5. Графика в MATLAB

MATLAB разполага с голям брой разнообразни средства за графично представяне и визуализация на данни и на получените от работата на програмите резултати, в това число двумерна, тримерна графика и анимация. В тази тема са представени някои основни и най-често използвани графични команди и функции, както и подходи за графично представяне на данни.

Пълен списък на всички графични команди и функции се извежда с командата **help:**

```
>> help graph2d  % за двумерна графика

>> help graph3d  % за тримерна графика

>> help specgraph  % за специална графика
```

5.1. Обикновена двумерна графика

Функциите и командите, които се използват за цялостното оформяне на една двумерна графика се разделят на три групи:

- Функции за начертаване на самата графика: **plot, loglog, semilogx, semilogy, plotyy, polar;**
- Команди за управление на осите: **axis, grid, box, hold, subplot;**
- Функции за нанасяне на надписи върху графиките: **xlabel, ylabel, title, legend, text, gtext, texlabel.**

5.1.1. Функции за начертаване на графики

Най-често използваната функция за начертаване на графики в MATLAB е **plot**. Тя е разгледана и най-подробно. Функцията **plot**, както и повечето функции в MATLAB**,** може да се използва с различен брой аргументи:

plot(y) – начертава графиката на функция, чиито стойности са записани във вектора y. Като абсциси се използват индексите на отделните елементи на вектора.

plot(Y) – начертава едновременно графиките на няколко функции, представени чрез отделните стълбове на матрицата `Y`. За абсциси се използват първите индекси на елементите.

plot(x, y) – начертава графиката на функцията y = f(x). Тук `x` е вектор с абсцисите, а `y` – вектор с ординатите.

plot(x, Y) – начертава едновременно графиките на няколко функции на един и същи аргумент: `y1 = f1(x), y2 = f2(x), ….` Стойностите на функциите са записани в отделните стълбове на матрицата `Y`;

plot(x, y1, x, y2, x, y3, …) – аналогична на **plot**`(x, Y)`, но функциите са представени с двойки вектори: `x` – на абсцисите, `y1, y2, y3, …`– на ординатите;

plot(x, f1(x), x, f2(x), …) – аналогична на предишната, но стойностите на отделните функции за всички абсциси от вектора x се пресмятат директно в командата **plot**;

plot(x, y1, '*str1*', x, y2, '*str2*', ...) – тук аргументите се въвеждат по тройки. Третият аргумент от всяка тройка е низ, който се състои от един до три управляващи символа. С помощта на тези символи може да се променят подразбиращите се стойности за цвят и стил на линията и вида на маркера. Всички възможни стойности на управляващите символи могат да се видят като се подаде командата **help plot**.

Стойностите за управляващите символи са:

Цвят на линия

`r` – red – червен;
`g` – green – зелен;
`b` – blue – син;
`y` – yellow – жълт;
`m` – magenta – малинов;
`c` – cyan – синьо-зелен;
`k` – black – черен;
`w` – white – бял.

Стил на линия

– непрекъсната линия
: точкова линия
-. пунктирна линия
-- прекъсвана линия (тирета)

Вид на маркера

`o` – кръгче;
`x` – кръстче;
`+` – символ "+";
`*` – звездичка;
`s` – квадратче;
`d` – ромбче;
`v` – триъгълниче с върха надолу;
`^` – триъгълниче с върха нагоре;
`>` – триъгълниче с върха надясно;
`<` – триъгълниче с върха наляво.

plot(x, y1, '*str1*', '*param*', *value_of_par*, ...) – двойките или тройките аргументи могат да бъдат следвани от двойки опции (*parameter/values pairs*), които задават допълнителни свойства на линиите и маркерите:

'LineWidth', *n* – дебелина на линията в pt;
'MarkerEdgeColor', '*symbol*' – цвят на маркера (ако е отворен) или цвят на рамката на маркера (ако е затворена фигура); низът '*symbol*' е един от описаните символи за задаване на цвят;
'MarkerFaceColor', '*symbol*' – цвят за запълване на маркера (ако е затворена фигура);
'MarkerSize', *n* – размер на маркера в pt.

Важно е да се знае:

- Ако се чертаят едновременно няколко графики, без да е указан явно цветът на линиите, системата автоматично избира различен цвят за всяка графика.

