Глава 7. Графические возможности системы MATLAB

Для многих исследований и инженерных расчетов важным этапом является визуализация данных, поддержка которой осуществляется в MATLAB при помощи набора высокоуровневых команд, а также реализации интерфейса вывода графики, обеспечивающего тонкую настройку ее параметров. Команды графики MATLAB реализуют построение графиков кривых различных типов, элементы трехмерной графики, а также широкие мультимедийные возможности, дающие графикам большую наглядность и четкость при их представлении и демонстрации. В этой главе мы рассмотрим основы графики в MATLAB и мультимедийные возможности системы.

7.1. Визуализация аналитически заданных функций

Наиболее простой и в то же время существенной возможностью визуализации в MATLAB является построение графиков зависимостей, заданных аналитически. Одной из основных команд, обеспечивающих это, является команда `ezplot`. При ее выполнении в окне Command Window (Окно команд) или М-файле на экран выводится окно вывода, содержащее график визуализируемой зависимости. Команда `ezplot` имеет следующие представления:

```
ezplot(fun);
ezplot(fun, [min,max]);
ezplot(funx,funy);
ezplot(funx,funy, [tmin,tmax]);
```

где `fun, funx, funy` – строковое представление функции либо ее описатель;

`min, max, tmin, tmax` – соответственно минимальные и максимальные значения задаваемого пользователем интервала, на котором строится график.

При применении команды `ezplot(fun)` строится график функции одной переменной, на отрезке $-2\pi<x<2\pi$, выбираемом по умолчанию.

При выборе команды `ezplot(fun, [min,max])` функция строится на отрезке, ограниченном значениями `min` и `max`, определяемыми пользователем.

Рассмотрим пример построения графика синусоиды с применением команды `ezplot`.

Используя первое указанное ее представление, наберем в окне Command Window (Окно команд) следующий код:

```
ezplot('sin(x)');
```

На рис. 7.1 представлен график синусоиды, полученной при применении этой команды.

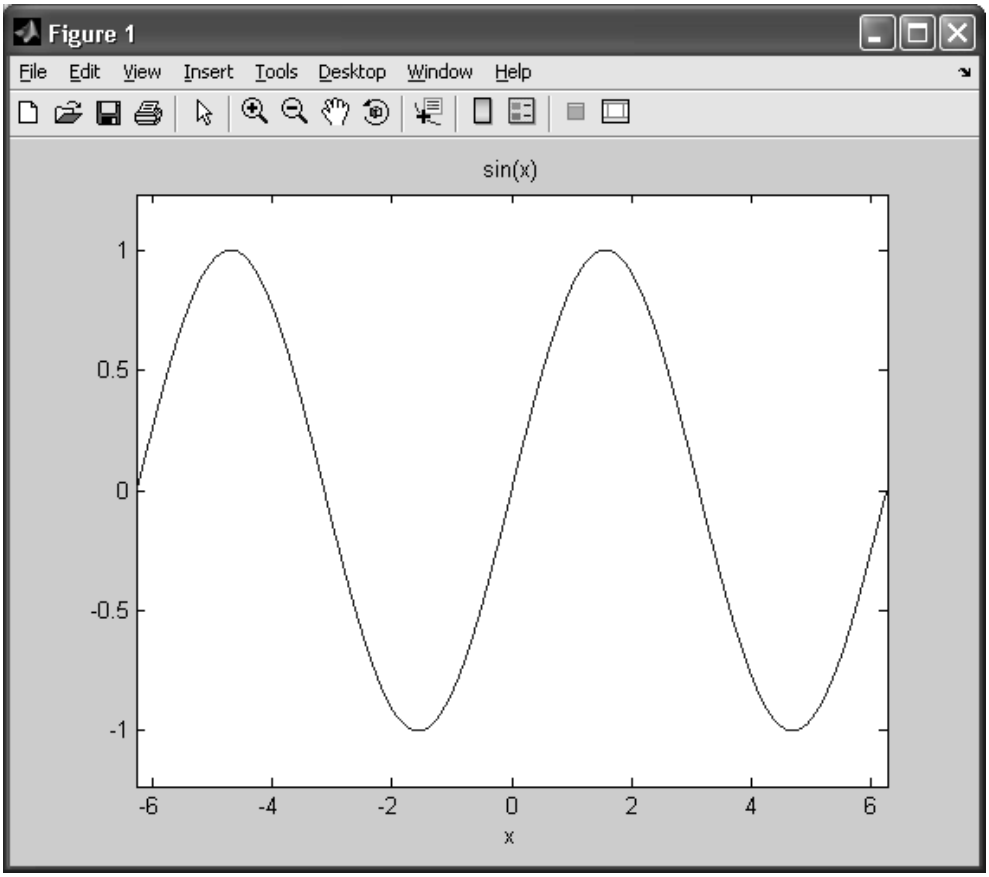

Рис. 7.1. График функции y=sin(x), заданной аналитически

В инженерной и расчетной практике возникают случаи, когда необходимо построить график быстро осциллирующей функции, при этом не задавая конкретных диапазонов изменения аргумента с целью выявления качественного характера ее поведения. Одним из интересных примеров такой функции является зависимость вида $y(x) = \sin\left(\frac{1}{x}\right)$. Для решения этой задачи, аналогично предыдущему, наберем в командной строке следующее выражение:

```
ezplot('sin(1/x)');
```

Построенный график имеет вид, приведенный на рис. 7.2.

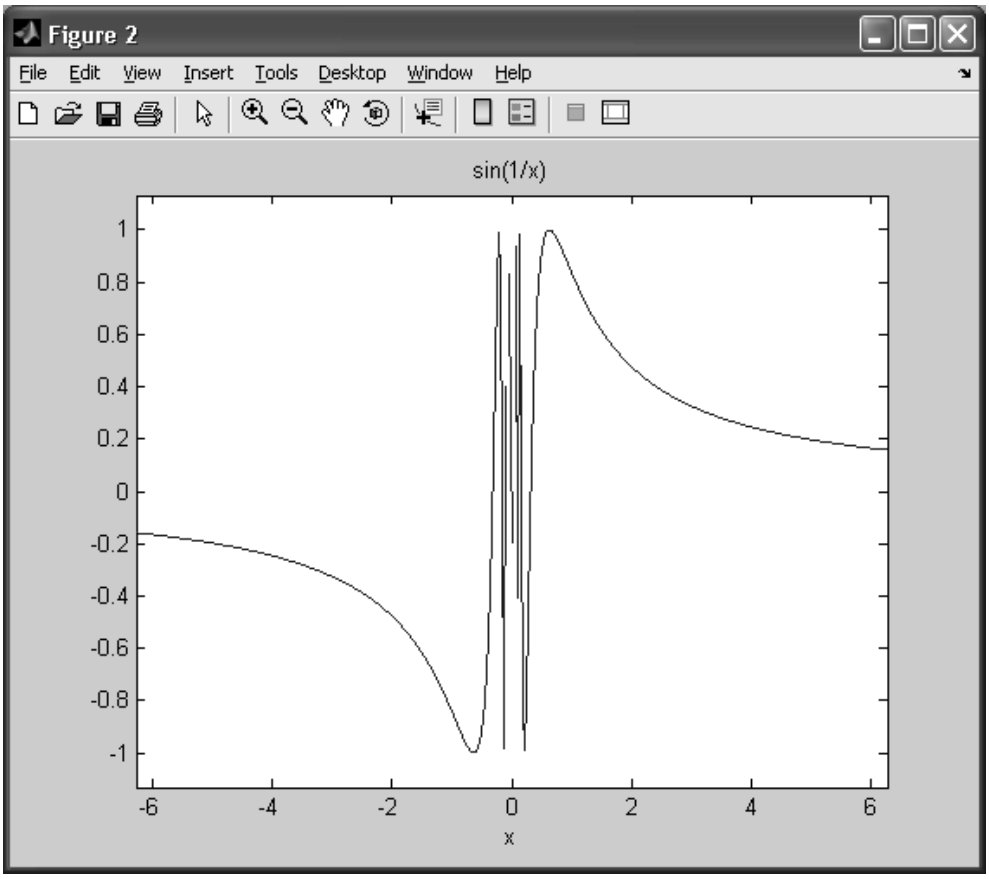

Рис. 7.2. График функции вида $y(x) = \sin\left(\frac{1}{x}\right)$

Приведем пример применения второго представления команды `ezplot`, учитывающий задание пользовательского интервала области определения функции.

