第3章 MATLAB图形可视化

MATLAB语言丰富且功能卓越的图形可视化功能，使得数学计算结果可以方便地、多样地实现可视化。这是其他编程语言所不能及的，而且得到的图形可方便地插入Word和LaTeX等其他排版系统。MATLAB图形可视化操作分析了常见二维、三维图形绘制，以及MATLAB动画设计，方便用户进行可视化设计。

学习目标：

- 熟练掌握MATLAB编程表示方法；
- 熟练运用MATLAB产生可视化图形；
- 熟练掌握MATLAB进行动画设计等。

3.1 图形绘制

基于由浅入深的原则，本节将从最简单的平面上的点的表示入手，逐步深入，由离散数据的表示到连续数据的表示，使得读者易于掌握其中规律。

3.1.1 离散数据图形绘制

一个二元实数标量对(x_0, y_0)可以用平面上的点来表示，一个二元实数标量数组$[(x_1, y_1)(x_2, y_2)\cdots(x_n, y_n)]$可以用平面上的一组点来表示，对于离散函数$Y=f(X)$，当$X$为一维标量数组$X=[x_1, x_2, \cdots, x_n]$时，根据函数关系可以求出$Y$相应地为一维标量$Y=[y_1, y_2, \cdots, y_n]$。

当把这两个向量数组在直角坐标系中用点序列来表示时，就实现了离散函数的可视化。当然，这些图形上的离散序列所反映的只是X所限定的有限点上或有限区间内的函数关系。应当注意的是，MATLAB是无法实现对无限区间上的数据的可视化的。

【例3-1】 离散数据的图形绘制。

```
clc,clear
x = 1:10;
```

```
y = [0.0370    0.0340    0.0270    0.0400    0.0350 0.0270    0.0260    0.0260
0.0270    0.0250];
plot(x,y,'ro--')
```

运行该程序文件，得到图形如图 3-1 所示。

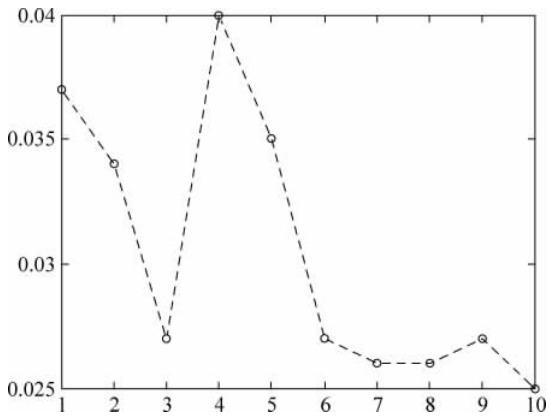

图 3-1　离散数据图形绘制

3.1.2　函数图形绘制

在 MATLAB 中，是无法画出真正的连续函数的，因此在实现连续函数的可视化时，首先也必须将连续函数用在一组离散自变量上计算函数结果，然后将自变量数组和结果数组在图形中表示出来。

当然，这些离散的点还是不能表现函数的连续性的。为了更形象地表现函数的规律及其连续变化，通常采用以下两种方法：

(1) 对离散区间进行更细的划分，逐步趋近函数的连续变化特性，直到达到视觉上的连续效果。

(2) 把每两个离散点用直线连接，以每两个离散点之间的直线来近似表示两点间的函数特性。

【例 3-2】　函数图形绘制。

```
clc,clear
x = 1:0.01:10;
y = tan(x);
plot(x,y,'r')
```

运行该程序文件，得到图形如图 3-2 所示。

【例 3-3】　对 x，y 界定的区域填充，并对各属性设置对应的属性值。编程如下：

```
clc,clear,close all
t = linspace(0,2 * pi,10);
```

```
x = sin(2 * t);
y = cos(2 * t);
area(x,y,'facecolor','r')
```

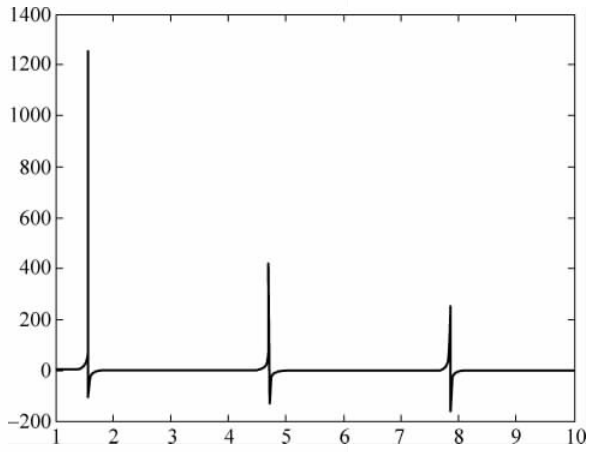

图 3-2　函数图形绘制

运行程序,输出结果如图 3-3 所示。

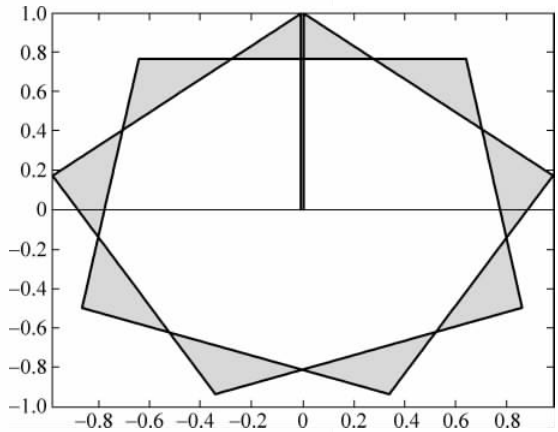

图 3-3　区域填充

3.1.3　图形绘制的基本步骤

通过上述实例,可以总结出利用 MATLAB 绘制图形的一般步骤。大致分为如下 7 个步骤:

(1) 数据准备。主要工作是产生自变量采样向量,计算相应的函数值向量。

(2) 选定图形窗口及子图位置。默认情况下,MATLAB 系统绘制的图形为 figure. 1、figure. 2、…依次类推。

(3) 调用绘图函数绘制图形,如 plot()函数。

(4) 设置坐标轴的范围、刻度及坐标网格。
(5) 利用对象属性值设置或者利用图形窗口工具栏设置线型、标记类型及其大小等。
(6) 添加图形注释,如图名、坐标名称、图例、文字说明等。
(7) 图形的导出与打印。

3.2 二维图形绘制

绘制二维图形常用的指令为 plot()。根据不同的坐标参数,它可以在二维平面上绘制出不同的曲线。MATLAB R2016a 主窗口中的"绘图"功能区能够利用工作空间的数据方便地画出各种类型的图形,不需要相应的绘图程序代码。

3.2.1 plot 指令

将数对排序的一种方法是使用 plot 指令。该命令可以带有不同数目的参数。最简单的形式就是将数据传递给 plot,但是线条的类型和颜色可以通过使用字符串来指定,这里用 str 表示。线条的默认类型是实线型。

下面给出 plot 指令的一般使用规范。

1) plot 指令使用规范 1: plot(x,y)

语句说明: 以 x 为横坐标,y 为纵坐标,按照坐标(x_j,y_j)的有序排列绘制曲线。

【例 3-4】 绘制给定数据点的增长率图。编程如下:

```
clc,clear,close all
load('x1 - 38.mat')
% 增长率
for i = 1:38
     x1(i) = i;
end
for i = 2:39
    y11((i - 1),:) = y1(i,:) - y1((i - 1),:);
end
for i = 1:38
    for j = 1:6
        y111(i,j) = y11(i,j)/y1(i,j);
    end
end
% 增长率时间曲线
for i = 1:6
    subplot(3,2,i);
    plot(x1,y111(:,i));
end
```

运行程序,输出图形如图 3-4 所示。

2) plot 指令使用规范 2: plot(y)

语句说明: y 为一维实数数组,以 1:n 为横坐标,y_j 为纵坐标绘制曲线(n 为 y 的长度)。

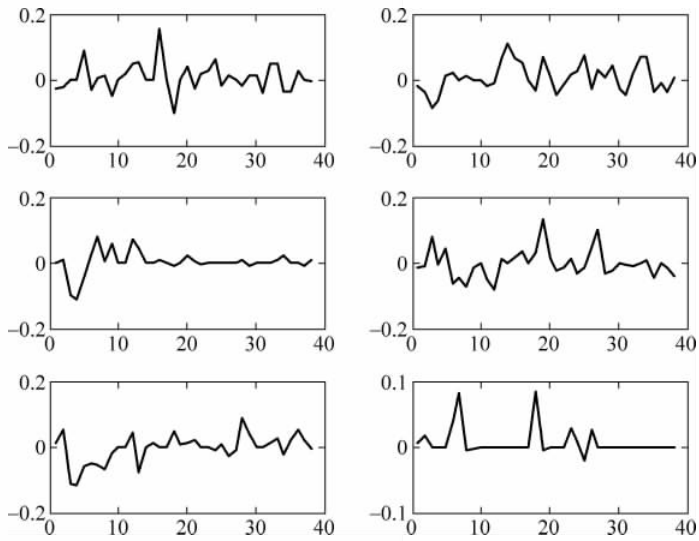

图 3-4　增长率曲线图

【例 3-5】 采用 plot()编程如下：

```
clc,clear,close all
y = [0.0370  0.0340  0.0270  0.0400  0.0350  0.0270  0.0260  0.0260  0.0270
0.0250];
plot(y','rs-','linewidth',2)
```

运行程序，输出图形如图 3-5 所示。

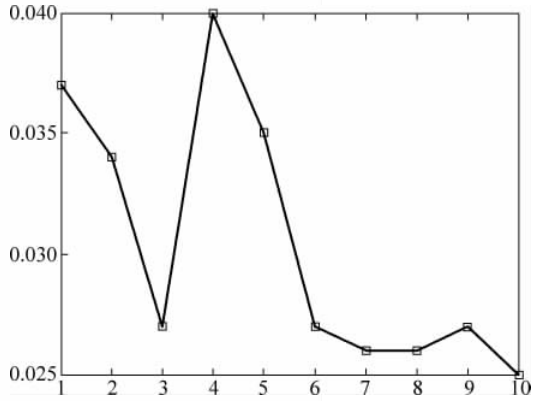

图 3-5　图形绘制

3.2.2　格栅

当图像需要对具体数值有更加清楚的展示时，在图形中添加格栅是十分有效的方法。在 MATLAB 中，利用 grid on 指令可以在当前图形的单位标记处添加格栅，利用

grid off 指令则可以取消格栅的显示，单独使用 grid 指令则可以在 on 与 off 状态下交替转换，即起到触发的作用。

【例 3-6】 画 $y=x^2\sin(x)$的图形。编程如下：

```
clc,clear,close all
ezplot('x^2*sin(x)',[-10,10])
grid on
```