- В аргумента '*str* ' управляващите символи от трите категории могат да се указват в произволен ред.
- Низът '*str*' може да съдържа 1, 2 или 3 управляващи символа, но не повече от един от дадена категория.
- Ако в аргумента '*str* ' няма символ за стил на линия, тя не се чертае. Означават се само точките с помощта на указания маркер.

Функциите **loglog, semilogx** *и* **semilogy** работят аналогично на функцията **plot**. Те могат да имат същите аргументи, като същесвената разлика е следната:
loglog – използва логаритмичен мащаб по двете оси x и y;
semilogx – използва логаритмичен мащаб по оста x;
semilogy – използва логаритмичен мащаб по оста y.

Функцията **polar**, както показва името □, начертава графиката на функцията r = r(φ) в полярни координати.

polar(*phi, r, 'str'***),**
където:
phi - вектор със стойностите на полярния ъгъл в радиани;
r – вектор със стойностите на радиуса;
'*str*' – низ, който задава стила на линията

5.1.2. Команди за управление на осите

Команда **axis** – управлява мащабирането и изгледа на координатните оси на текущата графика:
axis([*xmin xmax ymin ymax***])** – задава интервалите на изменение на променливите по координатните оси;
axis auto – връщане към стойности, избрани автоматично от системата;
axis equal – задава еднакъв мащаб по двете оси, така че графиката да не е деформирана;
axis square – прави координатната рамка (axis box) квадратна;
axis tight – скалите по двете оси съответстват на диапазона на изменение на данните;
axis normal – премахва всякакви ограничения по мащабирането;
axis ij – поставя координатното начало в горния ляв ъгъл; оста y е насочена надолу;
axis xy – възстановява нормалната ориентация на координатната система – началото е в долния ляв ъгъл, а оста y е насочена нагоре.

grid on - добавя мрежа към текущата графика;
grid off – премахва мрежата от текущата графика;

box on - добавя "кутия" към текущите оси,
box off - премахва "кутията" от текущите оси;

hold on - няколко графики, създавани от последователно изпълнени графични команди се изобразят в един и същи прозорец при запазване на всички свойства на осите. От този режим се излиза с командата **hold off**.

subplot - разделя един прозорец на по-малки прозорци, в които да се изобразят различни графики. Самата команда **subplot** не чертае графиките, а подготвя прозореца, в който ще попадне графиката, начертана от функцията, следваща непосредствено след командата **subplot.**

subplot(*m, n, k***)** - разделя основния прозорец на по-малки прозорци (панели), които можем да си представим като матрица с размери *m x n*. Третият аргумент *k* указва номера на прозореца, в който ще бъде изобразена графиката, начертана от функцията **plot** или друга графична функция, следваща непосредствено след командата **subplot(***m, n, k***).** Номерацията на прозорчетата е по редове отляво надясно и отгоре надолу. На схемата е показано разделянето на основния прозорец, след подадена команда **subplot(3, 3, k)**, където k може да има стойност от 1 до 9.

1	2	3
4	5	6
7	8	9

Няколко от панелите могат да се обединят в един, като за трети аргумент на функцията **subplot** се подаде вектор с номерата на обединяваните панели, например

```
>>subplot(3,3,[3 6 9]).
```

Някои от панелите могат да се използват само за въвеждане на пояснителен текст, след като в тях се премахнат осите с командата **axis('off').**

5.1.3. Функции за нанасяне на надписи върху графиките

xlabel('*string*') – нанасяне на надпис по оста x.
ylabel('*string*') – нанасяне на надпис по оста y.
title('*string*') – нанасяне на заглавие в горната част на графиката.
legend('*str1*','*str2*', …) – изобразява легенда в горната дясна част на графичния прозорец. Използва се, когато прозорецът съдържа няколко графики. За всяка графика в легендата е начертана права линия с цвета и стила на линията на графиката, а след нея следва съответният пояснителен текст, взет от поредния аргумент на функцията **legend**. Последователността на аргументите-низове трябва да отговаря на реда на построяване на съответните графики. Очевидно е, че броят на низовете трябва да е равен на броя на графиките.

legend(*str1*, *str2*, …, 'Location', *LOC*) – задава се явно положението на легендата. Някои от възможните стойности на аргумента *LOC* са:
'North' – в горната част на рамката;
'South' – в долната част на рамката;
'East' – в дясната част на рамката;
'West' – в лявата част на рамката;
'Best' – подбира се най-доброто място, така че да има възможно най-малко застъпване с графиката.
'NorthOutside', 'SouthOutside', 'EastOutside', 'WestOutside' и **'BestOutside'** – поставят легендата извън рамката (кутията).