Построим график функции $f(x) = \frac{1}{x^2 + 1}$, где $-5 \le x \le 5$. Наберем в окне Command Window следующую команду:

```
ezplot('1/(x^2+1)',[-5,5])
```

В результате получим следующий график, представленный на рис. 7.3.

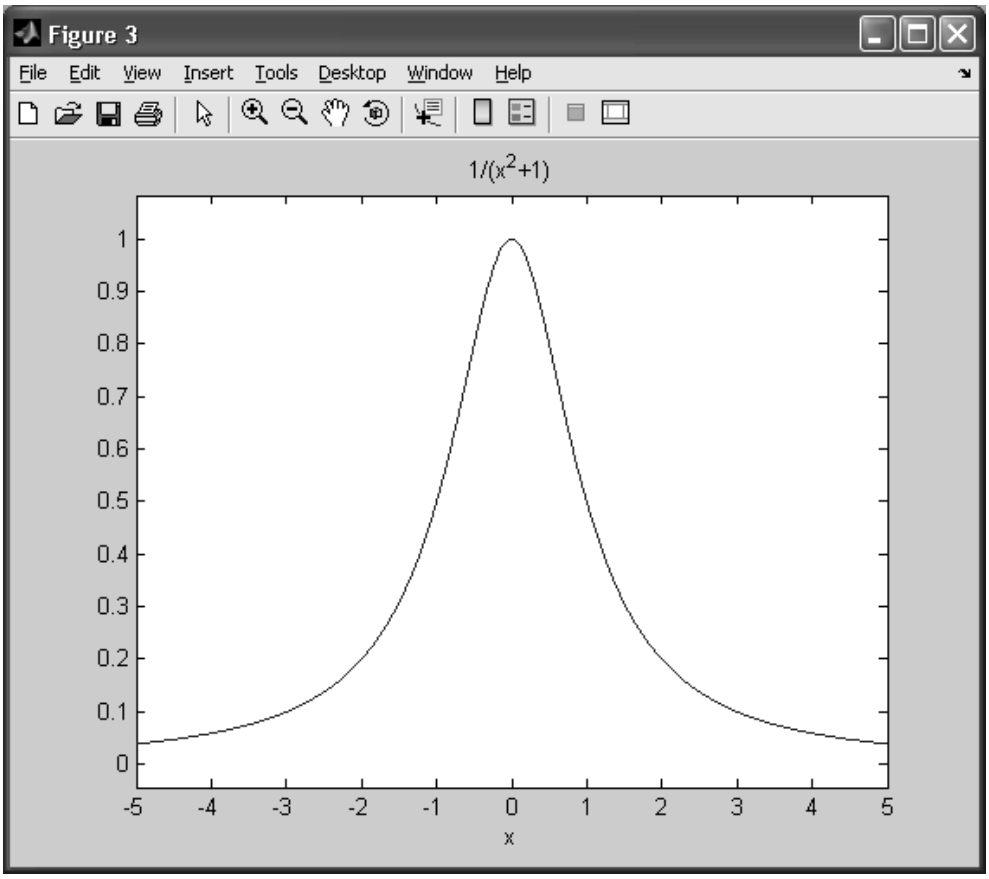

Рис. 7.3. График функции $f(x) = \dfrac{1}{x^2 + 1}$ на интервале $-5 \le x \le 5$

Из рис. 7.3. видно, что функция меняется достаточно плавно и убывает как при возрастании, так и при убывании аргумента, так что MATLAB успешно справился с данной задачей.

В отличие от ранее рассмотренных примеров, представления команды `ezplot` вида `ezplot(funx,funy), ezplot(funx,funy, [tmin,tmax])` применяются для изображения параметрических зависимостей. В последнем представлении параметры `tmin,tmax` определяют минимальное и максимальное значение интервала изменения параметра, и близки по смыслу величинам `min` и `max`, рассмотренным ранее.

Рассмотрим задание параметрической функции на примере. Пусть необходимо построить эллипс, уравнение которого задано параметрически и имеет вид:

$$\begin{cases} x(t) = a \cdot \cos(t) \\ y(t) = b \cdot \cos(t) \end{cases}, \; 0 \le t \le 2\pi$$

Пусть параметры эллипса принимают значения $a = 5$, $b = 3$.

Для построения эллипса наберем в Command Window (Окно команд) следующую команду:

```
ezplot('5*cos(t)','3*sin(t)',[0,2*pi])
```

В результате будем иметь график эллипса с большой полуосью, равной 5, и малой полуосью, равной 3.

