第3章 MATLAB高等数学计算

MATLAB在高等数学计算中的应用非常广泛，它是工程科学的基础，具有抽象性、逻辑性，主要涵盖傅里叶变换、拉普拉斯变换、Z 变换、极限、导数、微分、积分与重积分、级数求和与泰勒级数展开、代数方程组、常微分方程、向量代数和插值运算等。为了将抽象的结果进行图形化展示，本章增加了绘图一节，包括二维绘图和三维绘图。

3.1 傅里叶变换与反变换

傅里叶变换是进行信号处理、图像处理、音视频处理的数学工具。

3.1.1 傅里叶变换

语法格式：

```
F = fourier(f,t,w)    % 求时域函数 f(t)的傅里叶变换
```

说明：返回结果 F 是符号变量 w 的函数，省略参数 w 则默认返回结果为 w 的函数；f 为 t 的函数，省略参数 t 时，则默认自由变量为 x。

3.1.2 傅里叶反变换

语法格式：

```
f = ifourier (F)       % 求频域函数 F 的傅里叶反变换 f(t)
f = ifourier (F,w,t) % 求频域函数 F 指定变量 w 和 t 算子的傅里叶反变换 f(t)
```

【例 3-1】 求 $f(t)=\dfrac{1}{t^2+1}$ 的傅里叶变换及 $f(t)$ 的傅里叶反变换。

程序命令：

```
syms t,w;
```

```
F = fourier(1/(t^2 + 1),t,w)        % 傅里叶变换
ft = ifourier(F,t)                  % 傅里叶反变换
f = ifourier(F)                     % 傅里叶反变换默认 x 为自变量
```

结果：

```
F = pi * exp( - abs(w))
ft = 1/(t^2 + 1)
f = 1/(x^2 + 1)
```

3.1.3 快速傅里叶变换

快速傅里叶变换(FFT)是离散傅里叶变换的快速算法，可将一个时域信号变换到频域。

傅里叶变换是将连续时序或信号表示为不同频率的余弦(或正弦)波信号的无限叠加，快速傅里叶变换是提取信号的频谱进行分析，找到信号的详细特征。MATLAB 提供了快速傅里叶变换和傅里叶反变换。快速傅里叶变换函数如下。

1. fft( )函数

语法格式：

```
Y = fft(X)
```

说明：若 X 是向量，则 fft(X)返回该向量的傅里叶变换，若 X 是矩阵，则 fft(X)将 X 的各列视为向量，并返回每列的傅里叶变换。若 X 是一个多维数组，则 fft(X)将沿大小不等于 1 的第一个数组维度的值视为向量，并返回每个向量的傅里叶变换。

```
Y = fft(X,n)          % n 为长度
```

说明：

① 若 X 是向量的长度小于 n，则 X 尾部补零到长度 n，否则截断 X 以达到长度 n。

② 若 X 是矩阵，则按列向量处理。

③ 若 X 为多维数组，则大小不等于 1 的第一个数组维度的处理与向量相同。

```
Y = fft(X,n,dim)   % 当 X 为矩阵时,在指定的 dim 维上进行傅里叶变换(dim = 1 按列,dim = 2 按行)
```

2. fft2( )函数

语法格式：

```
Y = fft2(X)
```

说明：使用快速傅里叶变换算法返回矩阵的二维傅里叶变换，这等同于计算 fft(fft(X).').'，如果 X 是一个多维数组，fft2 将采用高于 2 的每个维度的二维变换。输出 Y 的大

小与 X 相同。

```
Y = fft2(X,m,n)
```

说明：将截断 X 或用尾随零填充 X，以便在计算变换之前形成 m×n 矩阵。Y 是 m×n 矩阵。如果 X 是一个多维数组，fft2 将根据 m 和 n 决定 X 的前两个维度的形状。