Пълна информация за възможните аргументи дава командата **help legend**.

Програмно нанасяне на текст на произволно зададено от нас място в текущия графичен (или текстов) прозорец може да се извърши с помощта на командите **gtext** и **text:**

Командата **gtext** дава възможност визуално да се избере най-подходящото място за начало на надписа.

gtext('*string***')** – след като бъде въведена тази команда, позиционирайте курсора на мишката върху графичната област. Той ще се превърне в две пресичащи се линии – хоризонтална и вертикална. Изберете с този "кръст" желаното място и щракнете с левия бутон. На мястото веднага се появява текстът, зададен с аргумента '*string*'.

gtext(*Cell_array_of strings***)** – работи като горната команда, но вмъква няколко реда, всеки от които е записан като низ в отделна клетка на въведения масив от клетки.

gtext(...,'*PropertyName***',** ***PropertyValue*****,...)** – позволява да се задават стойности на някои свойства, например име и размер на шрифта и др.

За командата **text,** е необходимо предварително да се зададат (в самата команда) координатите на мястото, където да се въведе текстът. Тя обаче е по-универсална и е за предпочитане, когато трябва да се въвежда по-голям по обем текст в отделен подпрозорец, зададен с командата **subplot.** Общата форма на командата има следния вид:

```
text(x, y, 'string', 'FontName', '<Име на шрифта>',...
'FontSize', <Размер на шрифта в pt>)
```

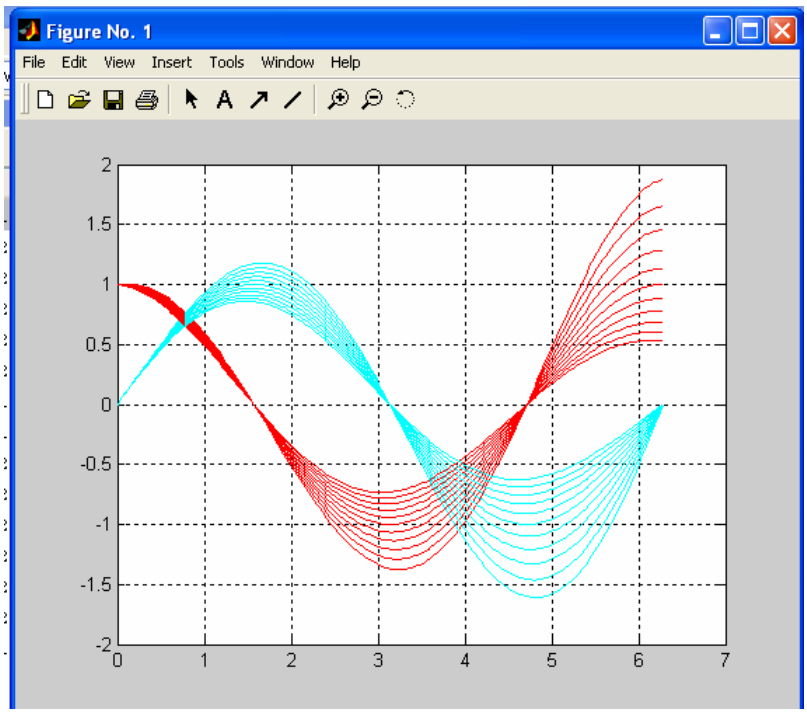
В този формат *x, y* са декартовите координати на началната позиция, от която ще се нанесе текстът *string* с указаното име и размер на шрифта. Тези координати могат да бъдат два типа:
• абсолютни – единици за измерване: inches, centimeters, points, pixels. Отчитат се от долния ляв ъгъл на координатните оси;
• нормализирани – отчитат се от горния ляв ъгъл на графичното поле и приемат стойности в диапазона от 0 до 1; това означава, че левият горен ъгъл на графичното поле има координати (0, 0), а десният долен – (1, 1).
По подразбиране се използват абсолютни координати в мащаба на начертаваната графика.

Пример 1: Да се начертаят графики на функцията
$y = e^{-ax} \cos x$
$y_1 = e^{-ax} \sin x$
за различни стойности на параметъра a .