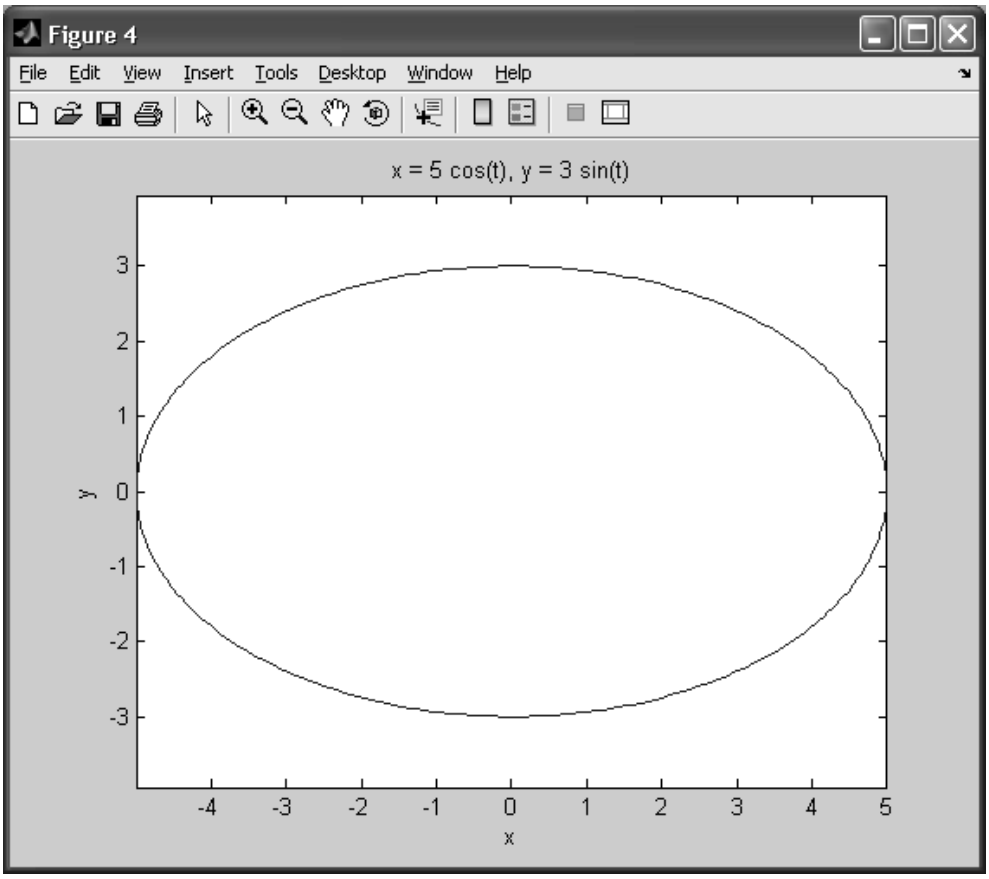

Рис. 7.4. График эллипса

ПРИМЕЧАНИЕ

> Аналогичный результат можно было бы получить, применив представление ezplot(funx,funy),в данном случае ezplot('5*cos(t)','3*sin(t)',[0,2*pi]). В этом случае параметр кривой изменяется по умолчанию от 0 до 2π.

Наряду с `ezplot` для различных случаев визуализации аналитически представленных зависимостей существуют близкие функции, состав которых представлен в табл. 7.1. Более подробно они будут рассмотрены в следующих пунктах главы, но в случае необходимости их описание можно будет найти в справке системы MATLAB.

Таблица 7.1. Функции MATLAB для визуализации аналитических зависимостей

Название функции	Описание действия функции
`fplot`	Построение графика функции по ее описателю. Во многом близка `ezplot`
`ezpolar`	Построение графика функции, заданной в полярных координатах
`ezplot3`	Построение кривой, параметрически заданной в трехмерном пространстве
`ezcontour`	Визуализация карты линий уровня функции двух переменных
`ezcontourf`	Визуализация карты линий уровня функции двух переменных с заливкой
`ezmesh`	Построение трехмерного сеточного полигона функции двух переменных
`ezmeshc`	Построение трехмерного сеточного полигона функции двух переменных и картой линий уровня
`ezsurf`	Построение закрашенной поверхности функции двух переменных
`ezsufc`	Построение закрашенной поверхности функции двух переменных и карты линий уровня

7.2. Построение графика функции одной переменной

В результате инженерных и математических расчетов имеет место накапливание больших объемов данных, который сложно анализировать без их наглядного присвоения. В большинстве случаев эти данные представляют собой одномерные числовые массивы большой длины, графическим представлением которых являются графики функции одной переменной или временные ряды. Для построения графика функции одной переменной в MATLAB реализована функция `plot`. Можно выделить два ее представления:

```
plot(y)
```

```
plot(x,y)
```

где `x,y` – массивы одинаковой длины

При использовании первого представления функции строится зависимость текущего элемента от его индекса в массиве, во втором – зависимость `y` от `x`.

Рассмотрим ее применение на следующем примере.

Пусть надо построить график синусоиды по ста точкам на интервале $-2\pi<x<2\pi$. Для этого необходимо выполнить следующие команды, представленные в листинге 7.1.

Листинг 7.1. Построение графика функции с применением команды plot

```
N=99; % Число точек
a=-2*pi; % Левый конец интервала
b=2*pi; % Правый конец интервала
h=(b-a)/N; % Шаг сетки аргументов
x=a:h:b; % Формирование сетки аргументов
y=sin(x);% Расчет значений функции
plot(x,y) % Построение графика функции
```

График функции имеет следующий вид (рис.7.5).

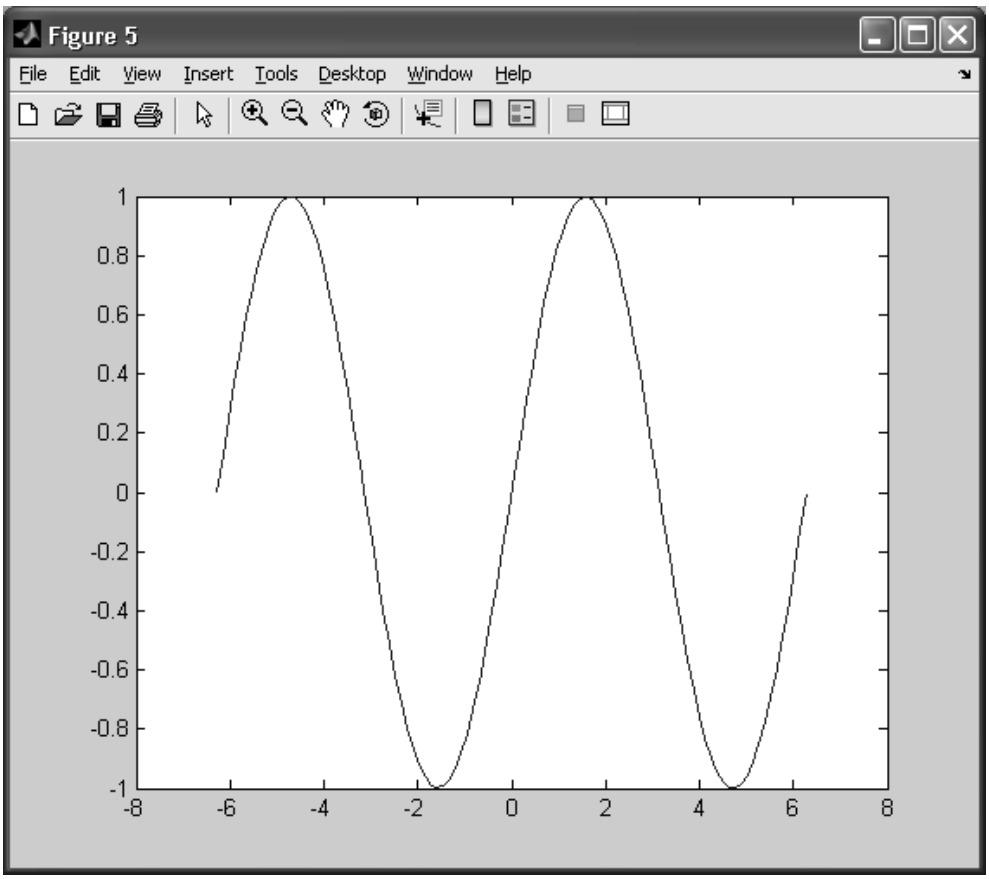

Рис. 7.5. Зависимость y=sin(x) с применением функции plot

ПРИМЕЧАНИЕ

Функцию `plot(x,y)` можно также применять для построения параметрических графиков, если ввести параметр, меняющийся в некотором диапазоне и вычислить массивы x и y как его зависимости.

7.3. Построение графика неявной функции

Наряду с графическим представлением таблично заданных исходных данных MATLAB предоставляет широкие возможности визуализации зависимостей, задаваемых различным образом. Неявной функцией будем называть функцию, заданную соотношениями между независимыми переменными, не разрешенными относительно последних. Уравнение неявной функции на плоскости имеет вид $F(x, y) = 0$ и ее график представляет собой геометрическое место точек, удовлетворяющих ему. Так, например, единичная окружность с центром в начале координат задается неявной функцией $F(x, y) = x^2 + y^2 - 1 = 0$

Визуализация неявно задаваемой функции осуществляется применением ранее рассмотренной команды `ezplot`. Ее представление при построении неявных функций имеет вид:

```
ezplot(equ)
```

```
ezplot(equ,[xmin,xmax],[ymin,ymax])
```

где `equ` – строковое представление левой части уравнения неявной функции;

`xmin, xmax, ymin, ymax` – соответственно минимальные и максимальные значения задаваемых пользователем интервалов, определяющих прямоугольник, в пределах которого строится график неявной функции.