3.1.4 快速傅里叶反变换

通过快速傅里叶反变换将频域信号反变换回原本的时域信号。快速傅里叶反变换函数如下。

1. ifft( )函数

语法格式：

```
X = ifft(Y)
```

说明：使用快速傅里叶变换算法计算 Y 的离散傅里叶反变换。X 与 Y 的大小相同。

① 若 Y 是向量，则 ifft(Y) 返回该向量的反变换；

② 若 Y 是矩阵，则 ifft(Y) 返回该矩阵每一列的反变换。

③ 若 Y 是多维数组，则 ifft(Y) 将大小不等于 1 的第一个维度上的值视为向量，并返回每个向量的反变换。

```
X = ifft(Y,n)
```

说明：通过用尾随零填充 Y 以达到长度 n，返回 Y 的 n 点傅里叶反变换。

```
X = ifft(Y,n,dim)
```

说明：返回沿维度 dim 的傅里叶反变换。若 Y 是矩阵，则 ifft(Y,n,2) 返回每一行的 n 点反变换。

2. ifft2( )函数

语法格式：

```
X = ifft2(Y)
```

说明：使用快速傅里叶变换算法返回矩阵的二维离散傅里叶反变换。

若 Y 是一个多维数组，则 ifft2 计算大于 2 的每个维度的二维反变换。输出 X 的大小与 Y 相同。

```
X = ifft2(Y,m,n)
```

说明：在计算反变换之前截断 Y 或用尾随零填充 Y，以形成 m×n 矩阵。使得 X 也是

m×n矩阵。若Y是一个多维数组，ifft2将根据m和n决定Y的前两个维度的形状。

【例3-2】 使用二种方法计算卷积，分别使用conv()和傅里叶反变换计算下列矩阵的卷积。

$$\boldsymbol{A}=\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}\qquad \boldsymbol{B}=\begin{bmatrix}1&1\\1&1\end{bmatrix}$$

说明：卷积是通过两个函数 f 和 g 生成第三个函数的一种数学算子，表征函数 f 与经过翻转和平移 g 的重叠部分的累积。

```
A = ones(3);
B = ones(2);
A(4,4) = 0;                    % 卷积一般变成 4 阶矩阵以查看所有信息，不满 4 阶用 0 补充
B(4,4) = 0;
C1 = conv2(A,B)
C2 = ifft2(fft2(A). * fft2(B))
```

结果：

```
C1 =
     1     2     2     1     0     0     0
     2     4     4     2     0     0     0
     2     4     4     2     0     0     0
     1     2     2     1     0     0     0
     0     0     0     0     0     0     0
     0     0     0     0     0     0     0
     0     0     0     0     0     0     0
C2 =
     1     2     2     1
     2     4     4     2
     2     4     4     2
     1     2     2     1
```

【例3-3】 设信号为：y＝sin(2π500t)＋ sin(2π600t)＋ sin(2π520t)根据采样点1000，采样周期1/10000，采用FFT分析频谱并绘图。

```
N = 1000;                                  % 采样点数
dt = 1/10000;                              % 采样周期
t = 0:dt:(N - 1) * dt;
y = sin(2 * pi * 500 * t) + sin(2 * pi * 600 * t) + sin(2 * pi * 520 * t);     % 信号
subplot(1,2,1);plot(t,y)                   % 绘制时域信号
title('时域信号')
xlabel('t (秒)')
ylabel('|f(t)|')
PH2 = (fft(y));                            % 将时域信号 fft 变成频域
P2 = (PH2/N);                              % 幅值修正
P1 = P2(1:N/2 + 1);                        % 选取前半部分(fft 变换后为对称的双边谱)
```

```
P1(2:end-1) = 2*P1(2:end-1);
f = 10000*(0:(N/2))/N;
subplot(1,2,2);plot(f,abs(P1))          % 绘制频域信号
title('频域信号')
xlabel('f (赫兹)')
ylabel('|P1(f)|')
```