Програма:

```
x = 0 : pi/100 : 2*pi;
for a = -0.1 : 0.02 : 0.1
    y = exp(-a*x).*cos(x);
    y1=exp(-a*x).*sin(x); % реализира се с поелементно умножение на
    вектори
    plot(x, y,'r-', x, y1, ('c-'))
    hold on  % всички графики в един прозорец
    grid on  % начертаване на мрежа
end
```

Резултатът от работата на програмата е представен на Фиг.5.1.



Фиг.51. Графики на функцията $y = e^{-ax}\cos x$ и $y_1 = e^{-ax}\sin x$

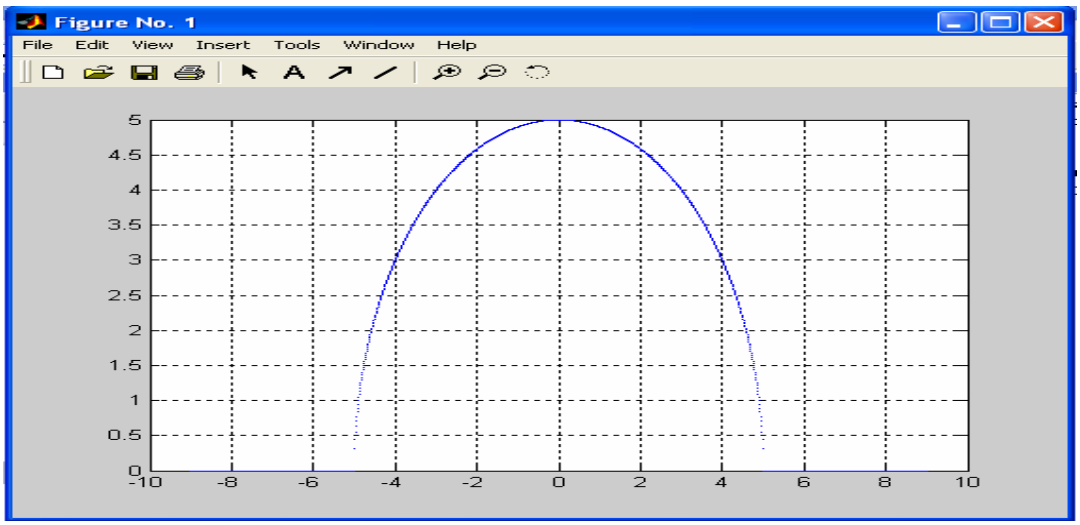
Пример 2:
Да се начертае графика на функцията *y=f(x),* която е зададена по следния начин

$$y = \begin{cases} 0, & \text{за } |x| > R \\ \sqrt{R^2 - x^2} & \end{cases}$$

Програма:

```
r=input('въведи радиус r ');
for x=-9:0.01:9,
    if abs(x)>r,
        y=0;
    else
        y=sqrt(r^2-x^2);
    end
    plot(x,y,'b--')
    hold on % всички графики в един прозорец
    grid on % начертаване на мрежа
end
```

Резултатът от работата на програмата е представен на Фиг.5.2.



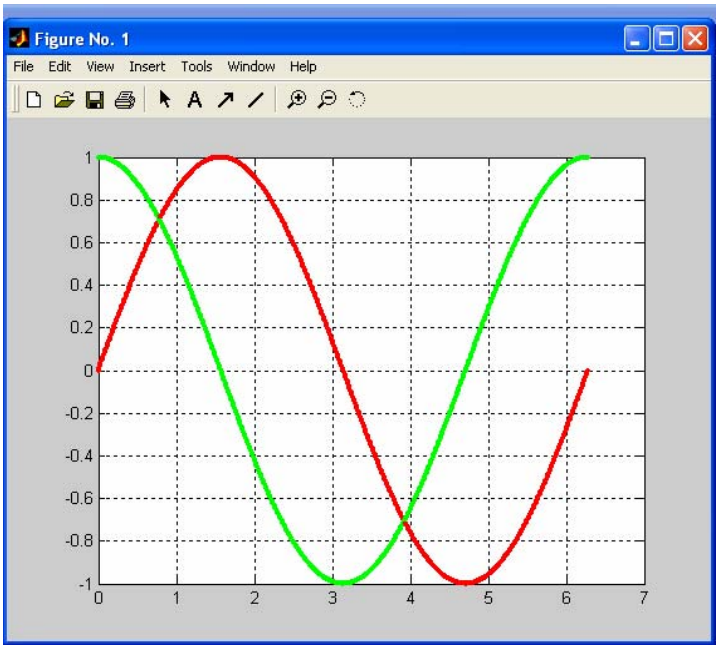
Фиг.5.2. графика на функцията от пример 2

Пример 3: Графики на функциите sin(x) и cos(x) за интервал за от 0 до 180^0.