ПРИМЕЧАНИЕ

> При применении первого представления команды `ezplot` для визуализации неявных функций необходимо учитывать, что она выполняет построение в пределах прямоугольника, определяемого интервалами $-2\pi<x<2\pi$ по оси Ox и $-2\pi<y<2\pi$ по оси Oy.

Рассмотрим некоторые примеры представления неявных функций посредством применения `ezplot`. Пусть необходимо построить график неявной функции вида $x^2e^{2y} - y^2e^{2x} = 0$ при условии, что область построения выбирается автоматически.

Применим первое представление ezplot в окне Command Window (Окно команд), записав в качестве аргумента левую часть уравнения:

```
ezplot('x^2*exp(2*y)-y^2*exp(2*x)')
```

Результат визуализации имеет вид, представленный на рис. 7.6 и представляет собой кривую с точками самопересечения.

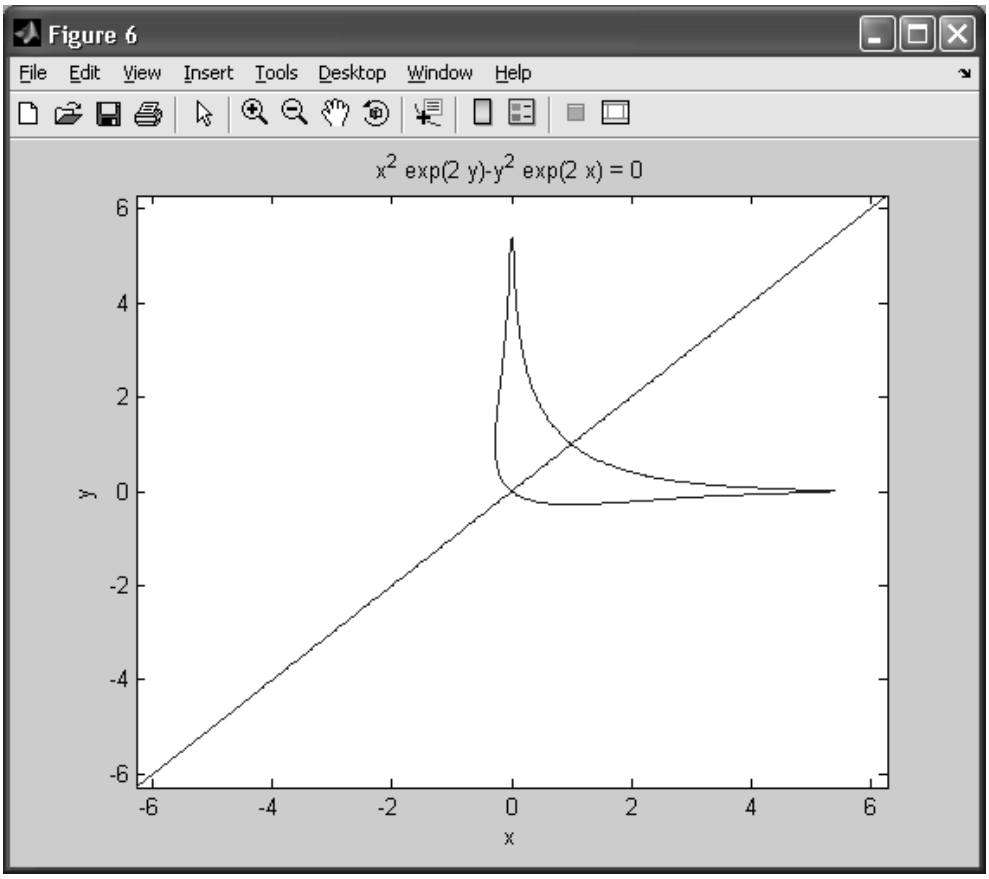

Рис. 7.6. График функции, заданной уравнением $x^2e^{2y} - y^2e^{2x} = 0$

Рассмотрим теперь в новом качестве ранее известный нам пример построения эллипса с большой полуосью *a* и малой полуосью *b* с такими значениями: *a* = 5, *b* = 3. Уравнение эллипса с заданными полуосями имеет вид:

$$\frac{x^2}{25} + \frac{y^2}{9} = 1$$

Перенеся единицу в левую часть, получим неявную функцию $F(x, y) = \frac{x^2}{25} + \frac{y^2}{9} - 1 = 0$.

При построении эллипса зададимся интервалом по оси Ox -5<*x*<5, по оси Oy зададим интервал -3<*y*<3. Тогда, применяя второе представление `ezplot`, запишем в командной строке выражение:

```
ezplot('x^2/25+y^2/9-1',[-5,5],[-3,3])
```

В результате выполнения команды получим график, представленный на рис. 7.7.

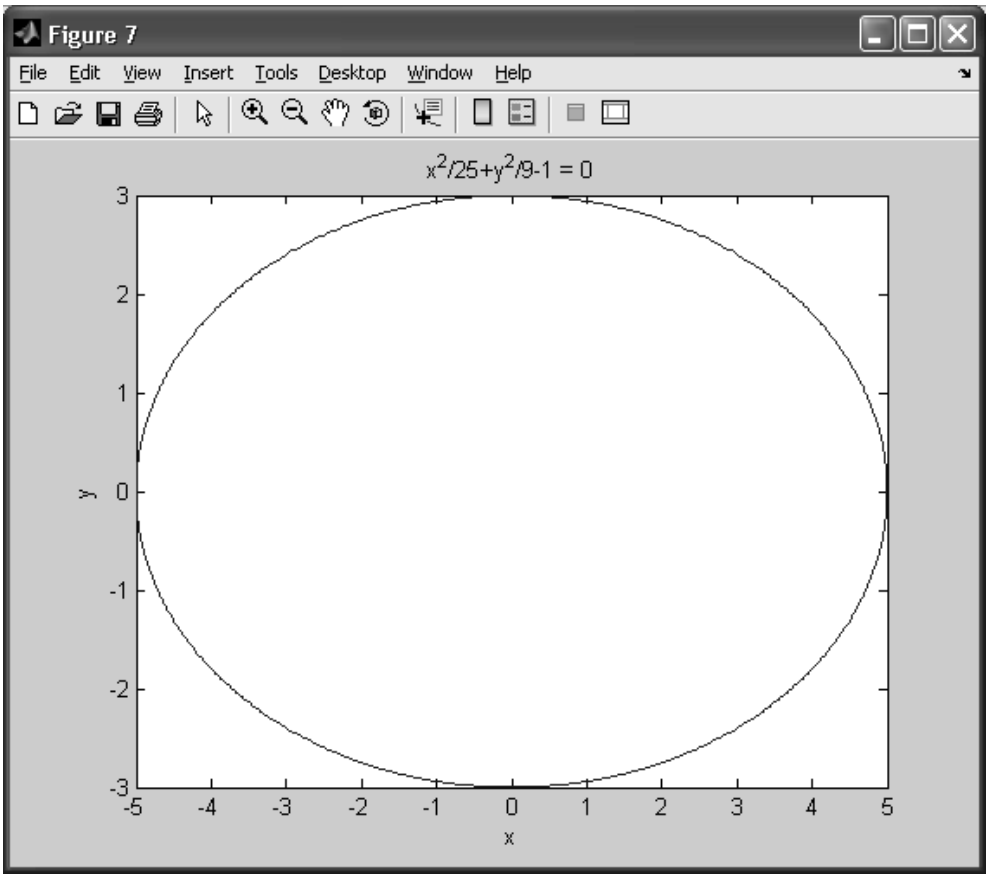

Рис. 7.7. График эллипса, заданного неявным уравнением

Из рис. 7.7 видно, что эллипс имеет несколько необычную форму, и больше напоминает окружность. Для получения корректного изображения эллипса применим масштабирование осей, набрав и выполнив команду:

```
axis equal
```

Более подробно возможности настройки графиков в MATLAB будут рассмотрены позже. В данном случае заметим, что масштабирование позволило изменить вид графика, что видно из рис. 7.8.

