- `dbup` – schimbă spaţiul de lucru curent, cu cel al unui fişier cu extensia `.m`, care a fost apelat;
- `dbdown` – efectuează operaţia inversă comenzii `dbup`;
- `dbmex` – execuă depanarea unui fişier cu extensia `.mex` (creat cu unul din limbajele Fortran şi C);
- `dbquit` – încheierea depanării.

1.6.1 Comenzi pentru gestionarea fişierelor

- `type file` – listează conţinutul fişierului `file` din directorul curent;
- `delete file` – şterge fişierului `file` din directorul curent;
- `what` – listează fişierele cu extensiile `.m` din directorul curent;
- `which numef` – listează calea directorului în care se află funcţia `numef`;

Comanda

```
>>what director
```

extrage lista fişierelor directorului specificat `...\toolbox\matlab\director` Lista acestor directoare poat fi vizualizată prin comanda `help` fără nici un parametru.

1.7 Grafică bidimensională

Sistemul Matlab dispune de o largă gamă de instrucţiuni sau proceduri (funcţii) pentru reprezentarea grafică a datelor, care permit o apelare simplă şi eficientă.

1.7.1 Instrucţiunea `plot`

Formele acceptate de sistemul Matlab pentru instrucţiunea `plot`, privind reprezentarea grafică în plan sunt

```
plot(y)
plot(x,y)
plot(x1,y1,x2,y2,...)
plot(y,s)
plot(x,y,s)
plot(y1,s1,y2,s2,...)
plot(x1,y1,s1,x2,y2,s2,...)
```

În cazul primei forme, dacă `y` este un vector de lungime `m`, se reprezintă grafic linia poligonală ce trece prin punctele de coordonate (i, y_i), $i = \overline{1, \mathtt{m}}$.

Programul 1.7.1. Să considerăm programul, care reprezintă grafic linia poligonală ce unește punctele de coordonate (i, y_i), $i = \overline{1, \mathtt{m}}$, unde `y` reprezintă elementele diagonalei unei matrice magice de ordin `m`:

```
m = input('m:');
y = diag(magic(m));
plot(y)
```

În urma execuţiei programului, pentru `m=10`, se obţine graficul din Figura 1.1.

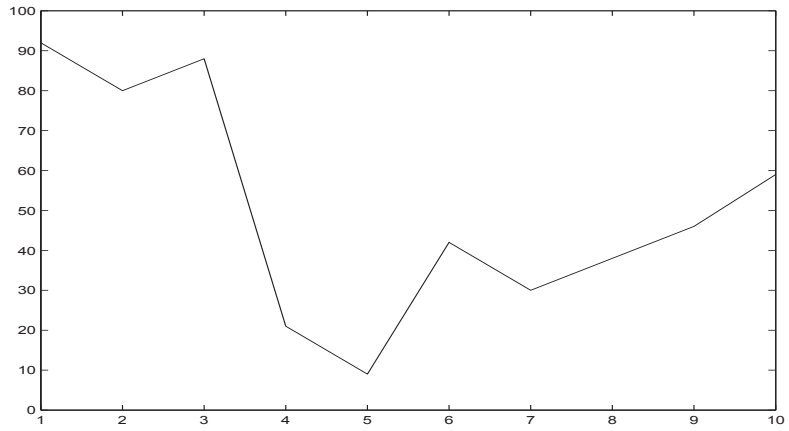

Figura 1.1: Linie poligonală

Dacă `y` este o matrice de tipul `(m,n)`, atunci se reprezintă grafic `n` linii poligonale, corespunzând celor `n` coloane ale matricei `y`, anume $(i, y_{ij})_{i=1}^{\mathtt{m}}$, $j = \overline{1, \mathtt{n}}$.

Programul 1.7.2. Să considerăm programul, care reprezintă grafic liniile poligonale ce unesc respectiv punctele de coordonate $(i, y_{ij})_{i=1}^{m}$, pentru fiecare $j = \overline{1, \mathtt{n}}$, unde `y` este o matrice magică de ordin `m`, iar `n` este un număr întreg pozitiv cel mult `m`:

```
m = input('m:');
n = input('n (n<=m):');
y = magic(m);
plot(y(:,1:n))
```

În urma execuţiei programului, pentru `m=10` şi `n=2`, se obţine graficul din Figura 1.2.

Dacă `y` este de tip complex, atunci prima formă este echivalentă cu a doua formă a instrucţiunii `plot`, adică cu

```
plot(real(y),imag(y))
```

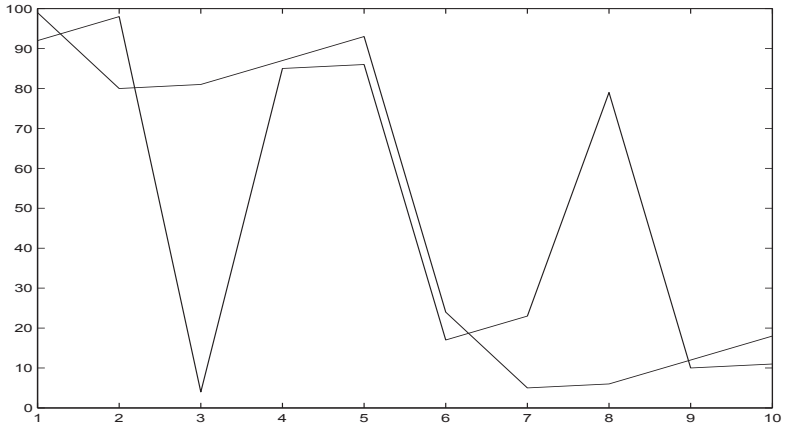

Figura 1.2: Linii poligonale

Programul 1.7.3. Să considerăm programul, care reprezintă grafic funcţia cu valori complexe $z = x\cos(2\pi\, x) + ix^2 \sin(2\pi\, x)\,, \quad x \in [0,1]\,.$

```
m = input('m:');
h =1/m; x = 0:h:1;
z = x.*cos(2*pi*x)+i*x.^2.*sin(2*pi*x);
plot(z)
```

În urma execuţiei programului, pentru `m=100`, se obţine graficul din Figura 1.3.

Remarcăm faptul că pentru toate celelalte forme, dacă avem parametri complecşi, partea imaginară este ignorată.

Dacă se utilizează a doua formă, iar `x` şi `y` sunt vectori de aceaşi lungime `m`, atunci se reprezintă grafic linia poligonală ce trece prin punctele de coordonate (x_i, y_i), $i = \overline{1, \mathtt{m}}$.