运行程序，输出图形如图 3-6 所示。

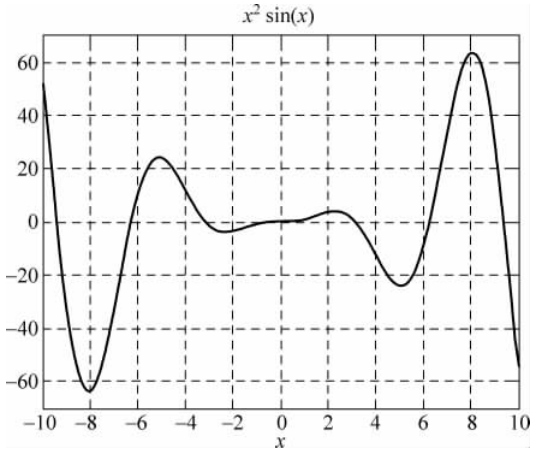

图 3-6 栅格化的二维图形绘制

【例 3-7】 去掉栅格。编程如下：

```
grid off
ezplot('x^2*sin(x)',[-10,10])
```

运行程序，输出图形如图 3-7 所示。

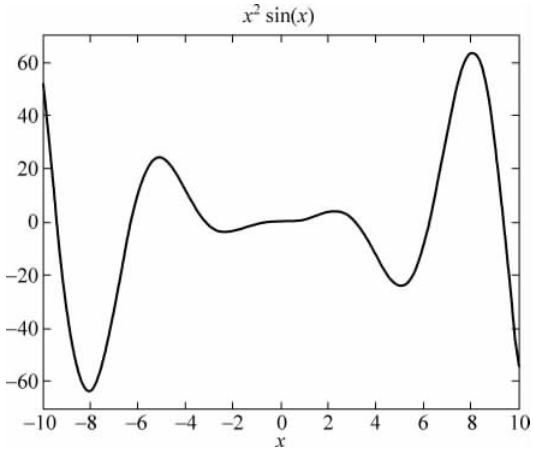

图 3-7 去栅格化的二维图形绘制

3.2.3 图形标记说明

通常，曲线所表示出的函数或数据的规律都需要进行一些文字的说明或标注。现将图形窗口中的文本操作指令列出如下：

(1) title('text')。指令的功能为在图形窗口顶端的中间位置输出字符串'text'作为标题。

(2) xlabel('text')。指令的功能为在 x 轴下的中间位置输出字符串'text'作为标注。

(3) ylabel('text')。指令的功能为在 y 轴边上的中间位置输出字符串'text'作为标注。

(4) zlabel('text')。指令的功能为在 z 轴边上的中间位置输出字符串'text'作为标注。

(5) legend(str1,str2,…,pos)。指令的功能为在当前图上输出图例，并用说明性字符串 str1、str2 等作标注。其中，参数 pos 的可选项目如表 3-1 所示。

表 3-1 曲线线型

线型代号	表 示 线 型
−1	将图例框放在坐标轴外的右侧
0	将图例框放在图窗内与曲线交叠最小的位置
1	将图例框放在图窗内右上角
2	将图例框放在图窗内左上角
3	将图例框放在图窗内左下角
4	将图例框放在图窗内右下角

(6) legend(str1,str2,…,'Location','pos')。指令的功能为在当前图上输出图例，并用说明性字符串 str1、str2 等作标注。其中，参数'pos'的可选项目列表如表 3-2 所示。

表 3-2 图形标记

标 记 代 号	表 示 标 记	标 记 代 号	表 示 标 记
North	图窗内最上端	SouthOutside	图窗外下部
South	图窗内最下端	EastOutside	图窗外右侧
East	图窗内最右端	WestOutside	图窗外左侧
West	图窗内最左端	NorthEastOutside	图窗外右上部
NorthEast	图窗内右上角(二维图窗的默认项)	NorthWestOutside	图窗外左上部
NorthWest	图窗内左上角	SouthEastOutside	图窗外右下部
SouthEast	图窗内右下角	SouthWestOutside	图窗外左下部
SouthWest	图窗内左下角	Best	图窗内与曲线交叠最小的位置
NorthOutside	图窗外上部	BestOutside	图窗外最不占空间的位置

(7) legend off。指令的功能为从当前图形中清除图例。

【例 3-8】 对于图形坐标轴以及曲面标记,具体的程序如下:

```
clc,clear,close all
A = rand(10,10);
surf(A)
xlabel('x')
ylabel('y')
zlabel('z')
title('三维曲面')
legend('曲面','NorthEast')
```

运行程序,输出图形如图 3-8 所示。

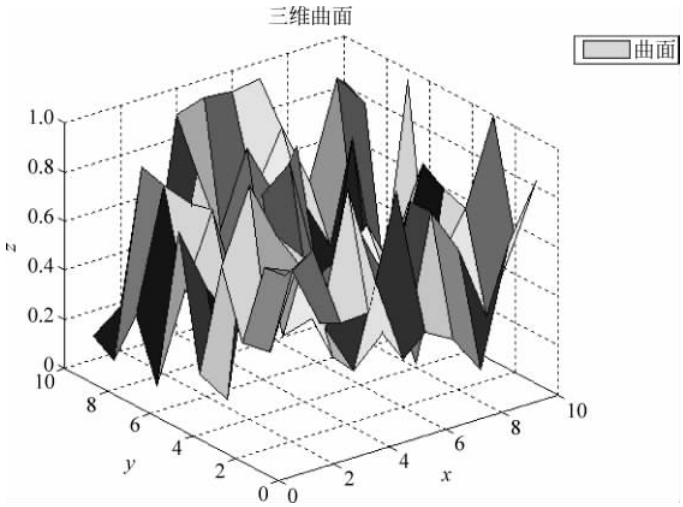

图 3-8 图形属性标记

3.2.4 线型、标记和颜色

当同一张图形中同时画了多条曲线时,则需要使用不同的线型、标记或不同的曲线颜色来区分不同的曲线。

1. 线型

MATLAB 平台中共有 5 种不同线型,如表 3-3 所示。

表 3-3 曲线线型

线型代号	表示线型	线型代号	表示线型
-	实线	:	点线
--	虚线	none	无线
-.	点画线		

2. 标记

MATLAB平台中共有14种不同标记方式，如表3-4所示。

表3-4 图形标记

标记代号	表示标记	标记代号	表示标记
.	点	。	o
*	星号	+	+
square	正方形	×	×
diamond	菱形	<	顶点指向左边的三角形
pentagram	五角星形	>	顶点指向右边的三角形
hexagram	六角星形	∧	正三角形
none	无点	∨	倒三角形

3. 颜色

MATLAB平台中有代号的颜色共有8种，如表3-5所示。

表3-5 曲线或标记颜色

颜色代号	表示颜色	颜色代号	表示颜色
g	绿色	w	白色
m	品红色	r	红色
b	蓝色	k	黑色
c	灰色	y	黄色

【例3-9】 对于线型、标记和颜色的程序书写，程序如下：

```
clc,clear,close all
figure
x = -20:0.01*pi:pi*8;
y = x.*(x).*(x)/1000;
plot(x,y,'r:','LineWidth',3);
hold on;
plot(x,y,'k--','LineWidth',2);
plot(x,y+3,'b--','LineWidth',1);
plot(x,y+5,'rs','LineWidth',3);
hold on;
plot(x,y-3,'kp--','LineWidth',2);
plot(x,y-5,'bh--','LineWidth',1);
axis tight
```

运行程序，输出图形如图3-9所示。

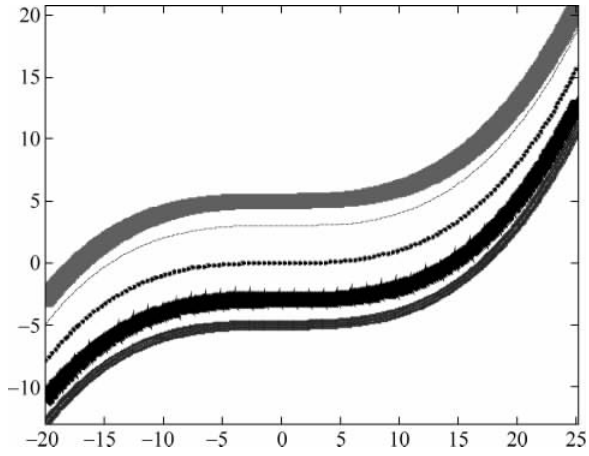

图 3-9 图形线型、标记和颜色

3.2.5 子图绘制

MATLAB 允许用户在同一个图形窗中同时绘制多幅相互独立的子图，这需要应用到 subplot 指令。其具体语句规范如下：

(1) subplot(m,n,k)。在 $m\times n$ 幅子图中的第 k 幅图作为当前曲线的绘制图。

(2) subplot('position',[left bottom width height])。在指定位置上生成子图，并作为当前曲线的绘制图。

subplot 指令说明：

(1) subplot(m,n,k)指令生成的图形窗中将会有 $m\times n$ 幅子图，k 是子图的编号，编号的顺序如下：左上为第 1 幅子图，然后先向右后向下依次排号，该指令产生的子图分割与占位完全按照默认值自动进行。

(2) subplot('position',[left bottom width height])指令所产生的子图的位置由用户指定，指定位置的 4 个元素采用归一化的标称单位，即认为图形窗的宽、高的取值范围均为[0,1]，左下角的坐标为(0,0)。

(3) 指令所产生的子图，彼此之间相互独立，所有的绘图指令都可以在任一子图中运用，而对其他的子图不起作用。

(4) 在使用 subplot 指令之后，如果再想绘制充满整个图形窗的图时，应当先使用 clf 指令对图形窗进行清空。

【例 3-10】 创建与分割图形窗口，编程如下：

```
clc,clear,close all
clf,b = 2 * pi;x = linspace(0,b,50);
for k  = 1:9
    y = sin(k * x);
    subplot(3,3,k),plot(x,y),axis([0,2 * pi, - 1,1])
end
```

运行程序，输出图形如图 3-10 所示。

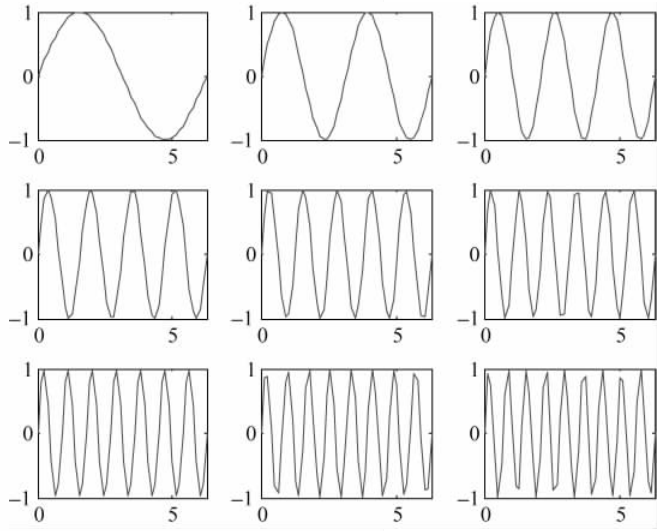

图 3-10　图形窗口设置

3.2.6　拓扑关系图

拓扑关系绘图指令如下：

```
gplot(A,xy,lc)
```

其中，***A*** 为一个图 G 的邻接矩阵，即若 $a(i,j)\neq 0$，则从节点 i 到节点 j 有一条边，但 ***A*** 未必为方阵；xy 为一个 $n\times 2$ 的矩阵，表示各节点的位置，即 xy(i,:)=[x(i),y(i)]；lc 为线型和颜色，默认为“b-”，其指明方式与 plot 画图属性设置相同。