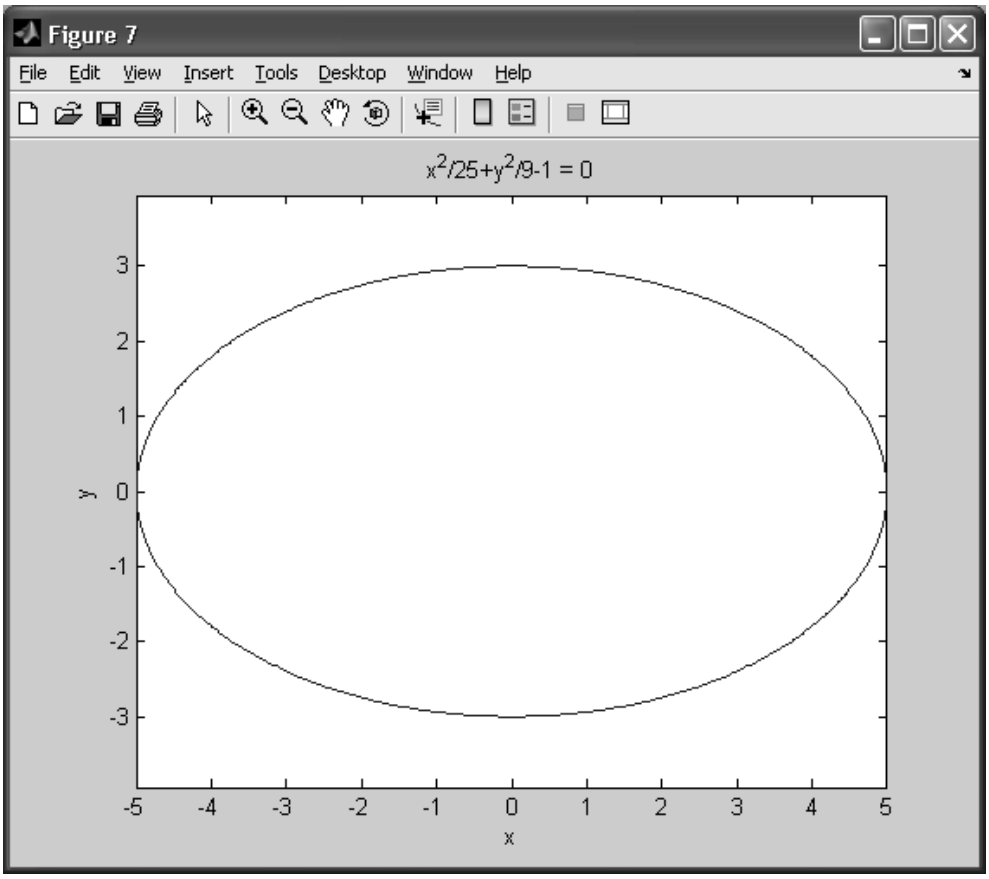

Рис. 7.8. Эллипс, полученный масштабированием осей графика

Таким образом, из рассмотренных примеров можно сделать вывод, что MATLAB обеспечивает качественную визуализацию различных зависимостей, в том числе и задаваемых неявно.

7.4. Построение кривой в трехмерном пространстве

Наряду с визуализацией двумерных графиков система MATLAB поддерживает визуализацию объектов трехмерной графики. Наиболее простым, и, в то же время, наглядным из них является кривая в трехмерном пространстве. Трехмерную кривую определяет соответствие скалярной переменной t некоторого вектора $\bar{r} = \{x(t), y(t), z(t)\}$. В ряде случаев кривая в трехмерном пространстве задается системой уравнений, определяющих связи между координатами ее точек. Для построения трехмерной кривой в MATLAB имеется команда `plot3`, имеющая следующее представление:

```
plot3(X,Y,Z)
```

где `X,Y,Z` – числовые массивы одинаковой длины, представляющие собой координаты точек пространственной кривой.

Альтернативным способом построения кривой является визуализация по ее аналитическому представлению с применением функции `ezplot3`, которая имеет представление:

```
ezplot3(funx,funy,funz)
```

```
ezplot(funx,funy,funz,[tmin,tmax])
```

где `funx, funy, funz` – строковое представление координат $x(t)$, $y(t)$, $z(t)$ от параметра t.

`tmin, tmax` – соответственно минимальные и максимальные значения определяемого пользователем интервала, в пределах которого изменяется параметр.

При первом представлении параметр по умолчанию принимает значение в интервале $0<t<2\pi$, в то время как во втором диапазон его значений определяется пользователем.

Рассмотрим построение кривой в пространстве на примере визуализации винтовой линии, определяемой параметрическим уравнением следующего вида:

$$\begin{cases} x(t) = \sin(t) \\ y(t) = \cos(t), \ 0 \le t \le 10\pi \\ z(t) = t \end{cases}$$

При реализации в MATLAB зададим диапазон изменения параметра t по 100 точкам. Затем рассчитаем зависимости координат $x(t)$, $y(t)$, $z(t)$ от параметра t в заданном диапазоне значений и представим их как векторы X, Y, Z. После этого применим функцию `plot3`, результатом которой является график винтовой линии. Программа построения графика приведена на листинге 7.2.

Листинг 7.2. Построение графика функции с применением команды `plot`

```
N=99; % Число точек
a=0; % Левый конец интервала параметра t
b=10*pi; % Правый конец интервала
h=(b-a)/N; % Шаг сетки аргументов
t=a:h:b; % Формирование сетки значений параметра
X=sin(t);% Расчет значений координаты кривой по оси Ox
Y=cos(t);% Расчет значений координаты кривой по оси Oy
Z=t;% Расчет значений координаты кривой по оси Oz
plot3(X,Y,Z) % Построение графика функции
```

График построенной линии имеет следующий вид (рис. 7.9).

