf(X,Y,Z);
subplot(122)
[X,Y,Z]=cylinder(r,n);
Z=Z.*3;
h=surf(X,Y,Z)
set(h,'EdgeColor','y');
set(h,'FaceColor','c');
```

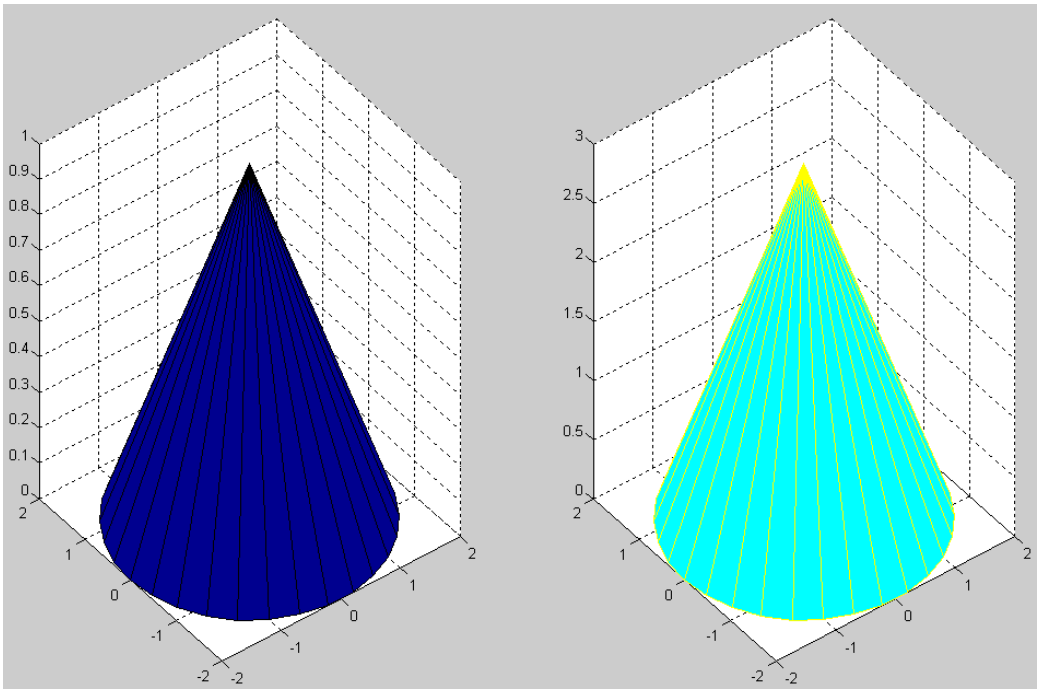

Fig. 9

Pentru a descrie un trunchi de piramida scriem :

```
r=[3.6,1.8];
  n=4;
  subplot(121)
  [X,Y,Z]=cylinder(r,n);

  h=surf(X,Y,Z);
  %h=mesh(X,Y,Z);
  %h=surface(X,Y,Z);

  subplot(122)

[X,Y,Z]=cylinder(r,n);
Z=Z.*3;
h=surf(X,Y,Z);
set(h,'EdgeColor','y');
set(h,'FaceColor','c');
```

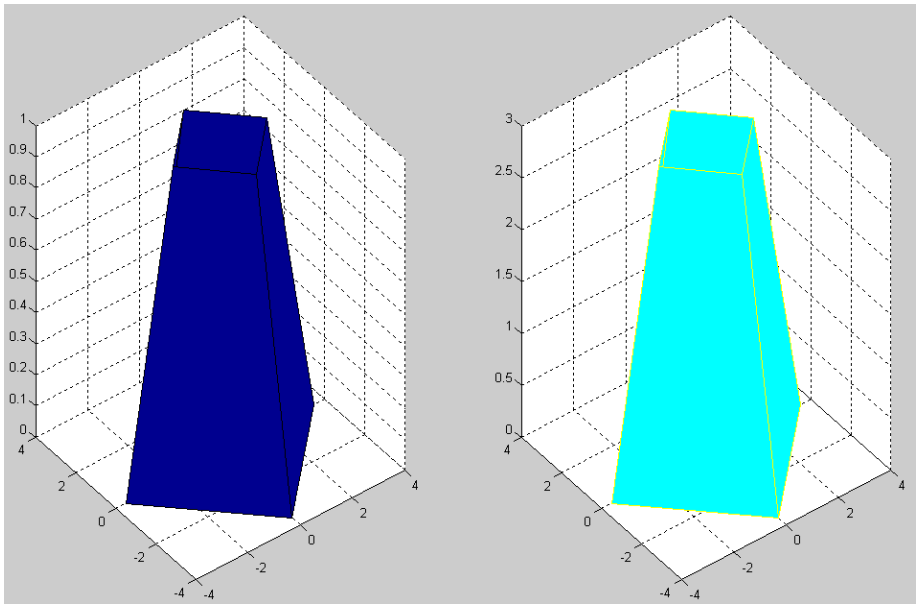

Fig. 10

Obtinerea unei forme geometrice diferite de cilindru, prin folosirea functiei cylinder este posibila datorata numarului de fete pe care dorim sa le vizualizam. Deci, spunem ca N da numarul de fete ale formei geometrice dorite.

Tema de laborator:

1. sa se reprezinte un trunchi de con
2. sa se reprezinte o piramida
3. sa se reprezinte o piramida cu baza triunghi
4. sa se realizeze un program care sa permita selectarea unei forme geometrice dorite dintr-o interfata.