Programul 1.7.4. Fie programul care reprezintă grafic funcţia $\sin(2\pi x)$ pe intervalul $[0,1]$:

```
m = input('m:');
h = 1/m; x = 0:h:1;
plot(x,sin(2*pi*x))
```

Execuţia programului, pentru `m=200`, generează graficul din Figura 1.4.

În cazul în care `x` şi `y` sunt matrice având acelaşi tip `(m,n)`, se reprezintă grafic `n` linii poligonale, corespunzând celor `n` coloane ale matricelor, adică $(x_{ij}, y_{ij})_{i=1}^{\mathtt{m}}$, pentru fiecare $j = \overline{1, \mathtt{n}}$. Dacă `x` este un vector de lungime `m`, iar `y` este un vector de lungime `m` sau matrice de tipul `(m,n)`, se reprezintă grafic liniile poligonale ce trec

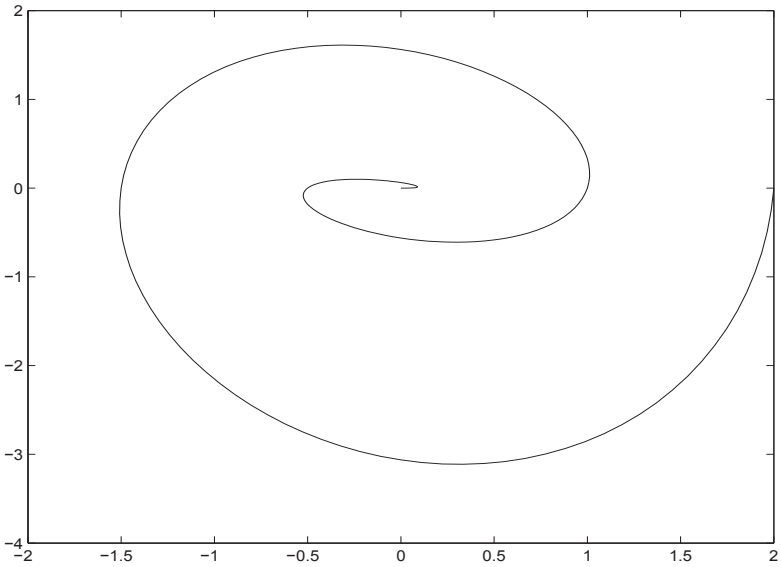

Figura 1.3: $z = x\cos(2\pi\, x) + ix^2 \sin(2\pi\, x)$

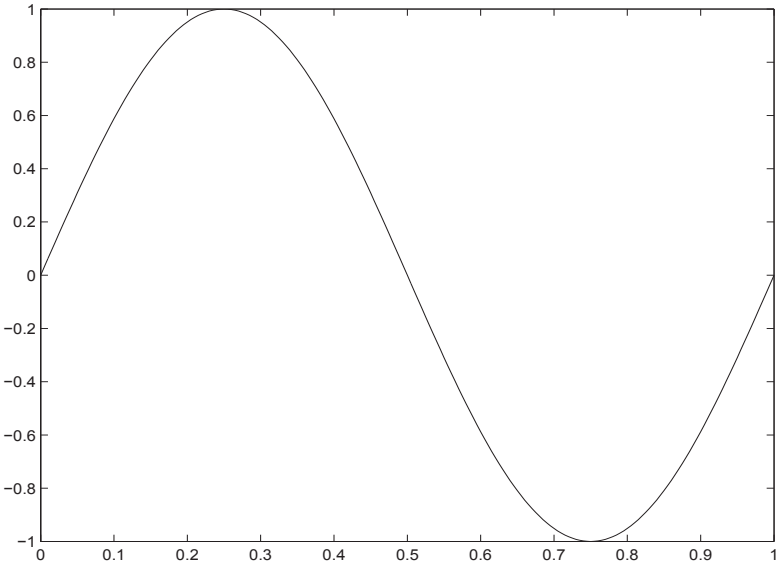

Figura 1.4: $y = \sin(2\pi x)$

prin punctele de coordonate $(x_i, y_i)_{i=1}^{\mathtt{m}}$, respectiv `n` linii poligonale ce trec respectiv prin punctele date prin $(x_i, y_{ij})_{i=1}^{\mathtt{m}}$, pentru fiecare $j = \overline{1, \mathtt{n}}$.

Forma a treia a instrucţiunii `plot` implică reprezentarea grafică pe aceeaşi figură, conform formei precedente, a grupelor de date `(x1,y1)`, `(x2,y2)` ş.a.m.d.

Programul 1.7.5. Vom prezenta două variante de programe, care reprezintă pe aceeaşi figură graficele funcţiilor $\sin(2\pi x)$ şi $\cos(2\pi x)$ pe intervalul $[0, 1]$:

```
m = input('m:');
h = 1/m; x = 0:h:1;
y = 2*pi*x;
plot(x,sin(y),x,cos(y))
```

respectiv

```
m = input('m:');
h = 1/m; x = 0:h:1;
y = 2*pi*x;
y = [sin(y)' cos(y)']; x=[x' x'];
plot(x,y)
```

Executând una din cele două variante, cu `m=200`, se generează graficul din Figura 1.5.

Remarcăm faptul că prima variantă ar fi potrivită dacă cele două funcţii se reprezintă grafic pe acelaşi domeniu, iar a doua variantă când cele două funcţii sunt reprezentate grafic pe domenii diferite.

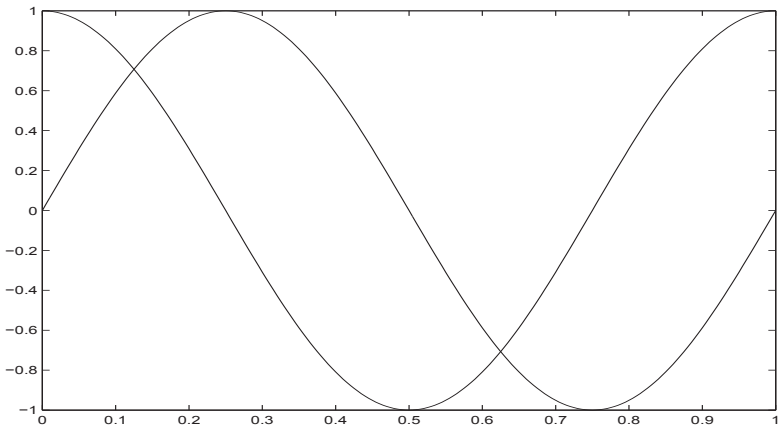

Figura 1.5: $y_1 = \sin(2\pi x)$, $y_2 = \cos(2\pi x)$

Pentru o reprezentare netedă este de dorit ca numărul punctelor ce generează liniile poligonale să fie suficient de mare.