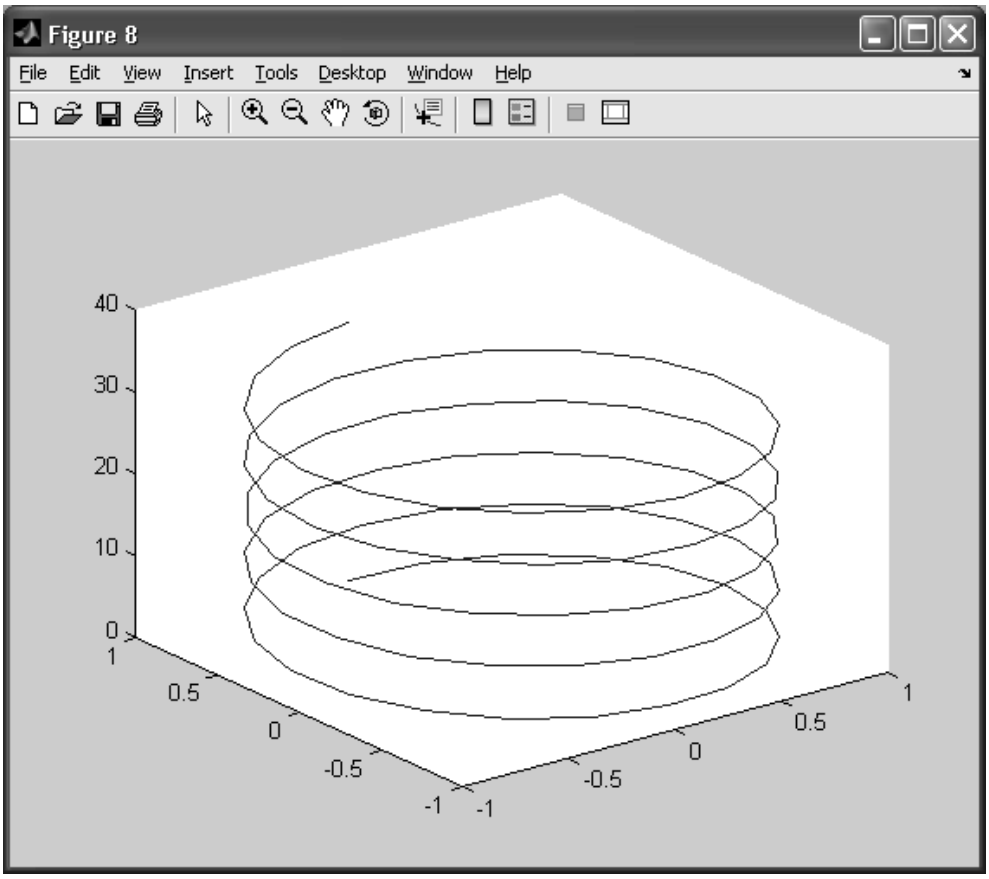

Рис. 7.9. График винтовой линии

Построим график параметрической функции с применением команды `ezplot3`. Пусть кривая задается уравнениями $x = y^2$, $y = z^2$. В данном случае по уравнениям можно сделать вывод, что кривая является результатом пересечения двух поверхностей. Параметризуем уравнения, введя новую переменную $z = t$. Тогда кривая будет описываться следующей системой уравнений:

$$\begin{cases} x(t) = t^4 \\ y(t) = t^2 \\ z(t) = t \end{cases}$$

Пользуясь функцией `ezplot3`, построим график функции в интервале изменения параметра $0<t<7$. Напишем и выполним в командной строке следующий ее вызов:

```
ezplot3('t^4', 't^2','t',[0,7])
```

Результатом выполнения данной команды будет график, представленный на рис. 7.10.

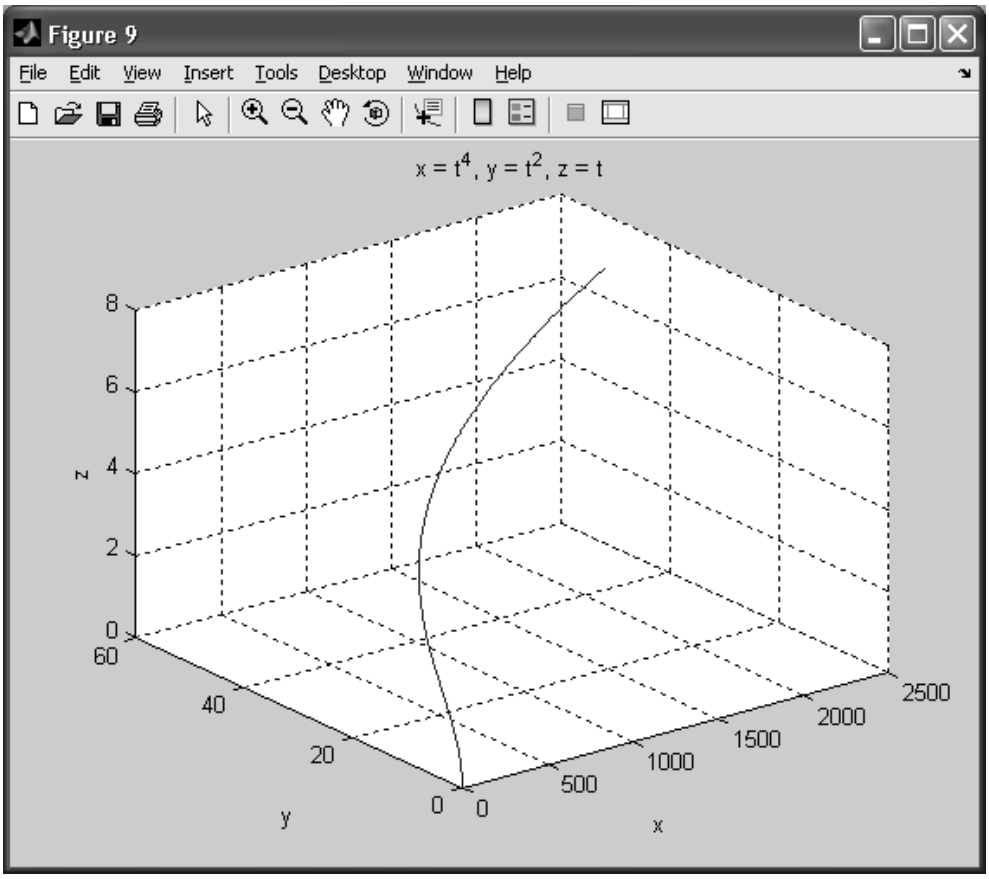

Рис. 7.10. Вид параметризованной кривой, заданной уравнениями $x = y^2$, $y = z^2$.

Из рис. 7.9, рис. 7.10 видно, что визуализация трехмерных кривых в MATLAB проходит корректно и точно вне зависимости от конкретных команд, применяемых для этого.

7.5. Визуализация поверхности в трехмерном пространстве

Наряду с построением кривых в трехмерном пространстве, Matlab располагает мощными средствами для визуализации сложных трехмерных объектов. Одним из них, наиболее наглядно описывающим результаты научных расчетов или динамику изменения состояния моделируемого в MATLAB объекта в зависимости от нескольких параметров, являются поверхности. Поверхность представляет собой геометрический образ функции двух переменных $z = f(x, y)$, областью определения которой является двумерная область, лежащая в плоскости Oxy декартовой системы координат. В MATLAB поверхности в большинстве случаев строятся следующим образом. Сначала задаются интервалы изменения переменных x и y. Затем формируется прямоугольная сетка по данным из числовых массивов, после чего

происходит расчет значений функции двух переменных в ее узлах. Непосредственно визуализация образа функции может проходить различными способами, которые будут рассмотрены ниже.

Формирование элементов дискретной сетки для вычисления массива значений функций двух переменных определяется командой `meshgrid`, имеющей следующее представление и формат выполнения

```
[X_write,Y_write]=meshgrid(x,y);
```

где `x,y` – массивы, описывающие интервалы изменения переменных x и y соответственно.

`X_write,Y_write` – матрицы, вычисляющиеся в результате работы функции `meshgrid`.