【例 3-11】 绘制拓扑关系图。编程如下：

```
%拓扑图
clc,clear,close all
a=[0,1,0,1,0;
    1,0,0,0,1;
    0,0,0,0,1;
    1,1,0,0,0;
    0,1,1,0,0;];
xy=[1,5;4,7.4;
    6,3.5;5,2;2,3];
gplot(a,xy,'r-')
text(1.1,5,'1')
text(4,7.4,'2')
text(5.9,3.6,'3')
text(5.1,2.1,'4')
text(2,2.8,'5')
```

运行程序，输出结果如图 3-11 所示。

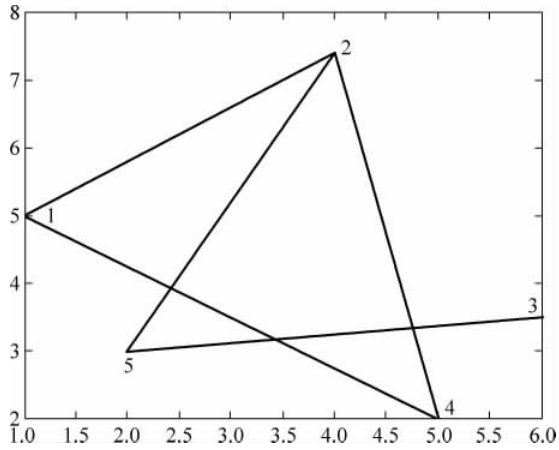

图 3-11　拓扑关系图

【例 3-12】　绘制足球网络拓扑图。编程如下：

```
clc,clear,close all
[a,v] = bucky;
H = sparse(60,60);
gplot(a - H,v,'r - ');
hold on
gplot(H,v,'bo - ')
```

运行程序，输出结果如图 3-12 所示。

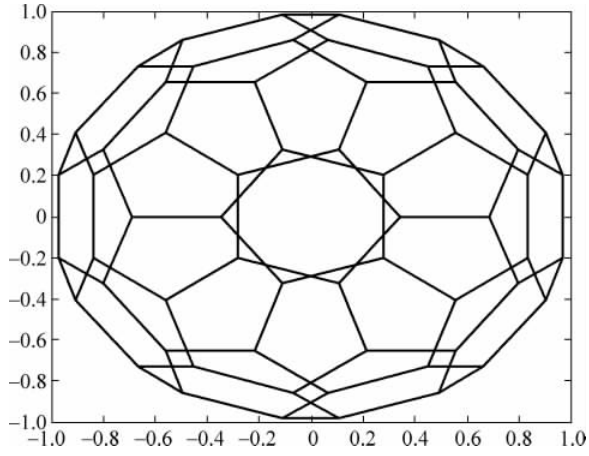

图 3-12　足球网络拓扑图

3.2.7　双坐标轴绘制

在实际的应用中，常常需要把同一自变量的两个不同量纲、不同量级的函数量的变化同时绘制在同一个图形窗中。例如，在同一张图中同时展示空间一点上的电磁波的幅

度和相位随时间的变化；或者不同时间内的降雨量和温湿度的变化；或者放大器的输入、输出电流的变化曲线等。MATLAB 中的 plotyy()函数可以对上述功能进行实现。其具体的句法格式如下：

(1) plotyy(X1,Y1,X2,Y2)。该语句的功能为以左、右不同的纵轴分别绘制 X1-Y1 和 X2-Y2 两条曲线。

(2) plotyy(X1,Y1,X2,Y2,Fun)。该语句的功能为以左、右不同的纵轴以 Fun 指定的形式分别绘制 X1-Y1 和 X2-Y2 两条曲线。

(3) plotyy(X1,Y1,X2,Y2,Fun1,Fun2)。该语句的功能为以左、右不同的纵轴分别以 Fun1、Fun2 指定的形式绘制 X1-Y1 和 X2-Y2 两条曲线。

使用 plotyy 指令时需要注意的是：左侧的纵轴用来描述 X1-Y1 曲线，右侧的纵轴用来描述 X2-Y2 曲线。轴的范围与刻度值都是自动生成的，进行人工设置时，使用的绘图指令与一般的绘图指令相同。

【例 3-13】 同坐标绘制不同图形。

在同一坐标中使用不同坐标系绘制不同的图形。具体编程实例如下：

```
%同一图形中不同坐标系绘制不同图形
clc,clear,close all
x = -2*pi:pi/20:2*pi;
y = sin(x);
z = 2*abs(cos(x));
subplot(211)
plot(x,y,x,z);
title('按相同坐标刻度绘制不同图形')
subplot(212)
plotyy(x,y,x,z,'plot','semilogy')
title('按不同的坐标系进行绘制')
```

运行程序，输出结果如图 3-13 所示。

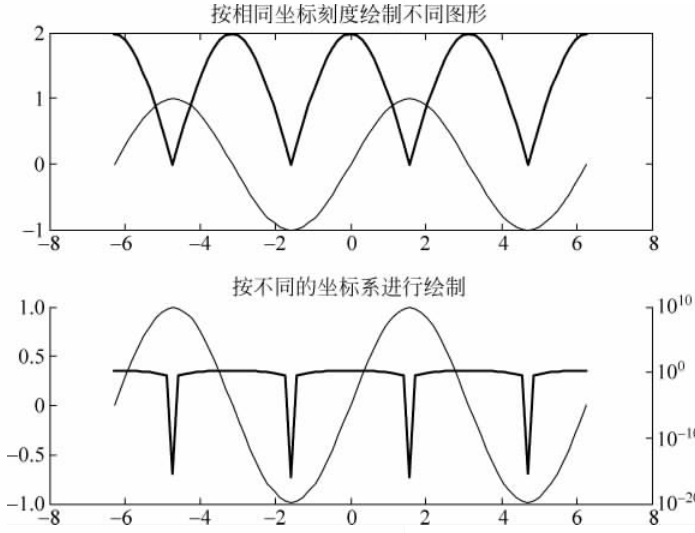

图 3-13 同一坐标绘图

3.2.8 二元函数的伪色彩

用颜色表示平面图表中二元函数值的大小(高度),方法如下。

1. 指定颜色集

(1) pcolor(X,Y,Z)。表示在由 X、Y 构造的平面上,用 Z 的元素确定相应小格子的颜色。

(2) pcolor(Z)。表示在由 Z 的下标值构造的平面上,用 Z 的元素确定相应小格子的颜色。

2. 使用 Shading 命令控制着色模式

```
shading flat/interp/faceted
```

默认为 faceted,即着色网格并附加黑色网线。

若为 flat,则无网线,且各网格颜色单一。若为 interp,则无网线,且各网格颜色通过相应 4 个顶点的颜色值进行双线性内插得出。

3. 使用 hold on 使以上设定一直保持

【例 3-14】 二元函数的伪色彩、二元函数图像的伪图像实例。编程如下:

```
clc,clear,close all
[x,y,z] = peaks(50);
pcolor(x,y,z);
shading interp
hold on
contour(x,y,z,10,'k')
pause
shading flat
contour(x,y,z,10,'k')
pause
shading faceted
contour(x,y,z,10,'k')
```

运行程序,输出结果如图 3-14～图 3-16 所示。

3.2.9 MATLAB 特殊符号标记

MATLAB 提供了特殊符号标记对图形进行修饰,对于上下标的标定,工程上应用较广泛,例如,$e^{-t}\sin t$、$x-\chi_{\alpha}^{2}(2)$ 等的标记,MATLAB 提供了上下控制指令,如表 3-6 所示。

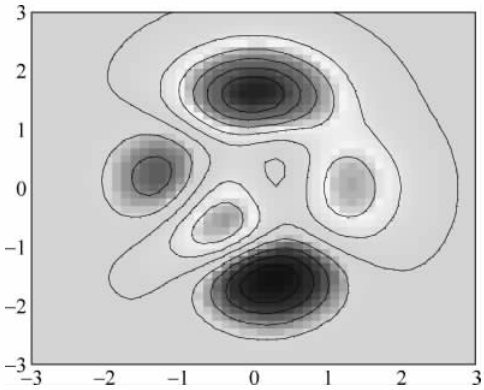

图 3-14　interp 彩色图

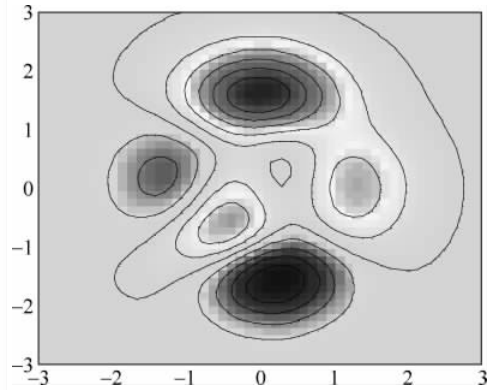

图 3-15　flat 彩色图

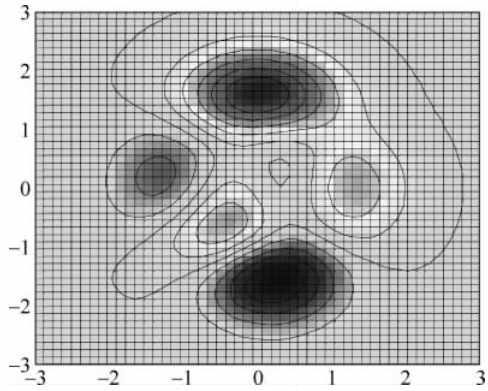

图 3-16　faceted 彩色图

表 3-6　上下控制指令

类型	指令	arg 取值	举　例	
			示例指令	效果
上标	^{arg}	任何合法字符	'\ite^{-t}sint'	$e^{-t}\sin t$
下标	_{arg}	任何合法字符	'x～{\chi}_{\alpha}^{2}(3)'	$x-\chi_{\alpha}^{2}(2)$

对于常见的阿拉伯字母，α、β、δ(Δ)等注释符号，MATLAB 也提供了特殊的字符指令，如表 3-7 所示。

表 3-7　希腊字母指令

指　令	字符	指　令	字符
\alpha	α	\eta	η
\beta	β	\theta(\Theta)	$\theta(\Theta)$
\gamma(\Gamma)	$\gamma(\Gamma)$	\iota	ι
\delta(\Delta)	$\delta(\Delta)$	\kappa	κ
\epsilon	ε	\lambda(\Lambda)	$\lambda(\Lambda)$
\zeta	ς	\mu	μ

续表

指　令	字符	指　令	字符
\vartheta	ϑ	\varsigma	ς
\Nu	υ	\upsilon(\Upsilon)	$\nu(\Upsilon)$
\xi(\Xi)	$\xi(\Xi)$	\phi(\Phi)	$\varphi(\Phi)$
\pi(\Pi)	$\pi(\Pi)$	\chi	χ
\rho	ρ	\psi(Psi)	$\psi(\Psi)$
\sigma(\Sigma)	$\sigma(\Sigma)$	\omega(\Omega)	$\omega(\Omega)$
\tau	τ	\varpi	ω