Pe de altă parte, observăm că prin primele trei tipuri de instrucţiune `plot` nu se specifică nici o caracteristică a liniilor reprezentate grafic. Sistemul Matlab face o alegere implicită a acestor caracteristici.

Ultimele patru forme ale instrucţiunii `plot` permit utilizatorului specificarea unor caracteristici ale liniilor reprezentate grafic cu ajutorul parametrului `s`.

Parametrul de tipul `s` este un şir de cel mult trei caractere, prin care se precizează culoarea, tipul de marcaj al punctelor şi tipul de linie.

Culoarea este specificată cu unul din simbolurile următoare: `y` – galben (yellow), `m` – violet (magenta), `c` – ciclamen (cyan), `r` – roşu (red), `g` – verde (green), `b` – albastru (blue), `w` – alb (white), `k` – negru (black).

Tipul de marcaj este dat prin unul din simbolurile: `.` – punct (point), `o` – cerculeţ (circle), `x` – cruciuliţă diagonală (x–mark), `+` – cruciuliţă (plus), $*$ – steluţă (star), `s` – pătrăţel (square), `d` – romb (diamond), `v` – triunghi cu vârf în jos (triangle down), `^` – triunghi cu vârf în sus (triangle up), $<$ – triunghi cu vârf spre stânga (triangle left), $>$ – triunghi cu vârf spre dreapta (triangle right), `p` – steluţă în cinci colţuri (pentagram), `h` – steluţă în şase colţuri (hexagram).

Următoarele patru tipuri de linie sunt disponibile: continuă (solid) (`-`), punctată (dotted) (`:`), întreruptă (dashed) (`--`), linie-punct (dashdot) (`-.`).

De exemplu, dacă `s='c:'`, graficul este trasat cu linie punctată ciclamen, iar dacă `s='bd'` punctele sunt marcate prin semnul romb de culoare albastră, dar nu sunt unite între ele.

Programul 1.7.6. Vom reprezenta grafic funcţiile $\sin(2\pi x)$ şi $\cos(2\pi x)$ pe intervalul $[0,1]$, prima funcţie prin linie continuă, iar a doua prin linie întrerupă, iar punctele de pe cele două curbe, care au abscisele multipli de $\frac{1}{6}$, să fie marcate prin cerculeţe, respectiv prin steluţe.

```
m = input('m:');
h = 1/m; x = 0:h:1; y = 2*pi*x;
y = [sin(y)' cos(y)'];
h = 1/m; p = 0:1/6:1; pp = 2*pi*p;
plot(x,y,p,sin(pp),'o',p,cos(pp),'*')
```

Pentru `m=200`, graficul este cel din Figura 1.6.

1.7.2 Comenzi şi instrucţiuni de gestionare a graficelor

Sistemul Matlab dispune de comenzi şi instrucţiuni prin care sunt gestionate şi specificate anumite caracteristici ale graficelor produse privind titlul, etichetarea axelor, inserarea unor texte, marcarea unor reţele, delimitarea graficului etc:

- `hold on` – păstrează graficul curent, un nou grafic va fi supraimprimat;
- `hold off` – graficul curent este abandonat, fără a fi şters;

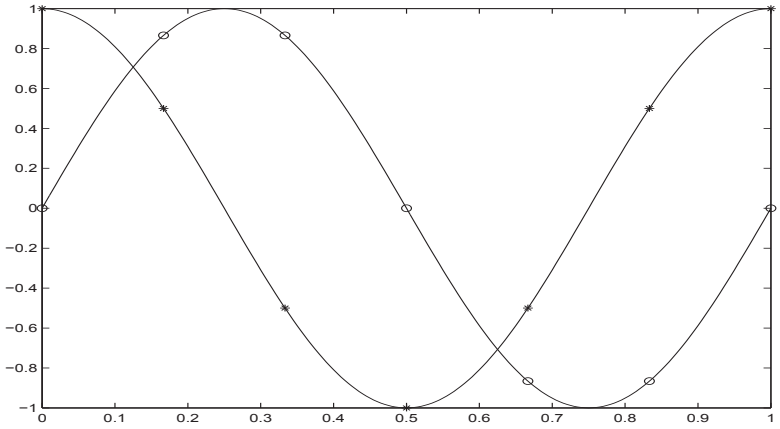

Figura 1.6: $y_1 = \sin(2\pi x)$ şi $y_2 = \cos(2\pi x)$

- `hold` – trece la modul `off`, dacă sistemul se afla în modul `on`, respectiv la modul `on`, când sistemul se află în modul `off`;
- `clf` – şterge graficul curent;
- `cla` – şterge graficul curent, păstrând axele;
- `figure(n)` – activează fereastra cu graficul `n` anterior obţinut;
- `axis([xmin xmax ymin ymax])` – fixează limitele pentru axele absciselor şi ordonatelor pentru figura curentă;
- `v = axis` – întoarce un vector linie ce conţine cele patru limite pentru axele de coordonate, care pot fi şi `-Inf`, `Inf`;
- `axis auto` – consideră forma implicită a limitelor axelor de coordonate;
- `axis manual` – păstrează limitele curente, astfel dacă este utilizată instrucţiunea `hold`, figurile următoare vor avea aceleaşi limite pentru axele de coordonate;
- `axis tight` – atribuie pentru limitele axelor cele mai apropiate valori ce rezultă din datele ce se reprezintă grafic;
- `axis ij` – configurează axele în modul `matrix`, adică originea axelor de coordonate este colţul din stânga sus, axa `i` fiind verticală şi marcată de sus în jos, iar axa `j` este orizontală şi este marcată de la stânga la dreapta;