结果如图 3.1 所示。

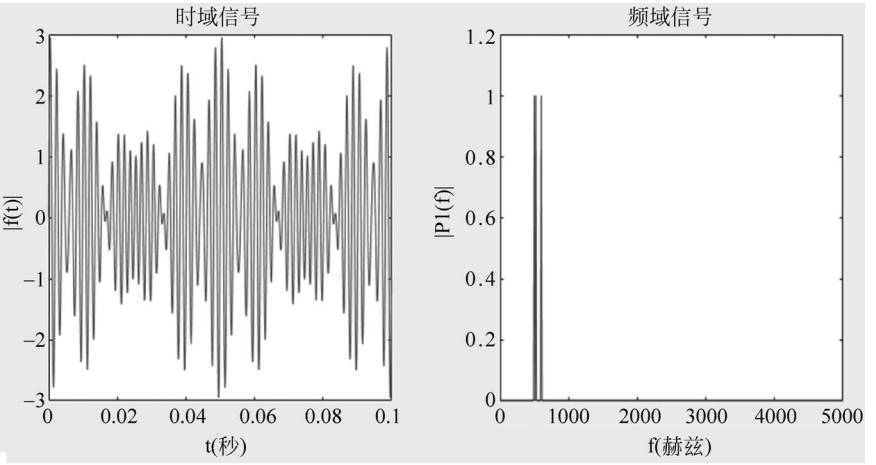

图 3.1　FFT 结果

3.2　拉普拉斯变换与反变换

拉普拉斯变换是工程数学中常用的一种积分变换，它是将时间域变换到复数 s 域的方法，自动控制理论中的传递函数大部分使用 s 域表示。

3.2.1　拉普拉斯变换

语法格式：

```
F = laplace(f,t,s)                      % 求时域函数 f 的拉普拉斯变换 F
```

说明：拉普拉斯变换也称拉式变换，返回结果 F 为 s 的函数。当参数 s 省略时，返回结果 F 默认为 s 的函数；f 为 t 的函数，当参数 t 省略时，默认自由变量为 t。

3.2.2　拉普拉斯反变换

语法格式：

```
f = ilaplace(F,s,t)                    % 求 F 的拉普拉斯反变换 f
```

说明：把 s 域转换成 t 域的函数。

【例 3-4】 求 $f(t)=\cos(at)+\sin(at)$的拉普拉斯变换和反变换。

程序命令：

```
syms  a,t,s;
F1 = laplace(sin(a * t) + cos(a * t),t,s)
f = ilaplace(F1)
fx = ilaplace(sym('1/s'))
```

结果：

```
F1 = a/(a^2 + s^2) + s/(a^2 + s^2)
 f = cos(a * t) + sin(a * t)
fx = 1
```

3.3 Z 变换与反变换

Z 变换是将离散系统的时域利用差分方程转化到频域数学模型的一种方法，离散系统传递函数使用 Z 变换表示。

3.3.1 Z 变换

Z 变换是对连续系统进行离散数学变换，常用于求解线性时不变差分方程的解。

语法格式：

```
Z = ztrans(f)                          % 求 Z 变换
```

3.3.2 Z 反变换

将离散系统变换成连续系统的变换称为 Z 反变换。

语法格式：

```
fz = iztrans(z)                        % 求 Z 反变换
```

【例 3-5】 求 $f(x)=x\mathrm{e}^{-10x}$ 的 Z 变换和 $f(z)=\dfrac{z(z-1)}{z^2+2z+1}$的反变换。

程序命令：

```
syms x,k,z;
f = x * exp( - x * 10);                % 定义表达式
```

```
F = ztrans(f)                          % 求 Z 变换
Fz = z * (z - 1)/(z^2 + 2 * z + 1);    % 定义 Z 反变换表达式
F1 = iztrans(Fz)
```

结果：

```
F  = (z * exp(10))/(z * exp(10)  -  1)^2
F1  = 3 * ( - 1)^n  +  2 * ( - 1)^n * (n  -  1)
```

3.4 求极限

极限是一种变化状态的描述。若函数中的某一个变量在变大或变小的过程中，逐渐趋近于某一个确定的数值 A，但不能到达 A，称数值 A 为极限值。在控制系统稳定性分析中，稳态时间就是指在 2%或 5%误差状态下，到达控制点的时间。

语法格式：

```
limit(f,x,a)   % 求符号函数 f(x)的极限值,即计算当变量 x 趋近于常数 a 时 f(x)函数的极限值
limit(f,a)   % 求符号函数 f(x)的极限值,由于没有指定符号函数 f(x)的自变量,则使用该格式。
             % 此时,符号函数 f(x)的变量使用函数 findsym(f)确定默认自变量,变量 x 趋近于 a
limit(f)   % 求符号函数 f(x)的极限值.符号函数 f(x)的变量使用函数 findsym(f)确定默认变量;
           % 没有指定变量的目标值时,系统默认变量趋近于 0,即 a = 0 的情况
limit(f,x,a,'right')        % 求符号函数 f 的极限值,'right'表示变量 x 从右边趋近于 a
limit(f,x,a,'left')         % 求符号函数 f 的极限值,'left'表示变量 x 从左边趋近于 a
limit(f,x,a,'inf')          % 求符号函数 f 的极限值,'inf'表示变量 x 趋近于无穷
```

【例 3-6】 已知表达式 F_1 和 F_2，求 F_1 和 F_2 的极限。

$$F_1 = \lim_{x \to 0} \frac{x(e^{\sin x} + 1) - 2(e^{\tan x} - 1)}{\sin^3 x}$$

$$F_2 = \lim_{x \to \infty} \frac{\sqrt{x + \sqrt{x}} - \sqrt{x}}{\sin(\pi/6)}$$

程序命令：

```
syms x;                          % 定义 x 为符号变量
f1 = (x * (exp(sin(x)) + 1) - 2 * (exp(tan(x)) - 1))/sin(x)^3;
f2 = (sqrt(x + sqrt(x)) - sqrt(x))/ sin(pi/6);
F1 = limit(f1,x,0)               % 求函数的极限
F2 = limit(f2.x,inf)
```