还有其他的一些数学字符指令，即数学公式编辑器中出现的字符，MATLAB 提供了相应的字符指令，如表 3-8 所示。

表 3-8　其他字符指令

指令	字符	指令	字符	指令	字符	指令	字符	指令	字符
\approx	$\approx$	\propto	$\propto$	\exists	$\exists$	\cap	$\cap$	\downarrow	$\downarrow$
\cong	$\cong$	\sim	$\sim$	\forall	$\forall$	\cup	$\cup$	\leftarrow	$\leftarrow$
\div	$\div$	\times	$\times$	\in	$\in$	\subset	$\subset$	\leftrightarrow	$\leftrightarrow$
\equiv	$\equiv$	\oplus	$\oplus$	\infty	∞	\supseteq	$\subseteq$	\rightarrow	$\rightarrow$
\geq	$\geqslant$	\oslash	ϕ	\perp	$\perp$	\subset	$\supset$	\uparrow	$\uparrow$
\leq	$\leqslant$	\otimes	$\otimes$	\prime	$'$	\supseteq	$\supseteq$	\circ	$\circ$
\neq	$\neq$	\int	$\int$	\cdot	$\cdot$	\Im	$\Im$	\bullet	$\bullet$
\pm	$\pm$	\partial	∂	\ldots	$\cdots$	\Re	$\Re$	\copyright	$\odot$

针对这些特殊字符的标记，用户可以将要使用的变量显示在图形中，达到一一对应的关系，因此实际应用中广泛使用。

MATLAB 中用于文字、字符等的标记函数为 text()。具体的使用方法如下：

(1) text(x,y,'text')。指令的功能为在图形窗口的(x,y)处写字符串'text'。坐标 x 和 y 按照与所绘制图形相同的刻度给出。对于向量 x 和 y，字符串'text'写在(x_i,y_i)的位置上。如果'text'是一个字符串向量，即一个字符矩阵，且与 x，y 有相同的行数，则第 i 行的字符串将写在图形窗口的(x_i,y_i)的位置上。

(2) text(x,y,'text','sc')。指令的功能为在图形窗口的(x,y)处输出字符串'text'，给定左下角的坐标为(0.0,0.0)，右上角的坐标则为(1.0,1.0)。gtext('text')通过使用鼠标或方向键，移动图形窗口中的十字光标，让用户将字符串 txt 放置在图形窗口中。当十字光标走到所期望的位置时，用户按下任意键或单击鼠标上的任意按钮，字符串将会写入在窗口中。

【例 3-15】 特殊字符标记具体的代码如下：

```
clc,clear,close all
load('x1 - 38.mat')
plot(1:39,y1(:,1))
hold on
 % grid on
```

```
text(10,13,'\approx')
text(10,12.5,'\exists')
text(5,13,'\approx')
text(5,12.5,'\exists')
```

运行程序,得到如图 3-17 所示标记图。

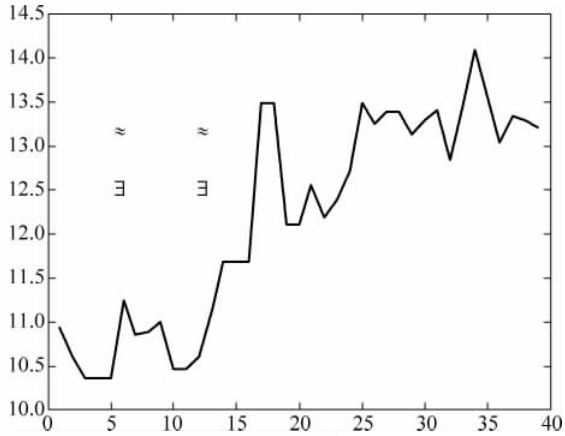

图 3-17　图形标记

【例 3-16】 在同一张图上画出 $y=\sin(t)$和 $y=3e^{-0.5t}$,这里 $t\in[0,3\pi]$,并进行适当的标注。编程如下:

```
clc,clear,close all
clf, x = linspace(0,8 * pi,100);
F = inline('sin(x + cos(x + sin(x)))');
y1 = sin(x + cos(x + sin(x)));
y2 = 0.2 * x + sin(x + cos(x + sin(x)));
plot(x,y1,'k:',x,y2,'k - ')
legend('sin(x + cos(x + sin(x))','0.2x + sin(x + cos(x + sin(x)))',2)
h = plot([0:0.1:2 * pi],sin([0:0.1:2 * pi])); grid on
set(h,'LineWidth',5,'color','red'); set(gca,'GridLineStyle','-','fontsize',16)
% 设置 y 坐标的刻度并加以说明,并改变字体的大小
h = plot([0:0.1:2 * pi],sin([0:0.1:2 * pi]));grid on;
set(gca,'ytick',[ - 1  - 0.5 0 0.5 1]), set(gca,'yticklabel','a|b|c|d|e'),
set(gca,'fontsize',20)
% 文字标注指令
plot(x,y1,'b',x,y2,'k - ') ,
set(gca,'fontsize',15,'fontname','times New Roman'),   % 设置轴对象的字体为 times New Roman
title('\it{Peroid and linear peroid function}');      % 加标题
xlabel('x from 0 to 8 * pi \it{x}'); ylabel('\it{y}'); % 说明坐标轴
text(x(49),y1(50) - 0.4,'\fontsize{15}\bullet\leftarrowThe period function {\itf(x)}');
% 在坐标(x(49),y1(50) - 0.4)处作文字说明, 各项设置用"\"隔开
% \fontsize{15}\bullet\leftarrow 的意义依次是:\字体大小 = 15 \ 画圆点 \左箭头
text(x(14),y2(50) + 1,'\fontsize{15}The linear period function {\itg(x)}\rightarrow\
bullet')
```

运行程序,输出图形如图 3-18 所示。

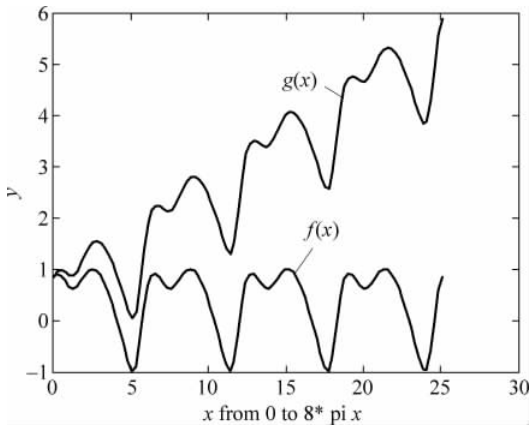

图 3-18　图形标记

【例 3-17】 填充多边形指令,填充由点$(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)$构成的多边形,其颜色由 c 指明。填充多边形区域编程如下:

```
%区域颜色填充
clc,clear,close all
x = linspace(0,10,50);
y = sin(x). * exp( - x/5);
fill(x,y,'b')                    %填充蓝色
text(4,0.01,'蓝色')
```

运行程序,输出结果如图 3-19 所示。

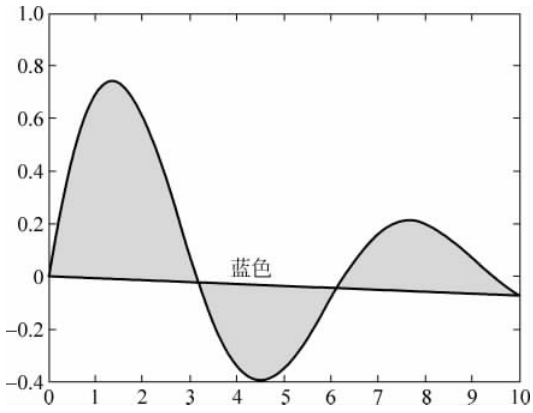

图 3-19　多边形涂色

3.3　三维图形绘制

本节通过例子观察 MATLAB 的三维绘图功能和技巧。

3.3.1 网格图绘制

三维网格图和曲面图的绘制比三维曲线图的绘制稍显复杂，主要是因为：绘图数据的准备以及三维图形的色彩、明暗、光照和视角等的处理。绘制函数 $z=f(x,y)$ 的三维网格图的过程如下：

(1) 确定自变量 x 和 y 的取值范围和取值间隔，即

```
x = x1:dx:x2,y = y1:dy:y2
```

(2) 构成 xoy 平面上的自变量采样个点矩阵。

① 利用“格点”矩阵的原理生成矩阵。

```
x = x1:dx:x2; y = y1:dy:y2;
X = ones(size(y)) * x;
Y = y * ones(size(x));
```

② 利用 meshgrid 指令生成“格点”矩阵。

```
x = x1:dx:x2; y = y1:dy:y2;
[X,Y] = meshgrid(x,y);
```

(3) 计算在自变量采样“格点”上的函数值：$z=f(x,y)$。

绘制网格图的基本 mesh 指令的句法格式如下：

(1) mesh(X,Y,Z)。其功能为以 $\boldsymbol{X}$ 为 x 轴自变量、$\boldsymbol{Y}$ 为 y 轴自变量，绘制网格图；$\boldsymbol{X}$、$\boldsymbol{Y}$ 均为向量，若 $\boldsymbol{X}$、$\boldsymbol{Y}$ 长度分别为 m、n，则 $\boldsymbol{Z}$ 为 $m\times n$ 的矩阵，即[m,n]=size(Z)，则网格线的顶点为(X_j,Y_i,Z_{ij})。

(2) mesh(Z)。其功能为以 $\boldsymbol{Z}$ 矩阵列下标为 x 轴自变量、行下标为 y 轴自变量，绘制网格图。

(3) mesh(X,Y,Z,C)。其功能为以 $\boldsymbol{X}$ 为 x 轴自变量、$\boldsymbol{Y}$ 为 y 轴自变量，绘制网格图；其中 $\boldsymbol{C}$ 用于定义颜色，如果不定义 $\boldsymbol{C}$，则成为 mesh(X,Y,Z)，其绘制的网格图的颜色随着 $\boldsymbol{Z}$ 值(即曲面高度)成比例变化。

④ mesh(X,Y,Z,'PropertyName',PropertyValue,…)。其功能为以 $\boldsymbol{X}$ 为 x 轴自变量、$\boldsymbol{Y}$ 为 y 轴自变量，绘制网格图；PropertyValue 用来定义网格图的标记等属性。

【例 3-18】 绘制 $z=f(x,y)=(1-x)^{-\frac{1}{2}}\ln(x-y)$ 的图像，作定义域的裁剪。

(1) 观察 meshgrid 指令的效果。编写程序如下：

```
clc,clear,close all
a = -0.98;b = 0.98;c = -1;d = 1;n = 10;
x = linspace(a,b,n); y = linspace(c,d,n);
[X,Y] = meshgrid(x,y);
plot(X,Y,'+')
```