Рассмотрим структуру матриц `X_write` и `Y_write`. Все строки матрицы `X_write` являются копиями массива аргументов переменной x `x`, а все столбцы матрицы Y_write являются копиями массива `y`. Легко заметить, что выбирая из каждой матрицы элемент с одинаковыми индексами, мы получаем узел сетки аргументов для вычисления значений функции двух переменных. Благодаря возможности векторизации вычислений при работе с массивами данных, пользователь по данным из матриц `X_write` и `Y_write` может формировать массив значений функции двух переменных $z = f(x, y)$, после чего может провести ее визуализацию. Обратим внимание на то, что непосредственное применение векторизации в MATLAB вместо применения операторов цикла при реализации работы с данными существенно повышает производительность расчетов, что является критически важным при работе с большими объемами данных.

Простейшим способом построения поверхностей является применение ранее известной нам команды `plot3`. В этом случае поверхность рассматривается как множество пространственных кривых, каждая из которых определена на интервале, принадлежащем прямоугольной сетке и является сечением поверхности плоскостью, параллельной плоскости Oyz. Рассмотрим ее применение на примере построения графика функции Розенброка, определяющейся уравнением $z = f(x,y) = 100(y - x^2)^2 + (1 - x)^2$ в интервалах изменения переменных $-2 \le x \le 2$, $-1 \le y \le 3$.

Следуя сформулированному в начале п. 7.5 алгоритму визуализации поверхности, определим область изменения переменных как. Программный код в MATLAB для построения поверхности по последовательности пространственных кривых приведен в листинге 7.3.

Листинг 7.3. Реализация построения поверхности функции Розенброка

```
N=100; %Число точек по оси Ox
M=100; %Число точек по оси Oy
%Начало и конец интервала изменения переменной x
ax=-2
bx=2;
hx=(bx-ax)/N;
```

```
x=ax:hx:bx;
% Задание интервала изменения переменной y
%Начало и конец интервала изменения переменной y
ay=-1;
by=3;
hy=(by-ay)/N;
y=ay:hy:by;
%Формирование массивов сетки значений аргументов функции двух переменных
[X,Y]=meshgrid(x,y);
% Вычисление функции Розенброка в узлах сетки. Матрица Z – массив значений
функции в узлах сетки
Z=100*(Y.-X.^2).^2+(1-X).^2;
plot3(X,Y,Z)
```

На рис. 7.11 изображена поверхность, описывающая функцию Розенброка, представленная как совокупность пространственных кривых.

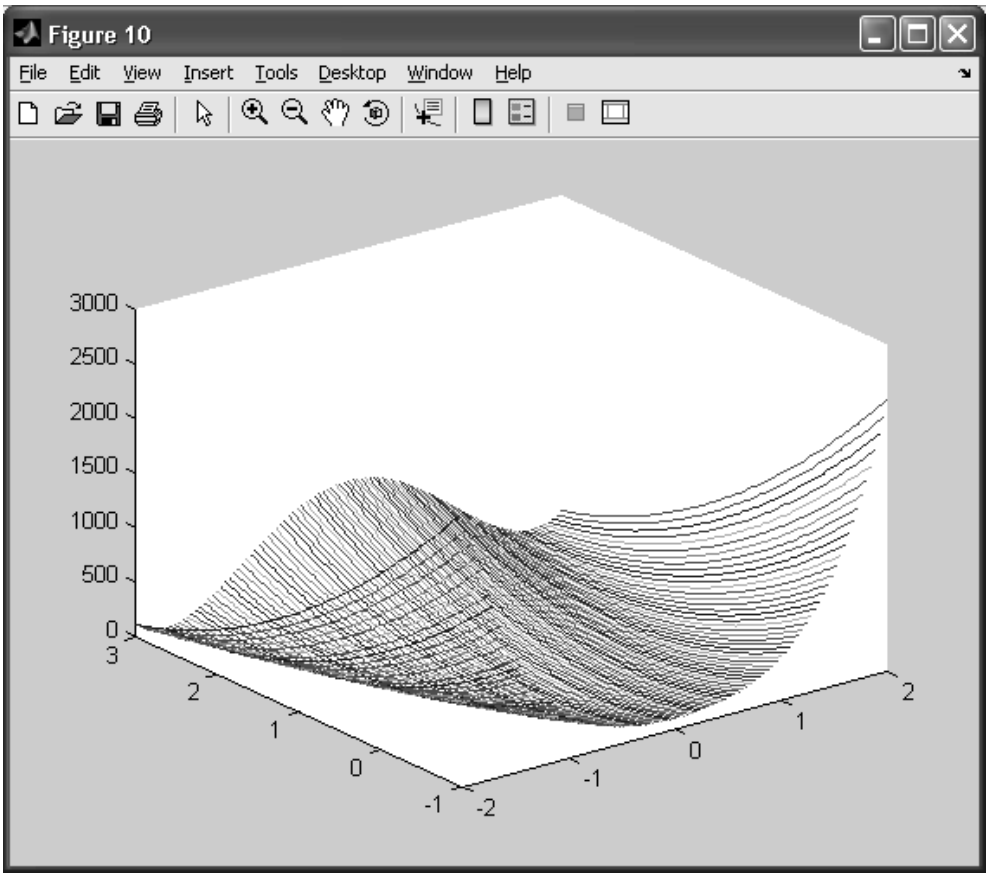

Рис. 7.11. Функция Розенброка. построенная с помощью команды plot3

Рассмотрим команды визуализации, позволяющие строить поверхности с более высокой степенью детализации и наглядности. Каждая из них имеет свои

особенности визуализации, благодаря чему пользователь располагает большим выбором способов графического представления своих данных.

Для изображения поверхностей в MATLAB служат функции `mesh` и `surf`. Функция `mesh` воспроизводит прозрачную сетчатую поверхность, функция `surf` строит затененную непрозрачную поверхность, цвет которой в каждой ее точке определяется значением функции двух переменных. Команды `mesh` и `surf` имеют следующее представление:

```
mesh(X,Y,Z)

mesh(Z)

surf(X,Y,Z)

surf(Z)
```

где `X,Y` – матрицы, содержащие информацию о сетке аргументов функции двух переменных;

`Z` – матрица значений функции двух переменных.

Также заметим, что MATLAB позволяет строить не только явно заданные поверхности, одним из которых является рассмотренная выше функция Розенброка, но и поверхности, заданные параметрически. Рассмотрим работу функций `mesh` и `surf` на ряде примеров.

Тором называется поверхность вращения, получаемая вращением окружности вокруг оси, лежащей в плоскости окружности и её не пересекающей.

Уравнение тора может быть задано параметрически в виде:

$$\begin{cases} x = (R + r \cdot \cos(u)) \cdot \cos(v) \\ y = (R + r \cdot \cos(u)) \cdot \sin(v) \\ z = r \cdot \sin(u) \end{cases}, \quad 0 \le u \le 2\pi, \quad 0 \le v \le 2\pi$$

где R — расстояние от центра окружности до оси вращения, r — радиус окружности.

Отметим, что область определения тора необходимо пересчитать с учетом параметрических уравнений. Для визуализации тора применим функцию `mesh`. Соответствующая программа представлена на листинге 7.3.

Листинг 7.3. Программа визуализации тора

```
% Определение интервалов переменных u и v и двумерной сетки, образующей
%область определения параметрической поверхности.
N= 50;% Число точек для интервала изменения переменной u
M=50; % Число точек для интервала изменения переменной v
u=0:2*pi/N:2*pi;
v=0:2*pi/M:2*pi;
[U,V]=meshgrid(u,v);
% Определение параметрических зависимостей для координат x,y,z
r=1; % радиус окружности
R=3; % расстояние от окружности до оси вращения
X=(R+r*cos(U)).*cos(V);
```

```
Y=(R+r*cos(U)).*sin(V);
Z=r*sin(U);
mesh(X,Y,Z);
axis equal % Масштабирование осей для корректного представления тора
```

Вид тора, построенного по данному листингу, приведен на рис. 7.12.