Програма:

```
x = 0 : pi/300 : 2*pi;
y = sin(x);
y1=cos(x);
plot(x, y, 'r.-', x, y1, 'g.:')
grid on   % начертаване на мрежа
```

Резултатът от работата на програмата е представен на Фиг.5.3.



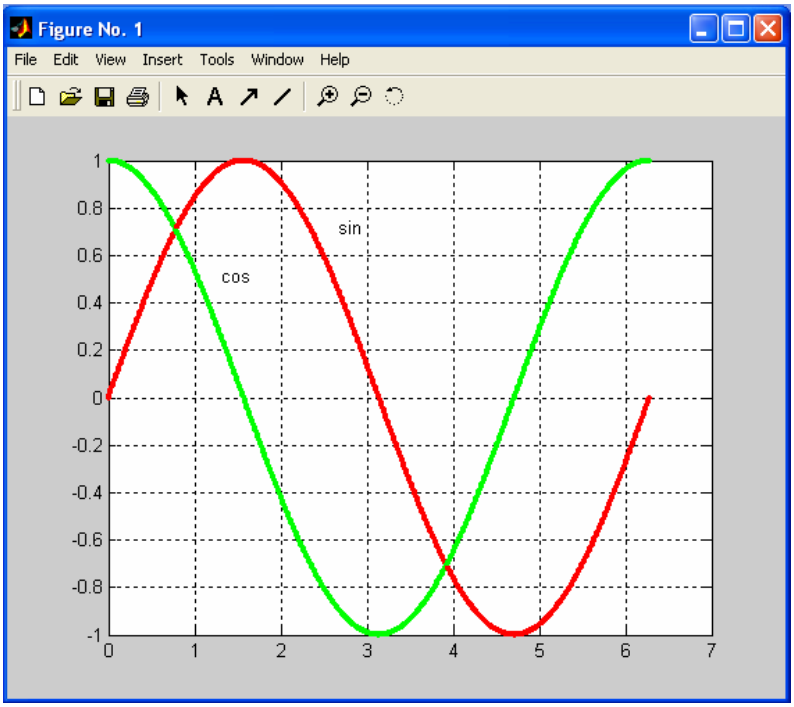
Фиг.5.3 Графики на функциите sin(x) и cos(x)

Пример 4: Графики на функциите sin(x) и cos(x) за интервал за от 0 до 180^0 с надписи

Програма:

```
x = 0 : pi/300 : 2*pi;
y = sin(x);
y1=cos(x);
plot(x, y, 'r.-', x, y1, 'g.:')
grid on  % начертаване на мрежа
gtext('sin')
gtext('cos')
```

Резултатът от работата на програмата е представен на Фиг.4.



Фиг.5.4. Графики на функциите sin(x) и cos(x) с надписи

5.2. Обикновена тримерна графика

За създаване на тримерна графика се използват и прилагат почти всички функции за управление на осите и за нанасяне на надписи, които се използват при създаване на двумерна графика.

Функцията **plot3** е аналог на **plot**. С нея се начертават пространствени криви.

plot3(x, y, z) – начертава графиката на пространствена крива, зададена в параметричен вид.

MATLAB разполага с функциите за изобразяване на повърхнини, които се описват с функцията $z = f(x, y)$. Функцията може да се зададе в аналитичен или табличен вид.

mesh – изобразяване с помощта на мрежа;
surf – изобразяване чрез непрекъсната повърхнина;
surfl – с допълнително осветяване (l от light – светлина).

Построяването на графиката е в следната последователност:

1) Пресмятат се матриците X и Y на координатите x и y на възлите на мрежата с помощта на функцията `meshgrid`:

```
[X, Y] = meshgrid(x, y)
```

Тук `x` и `y` са вектори с абсцисите и ординатите.

2) Пресмята се матрицата Z със стойностите на функцията във всички възли:

```
Z = F(X, Y)
```

В дясната част на това равенство трябва да е въведена конкретната функция с използване на поелементните оператори .*, ./, .^ !