- `axis xy` – configurează axele în modul `cartezian`, mod implicit;
- `axis equal` – unităţile de măsură pe cele două axe au aceeaşi mărime;
- `axis image` – la fel cu `axis equal`, cu excepţia că figura este restrânsă la domeniul datelor;
- `axis square` – pune unităţile de măsura pe cele două axe de coordoante astfel încât figura să fie prezentată într–un pătrat;
- `axis normal` – rearanjează axele în forma implicită;
- `axis off` – elimină axele de coordoante;
- `axis on` – inserează axele de coordoante;
- `title('text')` – afişează textul specificat în partea de sus a graficului;
- `legend('str1','str2',...)` – afişează legenda prin care se efectuează corespondenţa dintre fiecare tip de curbă a graficului şi textele specificate prin `str1`, `str2` etc.;
- `legend('str1','str2',...,p)` – afişează legenda de tipul precizat în poziţiile respectiv înafara axelor (`p=-1`), în interiorul axelor, dar pe cât e posibil să nu acopere figurile trasate (`p=0`), în partea dreaptă sus (`p=1`), care este poziţia implicită, în partea stângă sus (`p=2`), în partea stângă jos (`p=3`), în partea dreaptă jos (`p=4`);
- `legend off` – elimină legenda;
- `xlabel('text')` – etichetează axa abscielor cu textul precizat;
- `ylabel('text')` – etichetează axa ordonatelor cu textul precizat;
- `text(x,y,'text')` – inserează textul precizat prin `text`, în punctul de coordonate `(x,y)`, iar dacă `x` şi `y` sunt vectori (de aceeaşi lungime), atunci inserează textul respectiv în toate punctele de coordoante (x_i, y_i), $i = 1, 2, \ldots$, pe când dacă textul este dat printr-o matrice cu acelaşi număr de linii cu cel al lungimilor vectorilor `x` şi `y`, atunci textul din linia `i` este inserat în punctul de coordonate (x_i, y_i), $i = 1, 2, \ldots$
- `gtext('text')` – afişează graficul, împreună cu cursorul plus, unde urmează să fie inserat textul, cursorul putând fi poziţionat cu ajutorul *mouse*-ului sau tastele cu săgeţi, după care prin apăsarea oricărei taste (*mouse* sau *keyboard*), se obţine inserarea textului;

- `grid` – adaugă pe grafic o reţea rectangulară;
- `grid off` – grafic fără reţea rectangulară;
- `zoom` – permite mărirea unei părţi a figurii pentru urmărirea anumitor detalii.

Se fixează cursorul *mouse*-ului pe poziţia în care suntem interesaţi a fi mărită. Prin *dublu–click* pe butonul din stânga se produce o mărire a zonei de două ori, iar prin *dublu–click* pe butonul din dreapta se produce o micşorare a zonei de două ori. Dacă se păstrează apăsat butonul din dreapta şi se deplasează cursorul, se obţine un dreptunghi, care prin eliberarea butonului *mouse*–ului, va umple întreaga figură.

Prin instrucţiunile

```
zoom off
zoom out
```

se dezactivează instrucţiunea `zoom`, respectiv se revine la dimensiunile iniţiale ale figurii.

Comenzile de tip `axis` se impun a fi inserate după instrucţiunea `plot`.

Menţionăm că textele introduse pe grafic admit sintaxa sistemului LaTeX simplificat.

Programul 1.7.7. Vom considera un exemplu de reprezentare grafică prin *mascarea* unor părţi a graficului. Să reprezentăm grafic cu linie continuă pe intervalul `[0,6]` funcţia care coincide cu $\sin(\pi x)$, când valorile acestei funcţii sunt pozitive, iar în rest să fie zero. Pe aceaşi figură, să reprezentăm prin linie întrerupă şi $\sin(\pi x)$.

```
m = input('m:');
h = 1/m; x = 0:h:6;
y = sin(pi*x); Y=ge(y,0).*y;
plot(x,y,'k--',x,Y,'k-')
```

Pentru `m=200`, graficul este cel din Figura 1.7.

1.7.3 Instrucţiunile `polar` şi `ezpolar`

Instrucţiunea `polar` este analogă cu instrucţiunea `plot`, numai că este considerată reprezentarea curbei în coordoante polare, iar instrucţiunea `ezpolar` are sintaxa

```
ezpolar('f',[a,b])
```

şi are ca efect reprezentarea grafică a funcţiei în coordonate polare, dată prin expresia algebrică `f`, pe intervalul `[a,b]`. Al doilea parametru este opţional. Valoarea implicită este `[0,2`π`]`.

De exemplu, comanda

```
ezpolar('1+cos(theta)')
```

va produce graficul din Figura 1.8. Mai remarcăm faptul că parametrul `f` poate fi numele unei funcţii Matlab predefinite sau a utilizatorului.

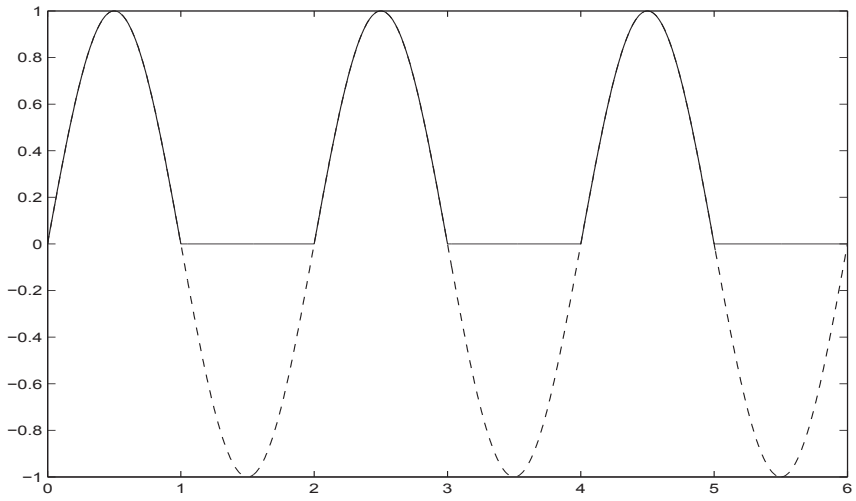

Figura 1.7: $y = \sin(\pi x)$

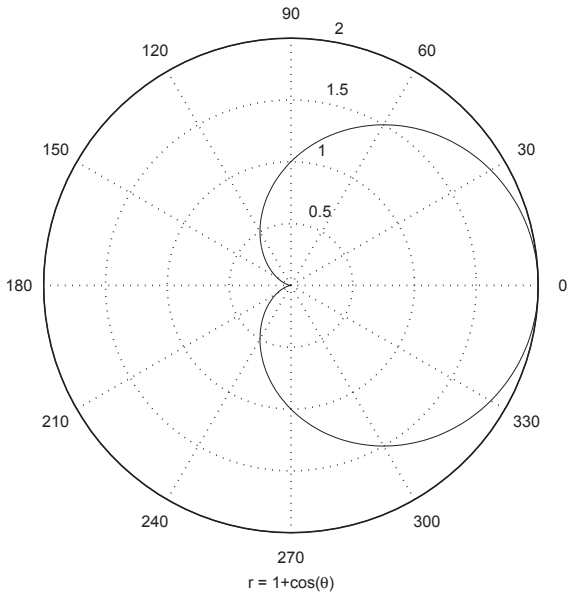

Figura 1.8: $r = 1 + \cos(\theta), \quad \theta \in [0, 2\pi]$

Programul 1.7.8. Următorul program

```
clf; t=0:0.01:pi;
r=sin(3*t).*exp(-0.3*t);
polar(t,r,'k-'), grid
```

are ca efect graficul din Figura 1.9.