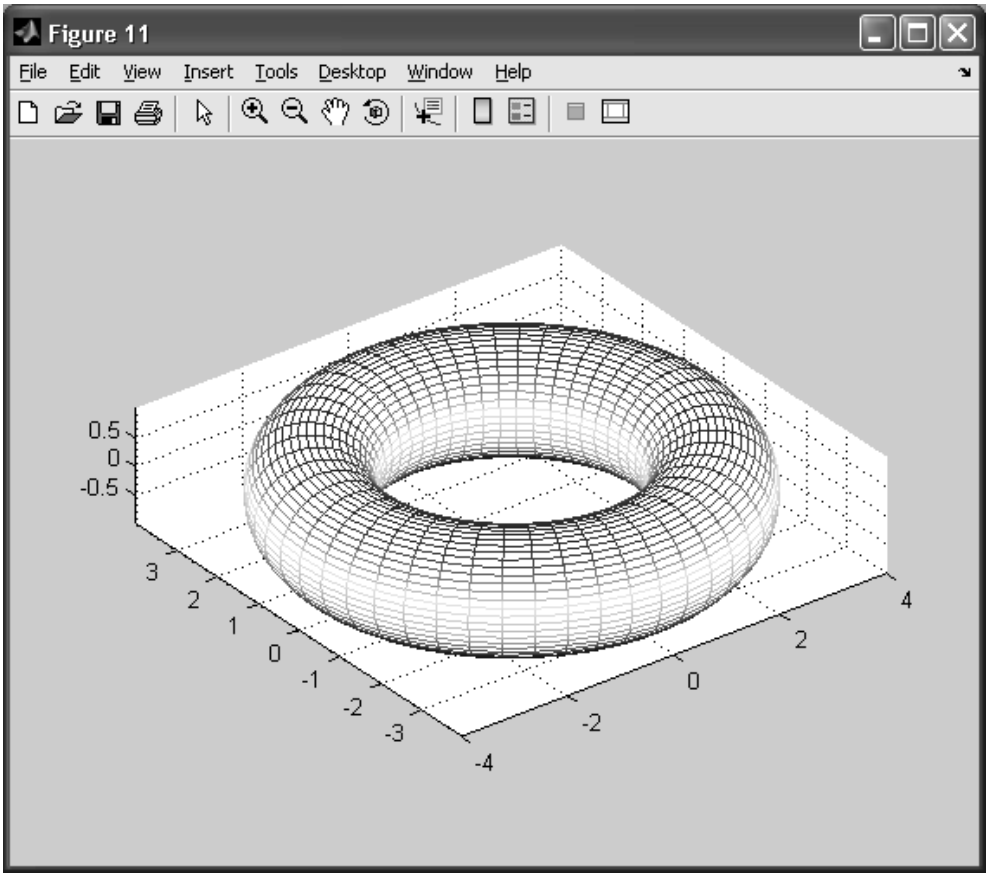

Рис. 7.12. Вид тора с радиусами r=1, R=3

Рассмотрим возможности функции `mesh` при необходимости визуализации более сложных поверхностей на примере геликоида. Геликоидом называется поверхность, образованная движением прямой, вращающейся вокруг оси и перпендикулярной к ней и одновременно поступательно движущейся в направлении этой оси, причем скорости этих движений пропорциональны. Геликоид можно рассматривать как винтовую поверхность, описываемую следующими соотношениями:

$$\begin{cases} x = u \cdot \cos(v) \\ y = u \cdot \sin(v) \\ z = a \cdot v \end{cases}$$

Пусть значения параметров меняются в диапазоне $0 \le u \le 4$, $0 \le v \le 2\pi$, параметр $a = 1$ Программа построения геликоида при заданных значениях приведена в листинге 7.4.

Листинг 7.4. Программная реализация построения геликоида

```
%Диапазон изменения переменной u
N=40; % Число точек для интервала изменения переменной
au=0;
bu=4;
u=au:(bu-au)/N:bu;
%Диапазон изменения переменной v
av=0; bv=4*pi;
M=50;% Число точек для интервала изменения переменной v
v=av:(bv-av)/M:bv;
[U,V]=meshgrid(u,v);
X=U.*cos(V);
Y=U.*sin(V);
a=1;
Z=a*V;
mesh(X,Y,Z);
axis equal % Масштабирование осей
```

Результат выполнения данной программы представлен на рис. 7.13.

Выбирая цвет линий сетки, близкий к фиолетовому, получим более декоративный вид поверхности. В случае выбора пункта No Color сетка на поверхности исчезает и она имеет вид, представленный на рис. 7.61.

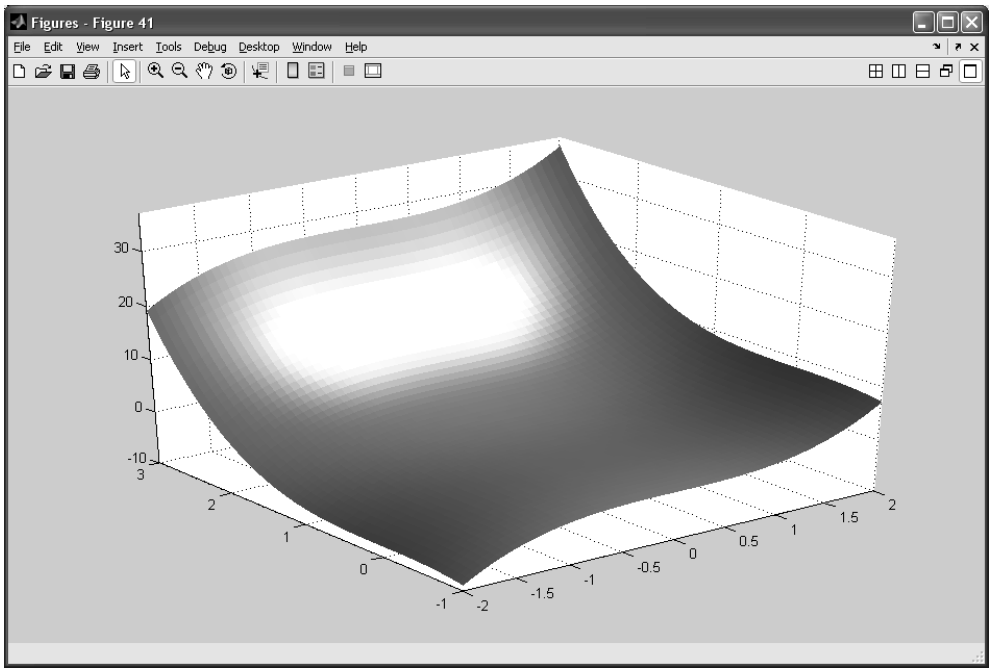

Рис. 7.61. Вид поверхности функции $f(x,y) = x^3 + y^3$ без линий сетки

Резюме

Таким образом, на примере можно увидеть, что MATLAB предоставляет большие возможности интерактивной настройки экранного представления графических объектов. Следует отметить, что наряду с этим, есть возможность автоматической генерации программного кода, обеспечивающего визуализацию графических объектов с заданными пользователем настройками. Комбинируя программный код и возможности ручной настройки, можно создавать высокоэффективные приложения.