3) Извиква се една от трите функции `mesh`, `surf` или `surfl` за построяване на графиката:

```
mesh(X, Y, Z); surf(X, Y, Z); surfl(X, Y, Z)
```

4) Извикват се някои от допълнителните функции за оформяне на графиката:

`colorbar` – извежда се цветова лента;

`colormap('colmap')` – цветова карта, която определя оцветяването на изображението;

Някои от възможните стойности на опцията '*colmap*' са `'gray','copper', 'pink', 'spring', 'summer'`. Пълният списък на цветовите карти се получава с командата >> **help graph3d**.

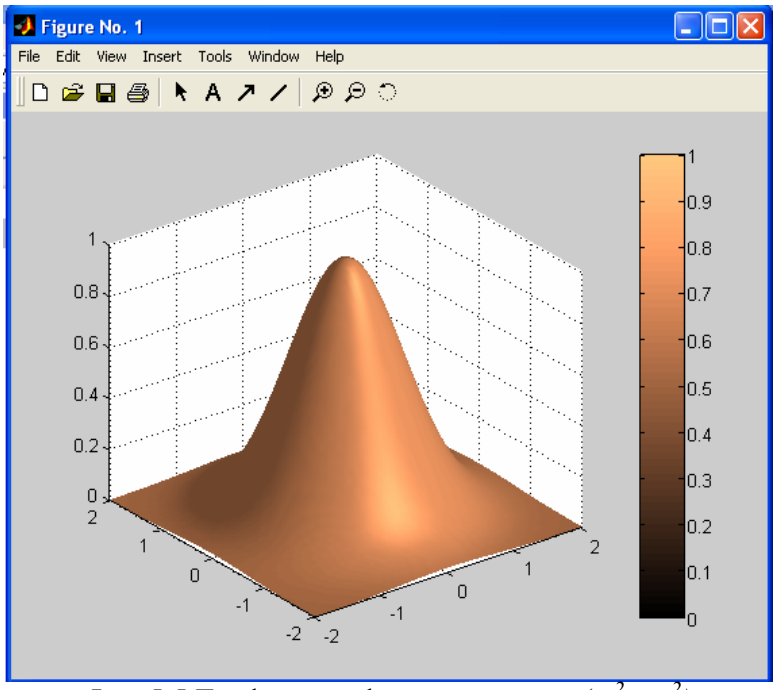
`shading interp` – плавен преход на сенките (командата се въвежда без аргументи).

Пример: Изобразяване на функцията $z = e(-x^2 - y^2)$

Програма:

```
[X, Y] = meshgrid(-2:0.05:2, -2:0.05:2);
Z = exp(-X.^2 - Y.^2);
surfl(X,Y,Z), shading interp
colormap('copper'), colorbar
```

Резултатът от работата на програмата е представен на Фиг.5.5.



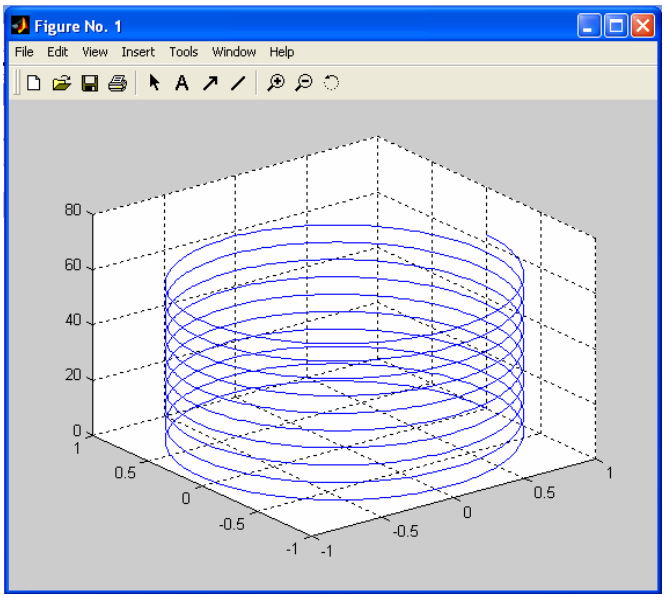
Фиг.5.5 Графика на функцията $z = e(-x^2 - y^2)$

Пример 2: Построяване на пространствена крива

Програма:

```
t = 0 : pi/100 : 20*pi;
x = cos(t); y = sin(t);
plot3(x, y, t);
grid on;
```

Резултатът от работата на програмата е представен на Фиг.5.6.



Фиг.5.6. Графика на пространствена крива