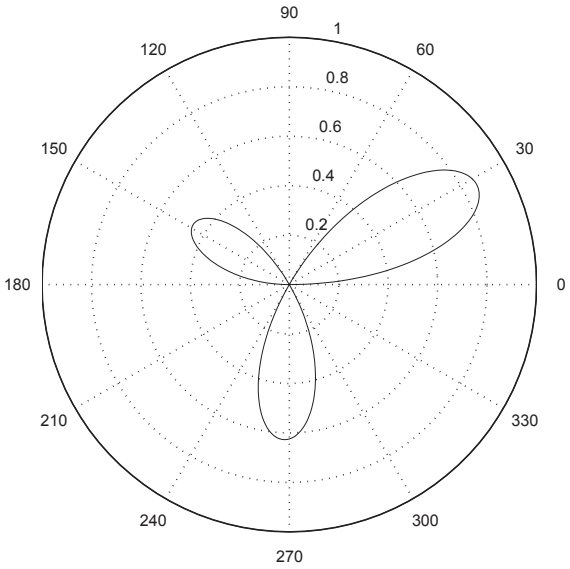

Figura 1.9: $r = \sin(3t)\,\mathrm{e}^{-0.3t}$

Instrucţiunea `colormap`

Sistemul Matlab dispune de un set de gestiune a culorilor în reprezentările grafice. Intrucţiunea `colormap` permite definirea unui set propriu pentru gestiunea culorilor. Lansarea unei instrucţiunii se face prin comanda

```
colormap(map)
colormap('default')
```

A doua comandă setează setul de culori la forma standard (implicită).

Parametrul `map` este o matrice cu trei coloane, fiecare linie conţinând elemente numere din intervalul `[0,1]`, prin care se defineşte câte o culoare. Modul de definire a culorii, după această regulă, se numeşte `RGB` (Red-Green-Blue), şi specifică în cadrul culorii, respectiv intensităţile culorilor roşu, verde şi albastru.

Sistemul Matlab dispune de asemenea de unele seturi de culori, care pot fi selectate şi specificate de utilizator. O parte din acestea sunt obţinute prin:

```
colormap spring
```

generează nuanţe de culori între violet (magenta) şi galben;

```
colormap summer
```

generează nuanţe de culori între verde şi galben;

```
colormap autumn
```

generează nuanţe de culori între roşu, trecând prin portocaliu spre galben;

```
colormap winter
```

generează nuanţe de culori între albastru şi verde;

```
colormap gray
```

generează nuanţe de culori gri.

1.7.4 Instrucţiunea `stairs`

Sistemul Matlab dispune de procedura `stairs` pentru reprezentarea grafică a funcţiilor în scară şi care acceptă formele instrucţiunii `plot`.

Instrucţiunea de forma

```
stairs(y)
```

reprezintă grafic funcţia în scară cu valorile precizate prin vectorul `y` de lungime `m`, iar abscisele sunt considerate numerele de la `1` la `m`. Dacă `y` este o matrice de tipul `(m,n)` vor fi reprezentate grafic `n` funcţii în scară, câte una pentru fiecare coloană a matricei `y`.

Dacă se consideră instrucţiunea de forma

```
stairs(x,y)
```

se va reprezenta grafic funcţia în scară cu valorile precizate prin vectorul `y` şi cu abscisele corespunzătoare date prin vectorul `x` (componentele lui `x` trebuie să fie crescătoare).

Dacă `y` şi `x` sunt matrice de acelaşi tip `(m,n)`, atunci se vor reprezenta grafic `n` funcţii în scară, câte una pentru fiecare din coloanele celor două matrice (elementele coloanelor matricei `x` trebuie să fie crescătoare). Dacă matricea `x` este un vector coloană cu `m` componente crescătoare, atunci funcţiile în scară corespunzătoare celor `n` coloane ale matricei `y` se consideră că au aceleaşi abscise precizate prin vectorul `x`.

Instrucţiunea de tipul

```
stairs(x,y,s)
```

permite prezentarea tipului liniei cu care se trasează graficul cu ajutorul parametrului `s` şi care are sintaxa cea precizată la instrucţiunea `plot`.

Prin urmare, putem spune că toate variantele folosite în cadrul instrucţiunii `plot` au corespondente pentru instrucţiunea `stairs`.

Programul 1.7.9. Fie *funcţia lui Haar*, numită în analiza wavelet *funcţie wavelet mamă*

$$\psi(x) = \begin{cases} 1, & \text{dacă } 0 \leqslant x < \dfrac{1}{2}, \\ -1, & \text{dacă } \dfrac{1}{2} \leqslant x < 1, \\ 0, & \text{în rest.} \end{cases}$$

Folosind funcţia ψ se defineşte familia de *funcţii wavelet*

$$\psi_{j,k}(x) = 2^{\frac{j}{2}} \psi\left(2^j - k\right)$$

unde j şi k se numesc *indice de dilatare*, respectiv *indice de translatare*. Dacă se are în vedere formula de definiţie pentru funcţia ψ, putem scrie

$$\psi_{j,k}(x) = \begin{cases} 2^{\frac{j}{2}}, & \text{dacă } \dfrac{1}{2^j}k \leqslant x < \dfrac{1}{2^j}k + \dfrac{1}{2^{j+1}}, \\ -2^{\frac{j}{2}}, & \text{dacă } \dfrac{1}{2^j}k + \dfrac{1}{2^{j+1}} \leqslant x < \dfrac{1}{2^j}k + \dfrac{1}{2^j}, \\ 0, & \text{în rest.} \end{cases}$$

Vom scrie un program care să reprezinte grafic funcţia f definită segementar cu ajutorul funcţiilor $\psi_{j,k}$, $k = \overline{r,s}$, unde j, r şi s sunt fixaţi ($r < s$).

Funcţia f este o funcţie în scară, având punctele de discontinuitate $\frac{k}{2^{j+1}}$, pentru $k = \overline{2r, 2s+2}$, cu valorile corespunzătoare acestor puncte, $2^{j/2}$, pentru k par, respectiv $-2^{j/2}$, pentru k impar.