运行程序，输出图形如图 3-20 所示。

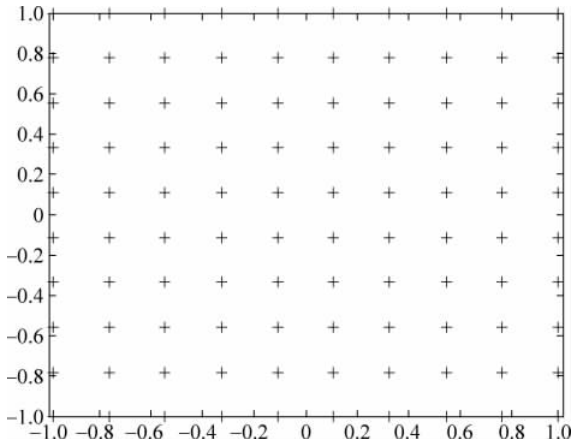

图 3-20　meshgrid 散点图

(2) 做函数的定义域裁剪,观察上述三维绘图指令的效果。编程如下:

```
clear,clf,
a = -1;b = 1;c = -15;d = 15;n = 20;eps1 = 0.01;
x = linspace(a,b,n);y = linspace(c,d,n);
[X,Y] = meshgrid(x,y);
for i = 1:n                                    %计算函数值 z,并作定义域裁剪
   for j = 1:n
     if (1 - X(i,j))< eps1|X(i,j) - Y(i,j)< eps1                    %if 语句这样用
        z(i,j) = NaN;                          %作定义域裁剪,定义域以外的函数值为 NaN
     else
        z(i,j) = 1000 * sqrt(1 - X(i,j))^ - 1. * log(X(i,j) - Y(i,j));
     end
   end
end
zz = -20 * ones(1,n);plot3(x,x,zz),grid off,hold on                %画定义域的边界线
mesh(X,Y,z)                                    %绘图,读者可用 meshz、surf、meshc 在此替换之
view([ -56.5 38]);
xlabel('x'),ylabel('y'),zlabel('z'), box on                    %把三维图形封闭在箱体里
```

运行程序,输出图形如图 3-21 所示。

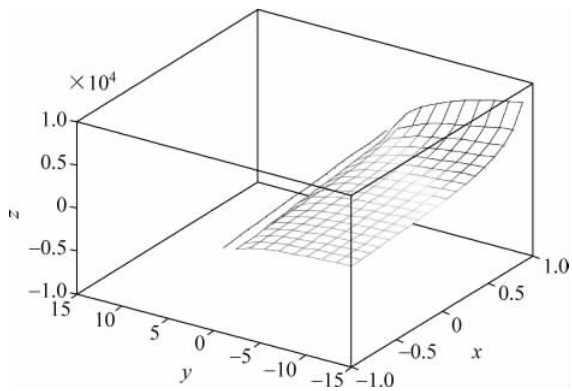

图 3-21　3D 图

3.3.2 曲线图绘制

在三维图形指令中，plot3 指令十分易于理解，其使用格式也与 plot 指令类似。其具体句法形式如下：

（1）plot3(X,Y,Z)。语句的功能为：X、Y、Z 为同维向量时，绘制以 X、Y、Z 为 x、y、z 坐标的三维曲线；X、Y、Z 为同维矩阵时，用 X、Y、Z 的对应列元素绘制 x、y、z 坐标的三维曲线，曲线的条数为矩阵的列数。

（2）plot3(X1,Y1,Z1,X2,Y2,Z2)。语句的功能为：绘制以 X1、Y1、Z1 和 X2、Y2、Z2 为 x、y、z 坐标的三维曲线。

（3）plot3(X,Y,Z,'PropertyName',PropertyValue,…)。语句的功能为：在 PropertyName 所规定的曲线属性下，绘制以 X、Y、Z 为 x、y、z 坐标的三维曲线。

（4）plot3(X1,Y1,Z1,'PropertyName1',PropertyName1,X2,Y2,Z2,'PropertyName1',PropertyName1)。语句的功能为：在 PropertyName1 所规定的曲线属性下，绘制以 X1、Y1、Z1 为 x、y、z 坐标的三维曲线；在 PropertyName2 所规定的曲线属性下，绘制以 X2、Y2、Z2 为 x、y、z 坐标的三维曲线。需要说明的是：plot3 指令用来表现的是单参数的三维曲线，而非双参数的三维曲面。

【例 3-19】 用平行截面法讨论由曲面 $z=x^2-2y^2$ 构成的马鞍面形状。编程如下：

```
clc,clear,close all
clf, a = -20;eps0 = 1;
[x,y] = meshgrid(-10:0.2:10);                    %生成平面网格
v = [-10 10 -10 10 -100 100];                    %设定空间坐标系的范围
%colormap(gray)                                  %将当前的颜色设置为灰色
z1 = (x.^2 - 2*y.^2) + eps;                      %计算马鞍面函数 z1 = z1(x,y)
z2 = a*ones(size(x));                            %计算平面 z2 = z2(x,y)
r0 = abs(z1 - z2)<= eps0;
%计算一个和 z1 同维的函数 r0。当 abs(z1 - z2)<= eps 时 r0 = 1; 当 abs(z1 - z2)> eps0 时,r0
= 0
%可用 mesh(x,y,r0)语句观察它的图形,体会它的作用,该方法可以套用
zz = r0.*z2;xx = r0.*x;yy = r0.*y;                %计算截割的双曲线及其对应的坐标
subplot(2,2,2),                                 %在第 2 图形窗口绘制双曲线
h1 = plot3(xx(r0~ = 0),yy(r0~ = 0),zz(r0~ = 0),'+');
set(h1,'markersize',2),hold on,
axis(v),grid on
subplot(2,2,1),                                  %在第 1 图形窗口绘制马鞍面和平面
mesh(x,y,z1);
grid,
hold on;
mesh(x,y,z2);
h2 = plot3(xx(r0~ = 0),yy(r0~ = 0),zz(r0~ = 0),'.');                    %画出二者的交线
set(h2,'markersize',6),hold on,axis(v),
for i = 1:5                                      %通过循环绘制一系列的平面去截割马鞍面
    a = 70 - i*30;                               %在这里改变截割平面
```

```
    z2 = a * ones(size(x));
    r0 = abs(z1 - z2)<= 1;
    zz = r0. * z2;
    yy = r0. * y;
    xx = r0. * x;
    subplot(2,2,3),
    mesh(x,y,z1);
    grid,hold on;
    mesh(x,y,z2);
    hidden off
    h2 = plot3(xx(r0~ = 0),yy(r0~ = 0),zz(r0~ = 0),'.');
    axis(v),grid
    subplot(2,2,4),
    h4 = plot3(xx(r0~ = 0),yy(r0~ = 0),zz(r0~ = 0),'o');
    set(h4,'markersize',2),
    hold on,
    axis(v),
    grid on
end
```

运行程序,输出图形如图 3-22 所示。

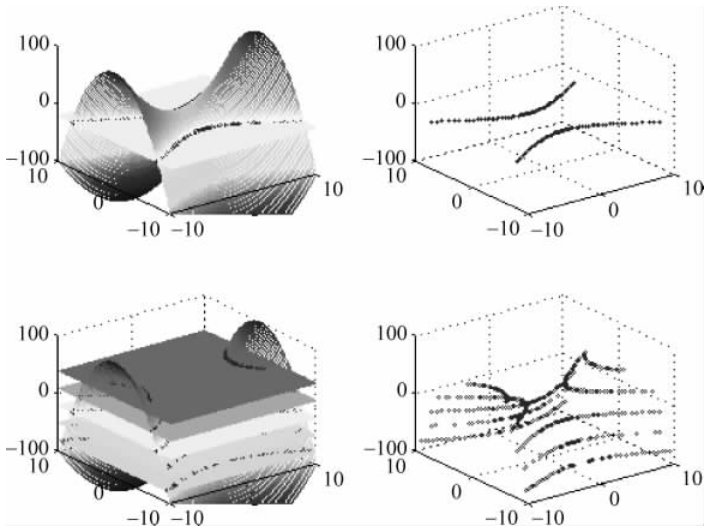

图 3-22　马鞍面

3.3.3　曲面图绘制

曲面图的绘制由 surf 指令完成,该指令的调用格式与 mesh 指令类似。具体如下:

(1) surf (X,Y,Z)。

(2) surf (Z)。

(3) surf (X,Y,Z,C)。

(4) surf(X,Y,Z,'PropertyName',PropertyValue,…)。

与 mesh 指令不同的是,mesh 指令所绘制的图形是网格划分的曲面图,而 surf 指令绘制得到的是平滑着色的三维曲面图,着色的方式是在得到相应的网格点后,对每一个网格依据该网格所代表的节点的色值(由变量 C 控制)来定义这一网格的颜色。

【例 3-20】 采用 surf 指令实现曲面的绘制。程序如下:

```
clc,clear,close all
X = -20:1:20;
Y = -20:1:20;
[X,Y] = meshgrid(X,Y);
Z = - X.^2 - Y.^2 + 100;
surf(X,Y,Z)
```

运行程序,输出图形如图 3-23 所示。

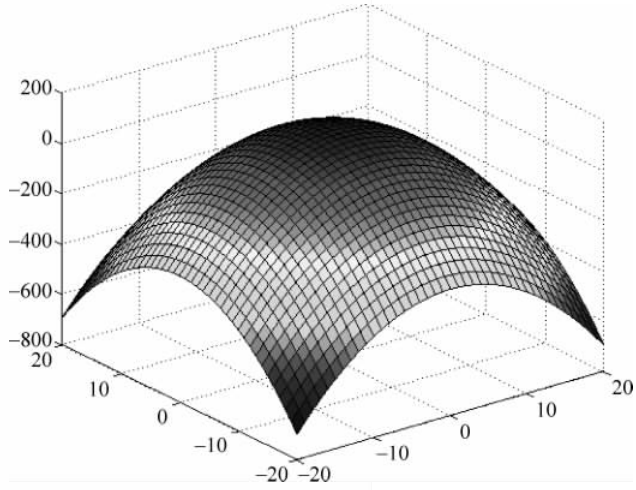

图 3-23 曲面绘制

3.3.4 等值线图绘制

等值线图,又叫作等高线图。绘制等值线图需要用到 contour 指令。其调用格式如下:

(1) contour(Z)。其功能为以 $\boldsymbol{Z}$ 矩阵列下标为 x 轴自变量、行下标为 y 轴自变量,绘制等值线图。

(2) contour(Z,n)。其中,n 为所绘制的图形等值线的条数。

(3) contour(Z,v)。其中,$\boldsymbol{v}$ 为向量,向量的长度为等值线的条数,并且等值线的值为对应的向量的元素值。

(4) contour(X,Y,Z)。其功能为以 $\boldsymbol{X}$ 为 x 轴自变量、$\boldsymbol{Y}$ 为 y 轴自变量,绘制等值线图;$\boldsymbol{X}$、$\boldsymbol{Y}$ 均为向量,若 $\boldsymbol{X}$、$\boldsymbol{Y}$ 长度分别为 m、n,则 $\boldsymbol{Z}$ 为 $m\times n$ 的矩阵,即[m,n]=size(Z),则网格线的顶点为(X_j,Y_i,Z_{ij})。