结果：

```
F1 =  - 1/2
F2 = 1
```

3.5 求导数

求导数就是求函数的平均变化率,利用 MATLAB 函数使得求导数变得非常简单。

3.5.1 语法格式

求导数语法格式:

```
diff(s)    % 没有指定变量和导数阶数,则系统按 findsym()函数确定的默认变量对符号表达式 s 求
           % 一阶导数
diff(s,'v')      % 以 v 为自变量,对符号表达式 s 求一阶导数
diff(s,n)        % 按 findsym()函数确定的默认变量对符号表达式 s 求 n 阶导数,n 为正整数
diff(s,'v',n)    % 以 v 为自变量,对符号表达式 s 求 n 阶导数
```

3.5.2 求导数案例

【例 3-7】 编写程序求函数 f_1 和 f_2 的导数。

$$f_1 = \sin x^2 + 3x^5 + \sqrt{(x+1)^3}$$

$$f_2 = \cos x^2 + \arctan(\ln x)$$

程序命令:

```
syms x;                              % 定义符号变量
f1 = sin(x)^2 + 3 * x^5 + sqrt((x + 1)^3);
f2 = cos(x)^2 + atan(log(x))
F1 = diff(f1,x)                      % 求函数的极限
F2 = diff(f2,x)
```

结果:

```
F1 = 2 * cos(x) * sin(x) + (3 * (x + 1)^2)/(2 * ((x + 1)^3)^(1/2)) + 15 * x^4
F2 = 1/(x * (log(x)^2 + 1)) - 2 * cos(x) * sin(x)
```

【例 3-8】 已知函数 f_1,求 f_1 的二阶导数 F_1 及在 $x=2$ 的值 X。

$$f_1 = \frac{5}{\ln(1+x)}$$

$$F_1 = \frac{\mathrm{d}^2 f}{\mathrm{d}x}$$

$$X = \left.\frac{\mathrm{d}^2 f}{\mathrm{d}x}\right|_{x=2}$$

程序命令:

```
syms x ;
f1 = 5/log(1 + x);
F1 = diff(f1,x,2)
x = 2;
X = eval(F1)
```

结果：

```
F1 = 5/(log(x + 1)^2 * (x + 1)^2) + 10/(log(x + 1)^3 * (x + 1)^2)
X = 1.2983
```

3.6 求积分

MATLAB 分别利用 int 函数和 quad 函数求积分，int 函数可获得解析解，quad 函数得到的是小梯形面积求和的解。

3.6.1 使用 int 函数求积分

int 函数根据解析的方法求解，对定积分和不定积分均可得到解析解，无任何误差，但速度稍慢。

语法格式：

```
int(s)    % 没有指定积分变量和积分阶数时，系统按 findsym()函数确定的默认变量对被积函数或
          % 符号表达式 s 求不定积分
int(s,v)        % 以 v 为自变量，对被积函数或符号表达式 s 求不定积分
int(s,v,a,b)    % 求以 v 为自变量的符号表达式 s 的定积分，a,b 分别表示定积分的下限和上限
```

说明：int(s,v,a,b)求被积函数在区间[a,b]上的定积分。a 和 b 可以是两个具体的数，也可以是符号表达式，还可以是无穷(Inf)。当函数 f 关于变量 x 在闭区间[a,b]上可积时，函数返回一个定积分结果；当 a,b 中有一个是 Inf 时，函数返回一个广义积分；当 a,b 中有一个为符号表达式时，函数返回一个符号函数。

【例 3-9】 求函数$\int \cos 2x \sin 3x \, dx$ 的不定积分。

程序命令：

```
syms x;
f1 = cos(2 * x) * sin(3 * x)
F1 = int(f1)
```

结果：

```
F1  = 2 * cos(x)^3 - cos(x) - (8 * cos(x)^5)/5
```

【例 3-10】 求表达式 f_1 和 f_2 的定积分。

$$f_1 = \int_{-T/2}^{T/2} (AT^2 + \mathrm{e}^{-\mathrm{j}xt})\,\mathrm{d}t$$

$$f_2 = \int_1^{\mathrm{e}} \frac{1}{x^2} \ln x\,\mathrm{d}x$$

程序命令：

```
syms A,t,T,x;
f1 = A * T^2 + exp( - j * x * t)
f2 = log(x)/x^2
F1 = int(f1,t, - T/2, T/2);
F2 = simplify(F1)
F3 = int(f2,x,1,exp(1));
F4 = eval(F3)
```