Cu aceste precizări, avem următorul program Matlab pentru reprezentarea grafică a funcţiei f:

```
j = input('j=');
r = input('r=');
s = input('s (>r)=');
rs = 2*s-2*r+3;
fprintf('rs=2s-2r+3=%2d\n',rs);
y = input('rs valori alternative [1, -1 ... 1]:');
y = 2^(j/2)*y;
x = 1/2^j*[r:1/2:s+1];
stairs(x,y)
```

Execuţia programului, pentru `j=1, s=1, r=-2`, produce graficul din Figura 1.10.

Singurul inconvenient la execuţia acestui program este legat de instrucţiunea `input` prin care se iniţializează vectorul `y`. Când numărul funcţiilor $\psi_{j,k}$ prin care se defineşte funcţia f este mare, atunci `rs` este mare şi introducerea listei lui `y` este anevoioasă.

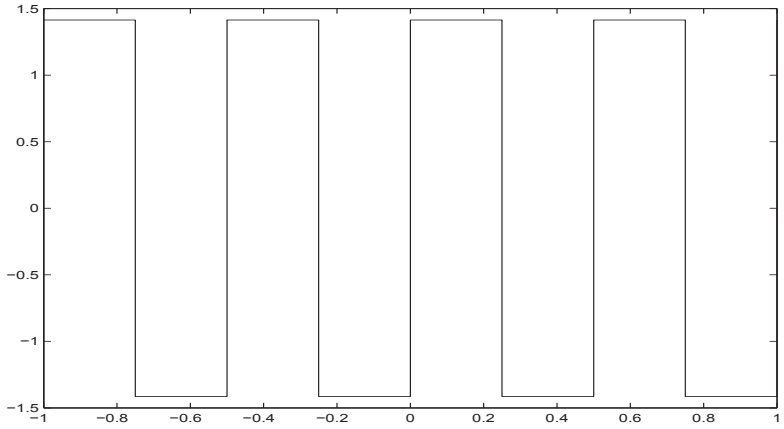

Figura 1.10: $f = (\psi_{1,-2}, \psi_{1,-1}, \psi_{1,0}, \psi_{1,1}, \psi_{1,2})$

O variantă a programului precedent, pentru care nu este necesară iniţializarea lui `y` de la tastatură, ar putea fi următorul program:

```
j = input('j=');
r = input('r=');
s = input('s (>r)=');
rs = 2*s-2*r+3;
y = (-ones(1,rs)).^[2*r:2*s+2];
y = 2^(j/2)*y;
x = 1/2^j*[r:1/2:s+1];
stairs(x,y)
```

Programul 1.7.10. Să reluăm reprezentarea grafică a funcţiilor wavelet Haar şi să scriem un program care să reprezinte pe aceeaşi figură două secvenţe de funcţii wavelet, pentru două valori consecutive ale indicelui de dilatare `j`, iar secvenţele alese să acopere intervalul $[-1, 1]$. Pentru aceasta trebuie ca `r`$=-2^j$, iar `s`$=2^j - 1$. Totodată să trasăm graficul primei secvenţe cu linie continuă şi punctele să le marcăm prin cerculeţe, iar pentru a doua secvenţă să alegem linie punctată pentru reprezentarea grafică, iar punctele să fie marcate cu steluţe în cinci colţuri:

```
clf, hold on, axis off
plot([-1.2 1.2],[0 0],'k:')
j = input('j=');
r = -2\^{}j; s=-r-1;
rs = 2*s-2*r+3;
y = (-ones(1,rs)).^[2*r:2*s+2];
y = 2^(j/2)*y;
x = 1/2^j*[r:1/2:s+1];
stairs(x,y,'ko-')
```

```
j = j+1; r = -2^j; s=-r-1;
rs = 2*s-2*r+3;
y = (-ones(1,rs)).^[2*r:2*s+2];
y = 2^(j/2)*y;
x = 1/2^j*[r:1/2:s+1];
stairs(x,y,'kp--')
```

Execuţia programului, pentru `j=0`, produce graficele din Figura 1.11. Se observă

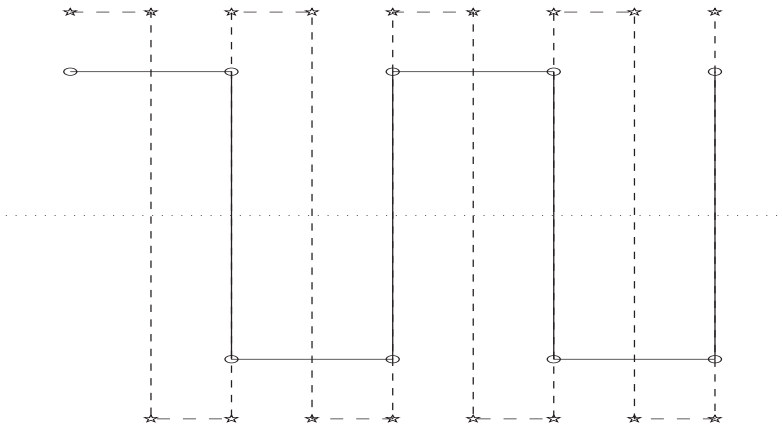

Figura 1.11: $(\psi_{0,-1}, \psi_{0,0})$, $(\psi_{1,-2}, \psi_{1,-1}, \psi_{1,0}, \psi_{1,1})$

că programul mai elimină axele sistemului Matlab, dar adaugă axa absciselor trasată punctat.

1.7.5 Instrucţiunile `bar` şi `barh`

Aceste două instrucţiuni, după cum spun şi denumirile, reprezintă anumite date numerice cu ajutorul barelor (batoanelor) verticale şi respectiv orizontale.

Forme ale acestor instrucţiuni sunt:

```
bar(y,w,'tip','color')
bar(x,y,w,'tip','color')
barh(y,w,'tip','color')
barh(x,y,w,'tip','color')
```

argumentele `w`, `'tip'` şi `'color'` sunt opţionale, dar când sunt specificate, trebuie să păstreze această ordine.

Parametrul `x` este un vector de lungime `m`.

Dacă `y` este un vector de aceeaşi lungime ca şi `x`, atunci vor fi reprezentate grafic în fiecare din punctele de abscise $\mathtt{x}_i$, $i = \overline{1, \mathtt{m}}$, câte o bară de înălţimile date prin $\mathtt{y}_i$,

$i = \overline{1,\mathtt{m}}$, care pot să fie şi negative. Dacă parametrul `x` lipseşte, atunci se consideră implicit că `x` are componentele ca fiind primele `m` numere naturale.

Dacă `y` este o matrice de tipul `(m,n)`, reprezentările sunt efectuate pentru fiecare din cele `n` coloane ale matricei, adică în fiecare punct $\mathtt{x}_i$, $i = \overline{1,m}$, de pe axa absciselor vor fi reprezentate câte `n` bare (batoane).