(5) contour(X,Y,Z,n)。其中,n 为所绘制的图形等值线的条数。

(6) contour(X,Y,Z,v)。其中,v 为向量,向量的长度为等值线的条数,并且等值线的值为对应的向量的元素值。

(7) surf(…,LineSpec)。其中,LineSpec 用来定义等值线的线型。

【例 3-21】 等高线绘制。编程如下:

```
clc,clear,close all
[X,Y,Z] = peaks;
figure(1),
contour(X,Y,Z)
title('peaks 等高线等值线图 1')
figure(2),
contour(X,Y,Z,[3 5 7 10])
title('peaks 等高线等值线图 2')
figure(3),
contour(Z)
title('peaks 等高线等值线图 3')
figure(4),
contour(Z,10)
title('peaks 等高线等值线图 4')
```

运行程序,输出图形如图 3-24～图 3-27 所示。

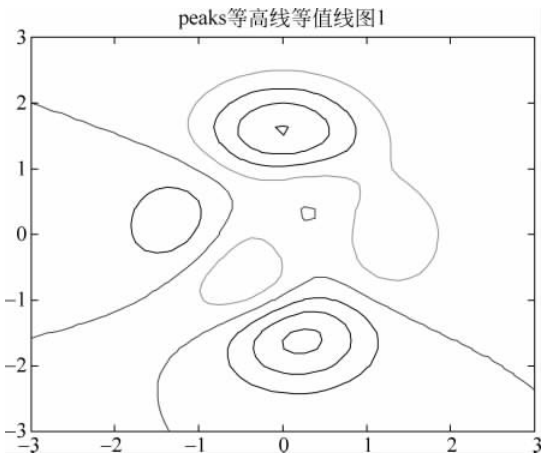

图 3-24 peaks 等高线等值线图 1

3.3.5 特殊图形绘制

MATLAB 对于不同的三维曲面绘制,提供了不同的画图函数,如 slice 切片函数、quiver3 三维箭头标记函数、sphere 等,因此 MATLAB 丰富的图形可视化工具箱函数应用相当广泛。

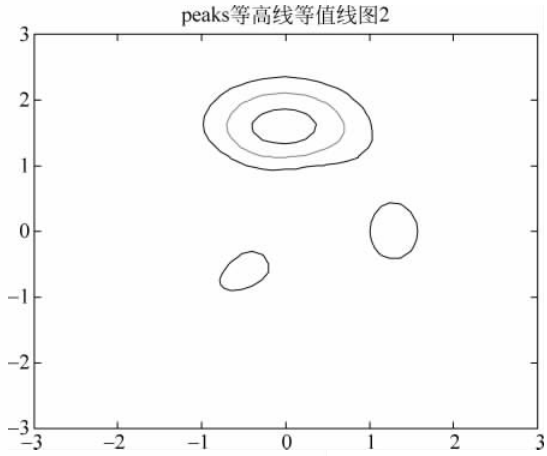

图 3-25　peaks 等高线等值线图 2

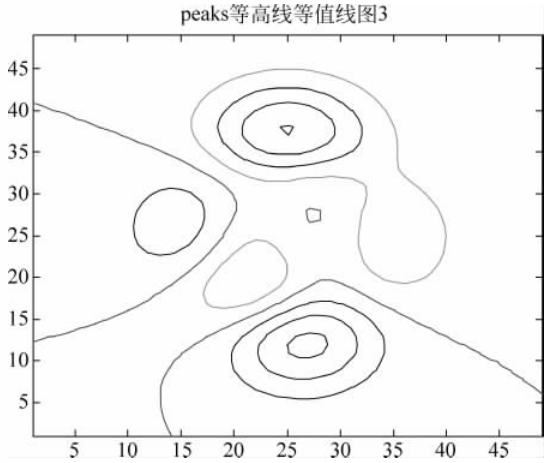

图 3-26　peaks 等高线等值线图 3

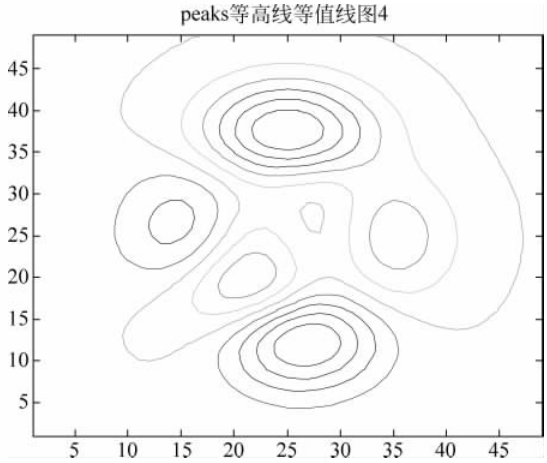

图 3-27　peaks 等高线等值线图 4

【例 3-22】 绘制三元函数 $w=x^2+y^2+z^2$ 的可视化图形。编程如下：

```
clc,clear,close all
clf,x = linspace( - 2,2,40);
y = x;
z = x;
[X,Y,Z] = meshgrid(x,y,z);
w = X.^2 + Y.^2 + Z.^2;
slice(X,Y,Z,w,[ - 1,0,1],[ - 1,0,1],[ - 1,0,1])
colorbar
```

运行程序，输出图形如图 3-28 所示。

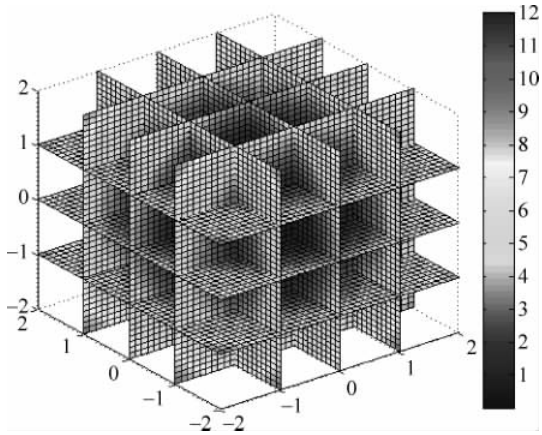

图 3-28 切片图

【例 3-23】 空间曲线及其运动方向的表现。编程如下：

```
clc,clear,close all
clf, t = 0:0.1:1.5;
Vx = 2 * t;Vy = 2 * t.^2;Vz = 6 * t.^3 - t.^2;
x = t.^2;y = (2/3) * t.^3;z = (6/4) * t.^4 - (1/3) * t.^3;      % 由速度得到曲线
plot3(x,y,z,'r. - '),hold on                                    % 画飞行轨迹
% 算数值梯度,也就是重新计算数值速度矢量,这只是为了编程的方便,不是必须的
Vx = gradient(x);Vy = gradient(y);Vz = gradient(z);
quiver3(x,y,z,Vx,Vy,Vz),grid on                                 % 画速度矢量图
xlabel('x'),ylabel('y'),zlabel('z')
```

运行程序，输出图形如图 3-29 所示。

【例 3-24】 用 sphere 指令绘制地球表面的气温分布。编程如下：

```
clc,clear,close all
[a,b,c] = sphere(40);
t = max(max(abs(c))) - abs(c);
surf(a,b,c,t);
axis('equal'),
colormap('hot'),
shading flat,
colorbar
```

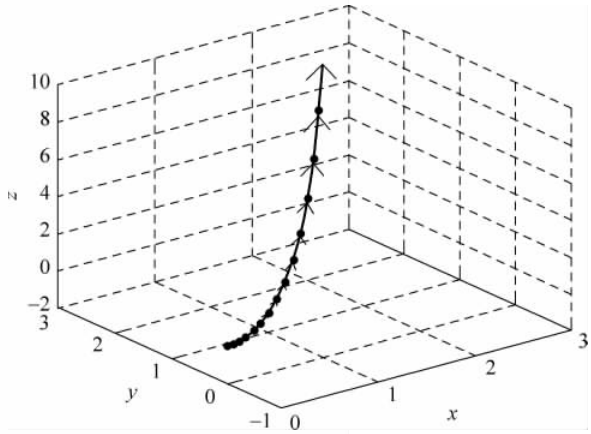

图 3-29　曲线指向图

运行程序，输出图形如图 3-30 所示。

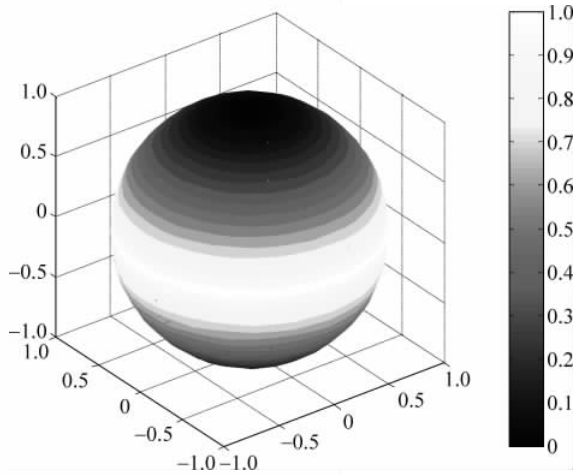

图 3-30　地球表面气温分布图

【例 3-25】 旋转曲面的生成：柱面指令 cylinder 和光照控制指令 surfl。编程如下：

```
clc,clear,close all
x = 0:0.1:10;
z = x;y = 1./(x.^3 - 2. * x + 4);
[u,v,w] = cylinder(y);
surfl(u,v,w,[45,45]);
shading interp
```

运行程序，输出图形如图 3-31 所示。

【例 3-26】 绘制马鞍面 $z=\frac{x^2}{2}-\frac{y^2}{2}$ 的图形。程序调用如下：

```
ezmesh('x^2/2 - y^2/2')
```

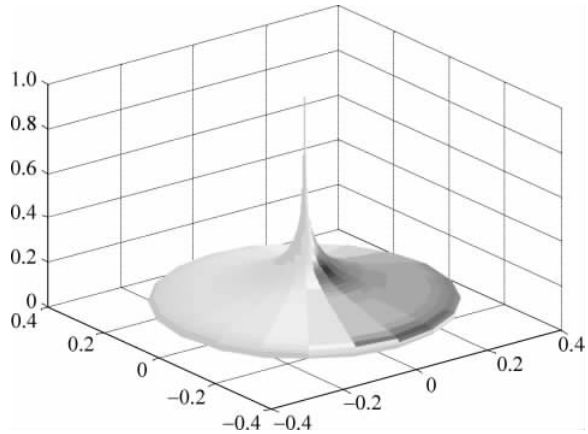

图 3-31　旋转曲面

运行程序，输出图形如图 3-32 所示。

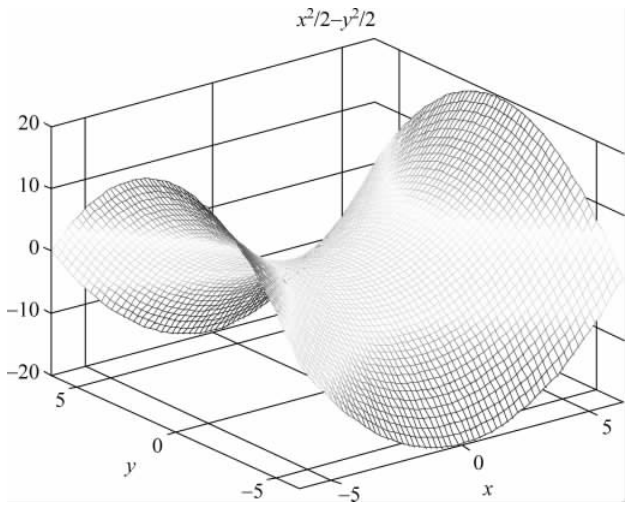

图 3-32　马鞍面

【例 3-27】 特殊图形的绘制。编程如下：

```
clc,clear,close all
x = [1:10];
y = [5 6 3 4 8 1 10 3 5 6];
subplot(2,3,1),
bar(x,y),axis([1 10 1 11])
subplot(2,3,2),
hist(y,x),axis([1 10 1 4])
subplot(2,3,3),
stem(x,y,'k'),axis([1 10 1 11])
subplot(2,3,4),
stairs(x,y,'k'), axis([1 10 1 11])
subplot(2,3,5),
```

```
x = [1 3 0.5 5];explode = [0 0 0 1];pie(x,explode)
subplot(2,3,6),
z = 0:0.1:100; x = sin(z);y = cos(z). * 10;
plot3(x,y,z)
```

运行程序,输出图形如图 3-33 所示。

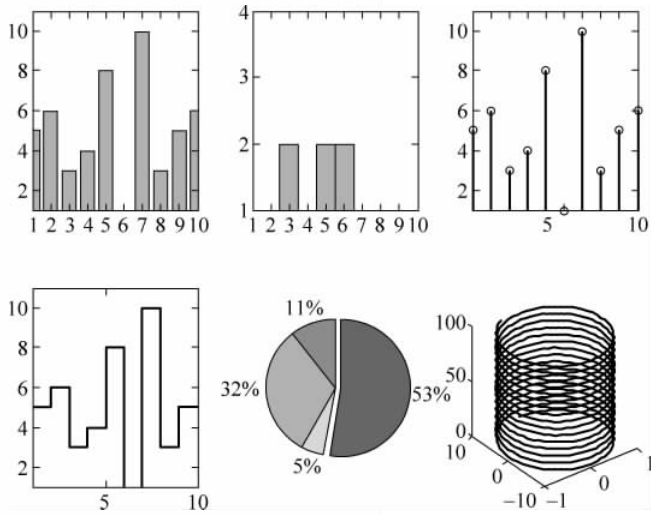

图 3-33　特殊图形绘制

3.4　四维图形可视化

3.4.1　用颜色描述第四维

在前面章节中介绍到,用 mesh 和 surf 等指令所绘制的图像,在未给出颜色参量的情况下,图像的颜色是沿着 z 轴的数据变化的。例如,surf(X,Y,Z)与 surf(X,Y,Z,Z)两个指令的执行效果是相同的。将颜色施加于 z 轴能够产生色彩亮丽的图画,但由于 z 轴已经存在,因此,它并不提供新的信息。因此,为更好地利用颜色,则可以考虑使用颜色来描述不受三个轴影响的数据的某些属性。为此,需要赋给三维作图函数的颜色参量所需要的“第四维”的数据。

如果作图函数的颜色参量是一个向量或矩阵,那么就用作颜色映像的下标。这个参量可以是任何实向量或与其参量维数相同的矩阵。

【例 3-28】 使用颜色描述第四维示例。

输入命令如下:

```
[X,Y,Z] = peaks(30);
R = sqrt(X.^2 + Y.^2);
subplot(1,2,1);surf(X,Y,Z,Z);
axis tight
```

```
subplot(1,2,2);surf(X,Y,Z,R);
axis tight
```

程序运行结果如图 3-34 所示。

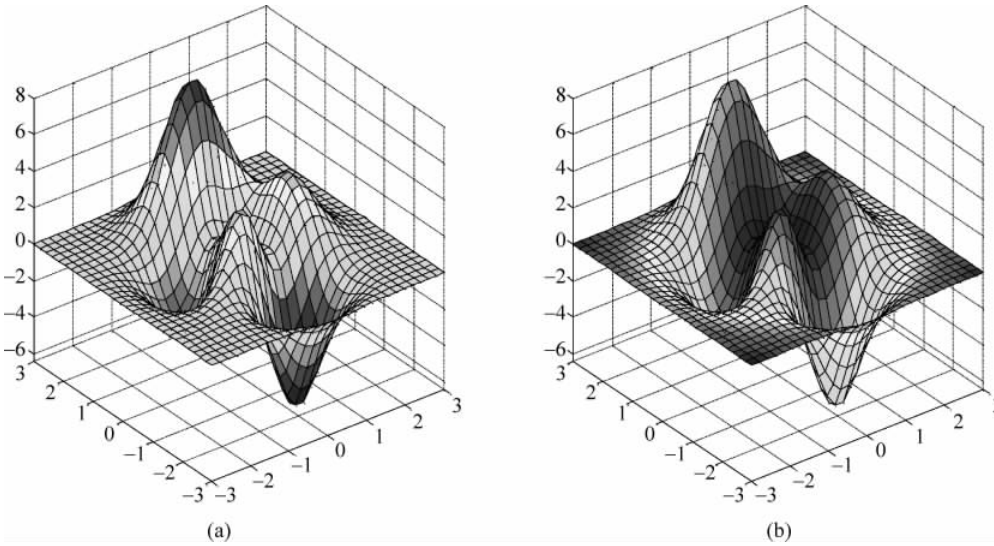

图 3-34　使用颜色描述第四维示例

其中，在坐标系中描述一个面需要三维数据，而另一维数据描述空间中的点的坐标值，则使用不同的颜色表现出来。在图 3-34(a)中，第四维数据为 Z；在图 3-34(b)中，第四维数据为 R。在图上可以看到两者的颜色分布发生了明显的变化。

3.4.2　其他函数

除了 surf()函数外，mesh()和 pcolor()函数也可以将第四维的数据附加到颜色属性上，并在图像中表现出来。各函数的句法列表如表 3-9 所示。

表 3-9　其他指令的句法和功能

句 法 格 式	说　　明
surf(X,Y,Z,fun(X,Y,Z))	根据函数 fun(X,Y,Z)来附加颜色数据
surf(X,Y,Z)＝surf(X,Y,Z,Z)	默认动作，附加颜色数据于 z 轴
surf(X,Y,Z,X)	附加颜色数据于 x 轴
surf(X,Y,Z,Y)	附加颜色数据于 y 轴
surf(X,Y,Z,X.^2＋Y.^2)	xoy 平面上距原点一定的距离附加颜色数据
surf(X,Y,Z,del2(Z))	根据曲面的拉氏函数值附加颜色数据
[dZdx,dZdy]＝gradient(Z);surf(X,Y,Z,abs(dZdx))	根据 x 轴方向的曲面斜率附加颜色数据
dz＝sqrt(dZdx.^2＋dZdy.^2);surf(X,Y,Z,dz)	根据曲面斜率大小附加颜色数据

除了表 3-9 所列出的函数之外，slice 函数也可以通过颜色来表示存在于第四维空间中的值。其具体句法格式如下：

(1) slice(V,nx,ny,nz)。显示三元函数 $V(X,Y,Z)$确定的立体图在 x 轴、y 轴、z 轴方向上的若干点(对应若干平面)的切片图,各点的坐标由数量向量 sx、sy、sz 指定,其中 V 为大小为 $m\times n\times p$ 的三维数组,默认值为 $X=1:m$、$Y=1:n$、$Z=1:p$。

(2) slice(X,Y,Z,V,nx,ny,nz)。显示三元函数 $V(X,Y,Z)$确定的立体图在 x 轴、y 轴、z 轴方向上的若干点(对应若干平面)的切片图。若函数 $V(X,Y,Z)$中有一个变量 X 取定值 X_0,则函数 $V(X_0,Y,Z)$为 $X=X_0$ 立体面的切面图(将该切面通过颜色表示 V 的值),各点的坐标由数量向量 sx、sy、sz 指定。参量 X、Y、Z 均为三维数组,用于指定立方体 V 的每点的三维坐标。

(3) slice(V,XI,YI,ZI)。显示由参量矩阵 ***XI***、***YI***、***ZI*** 确定的立体图的切片图,参量 ***XI***、***YI***、***ZI*** 定义了一个曲面,同时会在曲面的点上计算立体图 V 的值。需要注意的是,***XI***、***YI***、***ZI*** 必须为同型矩阵。

(4) slice(X,Y,Z,V,XI,YI,ZI)。沿着由矩阵 ***XI***、***YI***、***ZI*** 定义的曲面穿过立体图 V 的切片图。

(5) slice(…,'method')。通过 method 来指定内插值的方法,method 可取 linear、cubic、nearest。linear 指定的内插值方法为三次线性内插值(若未指定,此即为默认值),cubic 指定使用三次立方内插值法,nearest 指定使用最近点内插值法。

3.5 MATLAB 动画设计

MATLAB 动画由一系列的帧图像组成,MATLAB 图形可视化工具箱为动画化设计提供了必备基础。以下将结合几个动画例子来说明。

【例 3-29】 绘制二维振动的弹簧的动画。编程如下:

```
clc,clear,close all
animinit('onecart1 Animation')
axis([-2 6 -10 10]); hold on; u=2;
xy=[ 0 0 0 0 u u u+1 u+1 u u;
-1.2 0 1.2 0 0 1.2 1.2 -1.2 -1.2 0];
x=xy(1,:);y=xy(2,:);
% Draw the floor under the sliding masses
plot([-10 20],[-1.4 -1.4],'b-','LineWidth',2);
hndl=plot(x,y,'b-','EraseMode','XOR','LineWidth',2);
set(gca,'UserData',hndl);
for t=1:0.025:100;
    u=2+exp(-0.00*t)*cos(t);
    x=[0 0 0 0 u u u+1 u+1 u u];
    hndl=get(gca,'UserData');
    set(hndl,'XData',x);
    drawnow
end
```

运行程序,输出图形如图 3-35 所示。

【例 3-30】 曲面 $z=\dfrac{\sin(\sqrt{x^2+y^2})}{\sqrt{x^2+y^2}}\cdot\sin\theta$ 随着 θ 的变化,$z=\dfrac{\sin(\sqrt{x^2+y^2})}{\sqrt{x^2+y^2}}$图形的

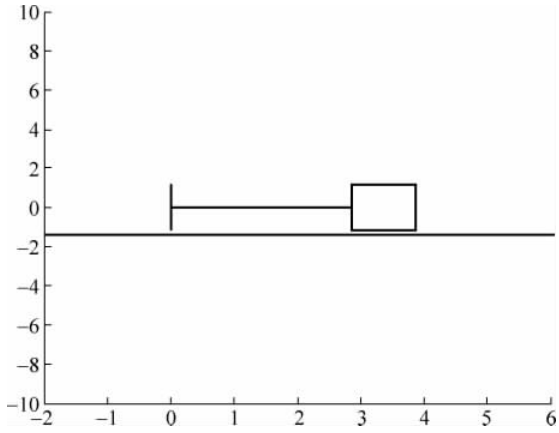

图 3-35　振动的弹簧

实时变化动画显示。编程如下：

```
clc,clear,close all
x = -8:0.5:8;                          %定义曲面
[XX,YY] = meshgrid(x);
r = sqrt(XX.^2 + YY.^2) + eps;
Z = sin(r)./r;
surf(Z);                               %画出帧
theAxes = axis;                        %保存坐标值,使得所有帧都在同一坐标系中
fmat = moviein(20);                    %创建一个动画的矩阵,保存 20 帧
for j = 1:20;                          %循环创建动画数据
    surf(sin(2 * pi * j/20) * Z,Z)     %画出每一步的曲面
    axis(theAxes)                      %使用相同的坐标系
    fmat(:,j) = getframe;              %复制帧到矩阵 fmat 中
end
movie(fmat,10)
```