Parametrul `w` specifică lăţimea barelor, implicit fiind $\mathtt{w} = 0.8$, iar pentru $\mathtt{w} > 1$ barele se suprapun.

Dacă parametrul `tip=grouped`, care este valoarea implicită, atunci avem reprezentările precizate mai sus, iar dacă `tip=stacked` barele (batoanele) vor fi stivuite pe verticală.

Parametrul `color` specifică culoarea barelor (batoanelor) şi are una din următoarele valori: `r` (roşu), `g` (verde), `b` (albastru), `y` (galben), `m` (violet), `c` (ciclamen), `k` (negru), `w` (alb).

Tot ce s–a spus despre `bar` se poate spune şi pentru `barh`, numai că orientarea barelor este pe orizontală.

1.7.6 Instrucţiunea `subplot`

Sistemul Matlab are posibilitatea de a împărţi o fereastră, pentru reprezentare grafică, în `m`×`n` zone (ferestre), cu ajutorul instrucţiunii `subplot`.

Forma generală a instrucţiunii este

```
subplot(m,n,p)
```

unde `m`, `n` şi `p` sunt numere naturale, care exprimă faptul că fereastra iniţială se împarte în `m`×`n` ferestre aranjate pe `m` linii şi `n` coloane, iar reprezentarea grafică, la momentul execuţiei instrucţiunii, se va face în fereastra `p`, numerotarea ferestrelor făcându–se pe linii.

Programul 1.7.11. Reprezentarea grafică poate fi făcută şi cu ajutorul reprezentărilor parametrice ale curbelor.

Pentru reprezentarea grafică a unui cerc cu centrul în originea axelor de coordoante am putea folosi secvenţa de instrucţiuni:

```
r = input('raza=');
t = 0:0.01:2*pi;
x = r*sin(t); y=r*cos(t);
subplot(2,1,1), plot(x,y)
axis normal
subplot(2,1,2), plot(x,y)
axis square
```

Pentru `r=2`, graficul din partea stângă a Figurii 1.12, se obţine când nu s–a impus ca unităţile de măsură pe cele două axe de coordoanate să fie aceleaşi, iar graficul din partea dreaptă s–a obţinut după ce s–a înlăturat acest neajuns cu instrucţiunea `axis square`.

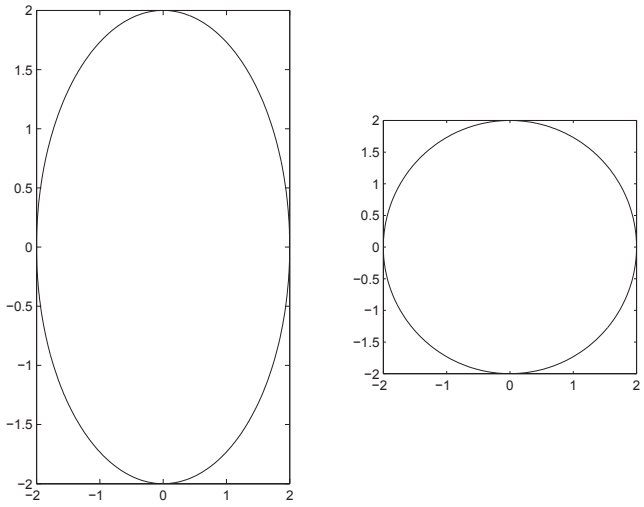

Figura 1.12: $x = r\cos t$, $y = r\sin t$, $r = 2$, $t \in [0, 2\pi]$

Programul 1.7.12. Pentru exemplificarea instrucţiunilor `bar`, `barh` şi `subplot`, facem din nou apel la o matrice magică de ordin `m` şi să reprezentăm în patru ferestre de tipul 2×2, prin bare verticale grupate şi prin bare verticale stivuite, valorile primelor două coloane ale matricei, respectiv prin bare orizontale grupate şi prin bare orizontale stivuite valorile ultimelor două coloane:

```
m=input('m:'); A=magic(m);
subplot(221), bar(A(:,[1 2]),'grouped')
title('Primele doua coloane - grupate')
subplot(222), bar(A(:,[1 2]),'stacked')
title('Primele doua coloane - stivuite')
subplot(223), barh(A(:,[m-1 m]),'grouped')
title('Ultimele doua coloane - grupate')
subplot(224), barh(A(:,[m-1 m]),'stacked')
title('Ultimele doua coloane - stivuite')
colormap summer
```

Reprezentarea grafică, pentru `m=4`, este dată în Figura 1.13.

1.7.7 Instrucţiunea `fplot`

Pentru reprezentarea grafică a unei funcţii definite cu ajutorul unei proceduri Matlab, se pot folosi una din următoarele variante ale instrucţiunii `fplot`:

```
fplot('numef',lim)
fplot('numef',lim,s)
fplot('numef',lim,tol)
```

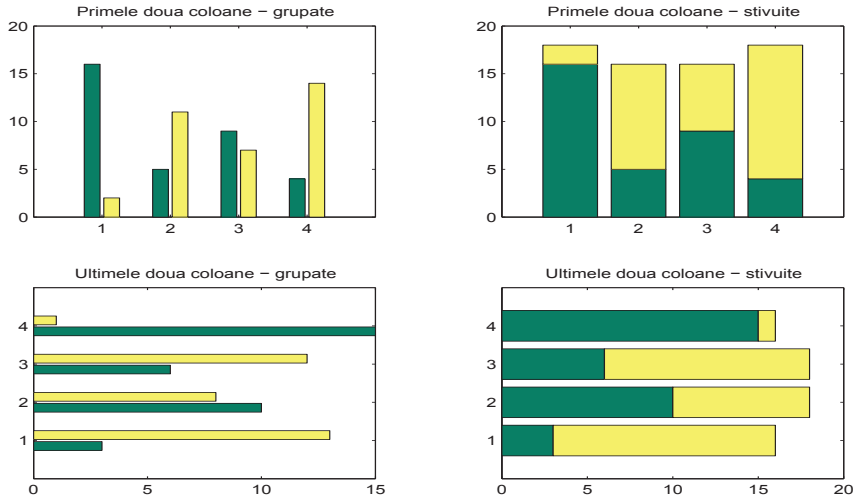

Figura 1.13: Bare verticale şi bare orizontale

```
fplot('numef',lim,tol,s)
fplot('numef',lim,n)
[X,Y] = fplot('numef',lim,...)
```

În toate cazurile se reprezintă grafic funcţia `numef`, cu limitele pentru axa abscise-lor specificate prin `lim=[xmin xmax]` sau `lim=[xmin xmax ymin ymax]`. Tipul liniei folosit în reprezentarea grafică este specificat prin parametrul `s`, ca şi în cazul instrucţiunii `plot`, iar eroarea relativă folosită în reprezentarea grafică este precizată prin parametrul `tol`, care are valoarea implicită `tol=2e-3`. Parametrul `n` specifică faptul că graficul se trasează cu ajutorul a `n+1` puncte, valoarea implicită a lui `n` este `n=1`, iar valoarea maximă a pasului nu depăşeşte $\frac{\texttt{xmax}-\texttt{xmin}}{\texttt{n}}$.