运行程序，输出图形如图 3-36 和图 3-37 所示。

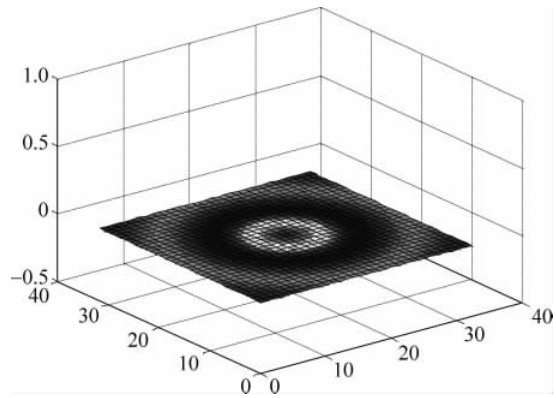

图 3-36　动画视图 1

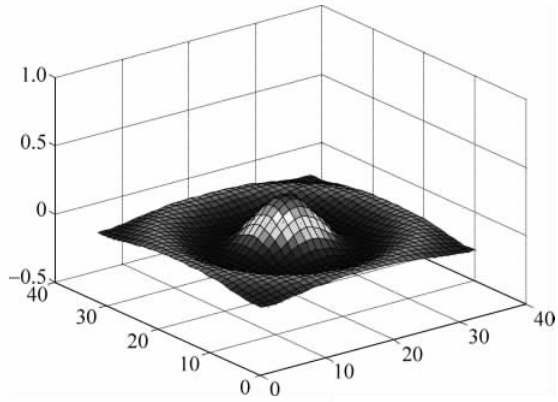

图 3-37　动画视图 2

【例 3-31】 地球、月球、嫦娥一号运动仿真模拟。编程如下：

```
clc,clear,close all
figure('name','嫦娥一号与月亮、地球关系','color',[1 1 1]);     %设置标题名字
s1 = [0:.01:2 * pi];
hold on;axis equal;                                  %建立坐标系
axis off                                             %除掉 Axes
r1 = 10;                                             %月亮到地球的平均距离
r2 = 3;                                              %嫦娥一号到月亮的平均距离
w1 = 1;                                              %设置月亮公转角速度
w2 = 12                                              %设置嫦娥一号绕月亮公转角速度
t = 0;                                               %初始时刻为 0
pausetime = .002;                                    %设置暂停时间
sita1 = 0;sita2 = 0;                                 %设置开始它们都在水平线上
set(gcf,'doublebuffer','on')                         %消除抖动
plot( - 20,18,'color','r','marker','.','markersize',40);
text( - 17,18,'地球');                               %对地球进行标识
p1 = plot( - 20,16,'color','b','marker','.','markersize',20);
text( - 17,16,'月亮');                               %对月亮进行标识
p1 = plot( - 20,14,'color','w','marker','.','markersize',13);
text( - 17,14,'嫦娥一号');                           %对嫦娥一号进行标识
plot(0,0,'color','r','marker','.','markersize',60);          %画地球
plot(r1 * cos(s1),r1 * sin(s1));                     %画月亮公转轨道
set(gca,'xlim',[ - 20 20],'ylim',[ - 20 20]);
p1 = plot(r1 * cos(sita1),r1 * sin(sita1),'color','b','marker','.','markersize',30);
                                                     %画月亮初始位置
l1 = plot(r1 * cos(sita1) + r2 * cos(s1),r1 * sin(sita1) + r2 * sin(s1));
                                                     %画嫦娥一号绕月亮公转轨道
p2x = r1 * cos(sita1) + r2 * cos(sita2);p2y = r1 * sin(sita1) + r2 * sin(sita2);
p2 = plot(p2x,p2y,'w','marker','.','markersize',20);                %画嫦娥一号的初始位置
orbit = line('xdata',p2x,'ydata',p2y,'color','r');                  %画嫦娥一号的运动轨迹
while 1
    set(p1,'xdata',r1 * cos(sita1),'ydata',r1 * sin(sita1));        %设置月亮的运动过程
```

```
    set(l1,'xdata',r1 * cos(sita1) + r2 * cos(s1),'ydata',r1 * sin(sita1) + r2 * sin(s1));
                                                    % 设置嫦娥一号绕月亮的公转轨道的运动过程
    ptempx = r1 * cos(sita1) + r2 * cos(sita2);ptempy = r1 * sin(sita1) + r2 * sin(sita2);
    set(p2,'xdata',ptempx,'ydata',ptempy); % 设置嫦娥一号的运动过程
    p2x = [p2x ptempx];p2y = [p2y ptempy];
    set(orbit,'xdata',p2x,'ydata',p2y); % 设置嫦娥一号运动轨迹的显示过程
    sita1 = sita1 + w1 * pausetime;        % 月亮相对地球转过的角度
    sita2 = sita2 + w2 * pausetime;        % 嫦娥一号相对月亮转过的角度
    pause(pausetime);                      % 暂停一会
    drawnow
end
```

运行程序,输出图形如图 3-38 和图 3-39 所示。

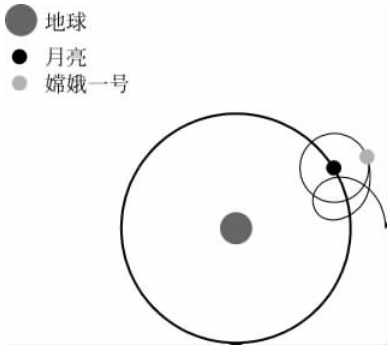

图 3-38　地球、月球、嫦娥一号运动视图 1

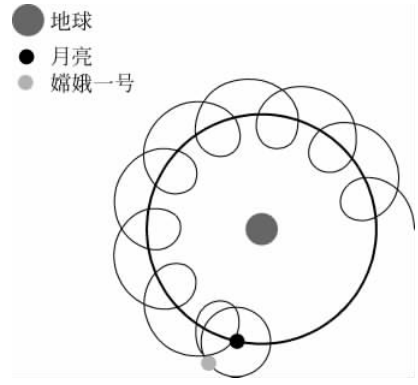

图 3-39　地球、月球、嫦娥一号运动视图 2

【例 3-32】 地球、月球、卫星运动仿真模拟动画制作。编程如下:

```
clc,clear,close all
h = figure('numbertitle','off','name','卫星绕地球旋转演示动画');        % 设置标题名字
s1 = 0:.01:2 * pi;
hold on;
axis equal;                                                    % 建立坐标系
axis off;                                                      % 除掉 Axes
r1 = 10;                                                       % 地球到太阳的平均距离
r2 = 3;                                                        % 卫星的轨道半径
w1 = 1;                                                        % 设置地球公转角速度
w2 = 12;                                                       % 设置卫星绕地球公转角速度
t = 0;                                                         % 初始时刻
pausetime = .002;                                              % 设置视觉暂留时间
sita1 = 0;
sita2 = 0;                                                     % 设置开始它们都在水平线上
set(gcf,'doublebuffer','on')                                   % 消除抖动
plot( - 20,18,'color','r','marker','.','markersize',40);
text( - 17,18,'太阳');                                         % 对太阳进行标识
plot( - 20,16,'color','b','marker','.','markersize',20);
text( - 17,16,'地球');                                         % 对地球进行标识
```

```
plot( - 20,14,'color','w','marker','.','markersize',13);
text( - 17,14,'卫星');                                    %对卫星进行标识
plot(0,0,'color','r','marker','.','markersize',60);  %画太阳
plot(r1 * cos(s1),r1 * sin(s1));                          %画地球公转轨道
set(gca,'xlim',[ - 20 20],'ylim',[ - 20 20]);
p1 = plot(r1 * cos(sita1),r1 * sin(sita1),'color','b','marker','.','markersize',30);
                                                          %地球初始位置
l1 = plot(r1 * cos(sita1) + r2 * cos(s1),r1 * sin(sita1) + r2 * sin(s1));
                                                          %画卫星绕地球的公转轨道
p2x = r1 * cos(sita1) + r2 * cos(sita2);
p2y = r1 * sin(sita1) + r2 * sin(sita2);
p2 = plot(p2x,p2y,'w','marker','.','markersize',20); %画卫星的初始位置
orbit = line('xdata',p2x,'ydata',p2y,'color','r');    %画卫星的运动轨迹
while 1

    if ~ishandle(h),
        return,
    end
    set(p1,'xdata',r1 * cos(sita1),'ydata',r1 * sin(sita1));          %设置地球的运动过程
    set(l1,'xdata',r1 * cos(sita1) + r2 * cos(s1),'ydata',r1 * sin(sita1) + r2 * sin(s1));
                                                          %设置卫星绕地球的公转轨道的运动过程
    ptempx = r1 * cos(sita1) + r2 * cos(sita2);
    ptempy = r1 * sin(sita1) + r2 * sin(sita2);
    set(p2,'xdata',ptempx,'ydata',ptempy);                %设置卫星的运动过程
    p2x = [p2x ptempx];
    p2y = [p2y ptempy];
    set(orbit,'xdata',p2x,'ydata',p2y);                   %设置卫星运动轨迹的显示过程
    sita1 = sita1 + w1 * pausetime;                       %地球相对太阳转过的角度
    sita2 = sita2 + w2 * pausetime;                       %卫星相对地球转过的角度
    pause(pausetime);                                     %视觉暂停
    drawnow                                               %刷新屏幕,重绘

end
```

运行程序,输出图形如图 3-40 和图 3-41 所示。

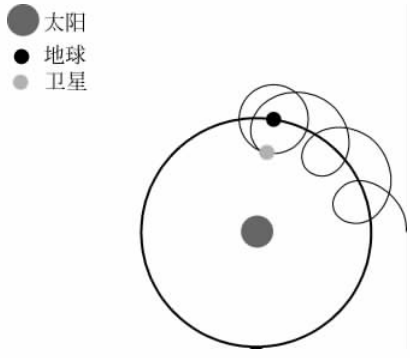

图 3-40　太阳、地球、卫星运动视图 1

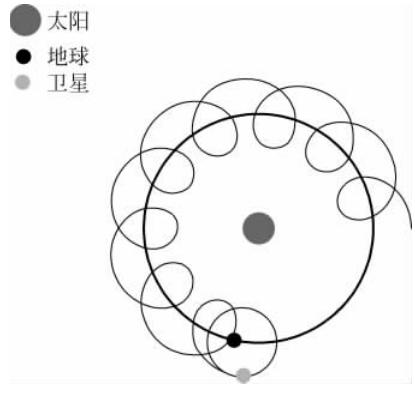

图 3-41　太阳、地球、卫星运动视图 2