Remarcăm faptul că parametrii opţionali `s`, `tol` şi `n` pot fi luaţi în orice ordine.

Funcţia `numef` poate să întoarcă, pentru un vector `x`, un vector `y` de aceeaşi lungime ca şi `x` sau o matrice cu mai multe coloane şi număr de linii ca şi lungimea lui `x`. În primul caz, se va reprezenta grafic o singură curbă, iar în al doilea caz, pentru fiecare coloană a matricei `y` câte o curbă. Parametrul `numef` poate fi înlocuit şi cu un şir de caractere de tipul `'[f(x),g(x),...]'`, unde `f`, `g`,... sunt nume de funcţii Matlab, caz în care, pe aceeaşi figură, se vor reprezenta grafic funcţiile respective.

De exemplu, instrucţiunea

```
fplot('[sin(2*pi*x), cos(2*pi*x)]',[0 1])
```

va reprezenta grafic, pe aceeaşi figură, funcţiile $\sin(2\pi x)$ şi $\cos(2\pi x)$, pe intervalul $[0, 1]$.

Ultima formă a instrucţiunii `fplot` nu produce nici un grafic, ci păstrează punctele graficului în `X` şi `Y`, care pot fi apoi folosite efectiv, prin instrucţiunea `plot(X,Y)`, la reprezentarea grafică.

1.7.8 Instrucţiunea `ezplot`

Sintaxa instrucţiunii `ezplot` poate fi:

```
ezplot('f')
ezplot('f',[a,b])
ezplot('f',[a,b,c,d])
ezplot('x','y')
ezplot('x','y',[a,b])
```

Parametrul `f` reprezintă o expresie algebrică de una sau două variabile.

Dacă `f` este o expresie algebrică de o singură variabilă, atunci este reprezentată grafic funcţia definită de această expresie, pe intervalul `[a,b]`. Valoarea implicită a acestui parametru este $[-2\pi, 2\pi]$.

Dacă `f` este o expresie algebrică de două variabile, atunci este reprezentată grafic funcţia implicită definită prin `f=0`, pe domeniul $[-2\pi, 2\pi] \times [-2\pi, 2\pi]$, dacă se foloseşte prima formă de apel, pe domeniul `[a,b]`$\times$`[a,b]`, pentru a doua formă, respectiv pe `[a,b]`$\times$`[c,d]`, pentru a treia formă. Cele două variabile sunt considerate ca fiind ordonate lexicografic.

Ultimele două forme sunt folosite pentru reprezentările parametrice, `x` reprezentând abscisa, iar `y` ordonata. Argumentul acestor parametri variază în intervalul `[a,b]`, iar dacă acesta lipseşte, atunci se consideră intervalul implicit $[0, 2\pi]$.

De exemplu, instrucţiunea

```
ezplot('sin(3*t)*cos(t)','sin(3*t)*sin(t)',[0,pi])
```

produce graficul din Figura 1.14. Mai remarcăm faptul că parametrii `f`, `x` şi `y`, pot fi numele unor funcţii Matlab predefinite sau ale utilizatorului.

1.7.9 Instrucţiunea `fill`

Instrucţiunea `fill` este folosită pentru umbrirea interiorului unui poligon specificat prin vârfurile sale:

```
fill(x,y,c)
```

unde `x` şi `y` specifică vârfurile poligonului, iar `c` reprezintă culoarea dorită pentru interiorul poligonului.

Instrucţiunea poate fi utilizată pentru umbrirea unei părţi dintr–o figură.

Programul 1.7.13. Să scriem un program care să umbrească aria cuprinsă între axa absciselor, curba lui Gauss şi dreptele perpendiculare pe axa absciselor în punctele `a` şi `b` (`-3<a<b<3`).

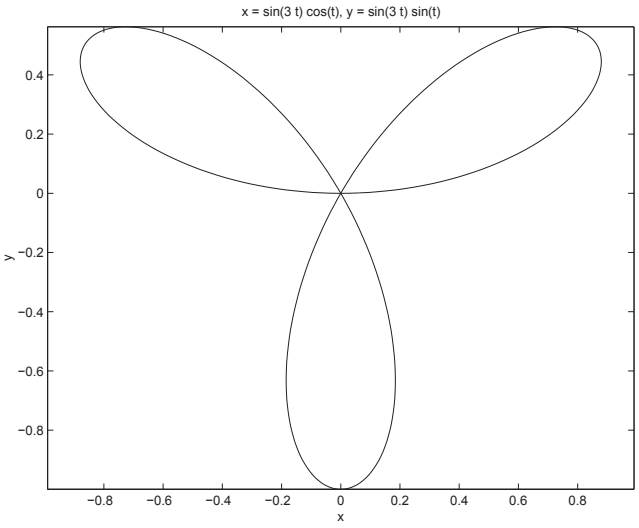

Figura 1.14: $x = \sin(3t)\cos(t)$, $y = \sin(3t)\sin(t)$, $t \in [0, \pi]$

Curba lui Gauss este dată prin funcţia

$$f(x) = \frac{1}{\sqrt{2\pi}} \mathrm{e}^{-\frac{x^2}{2}}, \quad x \in \mathbb{R}.$$

```
a = input{'a='}; b = input('b=');
xmin = -3; xmax = 3;
fplot{'1/sqrt(2*pi)*exp(-x^2/2)',[xmin xmax]}
x = a; y = 0;
t = a:0.01:b;
x = [x,t]; y = [y,1/sqrt(2*pi)*exp(-t.^2/2)];
x = [x,b]; y = [y,0];
fill(x,y,'c')
```

Pentru `a=-2` şi `b=1.5`, se obţine graficul din Figura 1.15.

1.8 Instrucţiuni de ciclare şi control

Ca şi în celelalte limbaje de programare evoluate, sistemul Matlab dispune de instrucţiuni, care permit repetarea unei secvenţe de instrucţiuni de un număr definit sau indefinit de ori, precum şi de instrucţiuni, care permit ieşirea din executarea secvenţială a instrucţiunilor unui program.