结果：

```
f1 =  A * T^2  +  exp( - t * x * 1i)
f2 = log(x)/x^2
F2 = A * T^3  +  (2 * sin((T * s)/2))/x
F4 = 0.2642
```

【例 3-11】 求 f_1 和 f_2 两个表达式的二重积分。

$$f_1(x) = \iint (x + y)\mathrm{e}^{-xy}\,\mathrm{d}x\,\mathrm{d}y$$

$$f_2(x) = \iint \log_{\mathrm{e}}(x)/y^2 + y^2/x^2\,\mathrm{d}x\,\mathrm{d}y, \quad 1/2 \leqslant x \leqslant 2, \quad 1 \leqslant y \leqslant 2$$

程序命令：

```
syms x y;
f1 = (x + 1) * exp( - x * y);
F1 = int(int(f1,'x'),'y')
f2 = log(x)/y^2 + y^2/x^2;
F2 = int(int(f2,x,1/2,2),y,1,2)
F3 = eval(F2)
```

结果：

```
F1 = (exp( - x * y) * (x  +  y))/(x * y)
F2 = (5 * log(2))/4  +  11/4
F3 = 3.6164
```

3.6.2 使用 quadl 函数求积分

计算一元函数的数值积分使用 quad 函数和 quadl 函数，它们通过小梯形的面积求和得

到积分值,而不是通过解析的方法。当有计算精度限制时,计算速度比 int 函数快。它们均采用遍历的自适应法计算函数的数值积分,quadl 是高阶数值积分,quad 是低阶数值积分,它们只能求定积分。

语法格式:

```
[Q,Fcnt] = quad(function,a,b)        % 求 function 的积分
```

其中,function 为被积函数(形式为函数句柄/匿名函数),a,b 分别是积分上下限,[Q,Fcnt]为返回数值积分的结果和函数计算的次数。

【例 3-12】 求表达式 $f_1=\int_0^2 \frac{2}{x^3-x+2}\mathrm{d}x$ 的定积分。

程序命令:

```
F = @(x) 2./(x.^3 - x + 2);
[Q ,Fcnt] =  quadl(F,0,2)
```

结果:

```
Q = 1.7037
Fcnt = 48
```

【例 3-13】 已知 $\boldsymbol{w}=[\pi/2,\pi,3\pi/2]$;$\boldsymbol{K}=[\pi/2-1,-2,-3\pi/2-1]$,求表达式 Y 的定积分。

$$Y=\left(\int_0^{\boldsymbol{w}(1)} x^2\cos(x)\mathrm{d}x-\boldsymbol{K}(1)\right)^2+\left(\int_0^{\boldsymbol{w}(2)} x^2\cos(x)\mathrm{d}x-\boldsymbol{K}(2)\right)^2+\left(\int_0^{\boldsymbol{w}(3)} x^2\cos(x)\mathrm{d}x-\boldsymbol{K}(3)\right)^2$$

程序命令:

```
w = [pi/2,pi,pi * 1.5];
K = [pi/2 - 1, - 2, - 1.5 * pi - 1];
F1 = @(x)(x.^2. * cos(x)  - K(1)).^2;
F2 = @(x)(x.^2. * cos(x)  - K(2)).^2;
F3 = @(x)(x.^2. * cos(x)  - K(3)).^2;
y = quadl(F1,0,w(1)) +  quadl(F2,0,w(2)) +  quadl(F3,0,w(3))
```

结果:

```
y = 1.378679143103574e + 02
```

3.7 求零点与极值

函数的零点和方程的根是等同的,函数极值一般指极大值或极小值,MATLAB 提供了相应函数进行计算。

3.7.1 求零点

语法格式：

```
x = fzero(fun,x0)                % 求出函数 fun 在 x0 最近的零点
x = fzero(fun,x0,options)        % 由指定的优化参数 options 进行最小化
x = fzero(problem)               % 对 problem 指定的求根问题求零点
```

说明：fzero()函数既可以求某个初始值的零点，也可求区间和函数值的零点。

【例 3-14】 求函数 $f(x)=x^5-3x^4+2x^3+x+3$ 的根。

程序命令：

```
f = 'x^5 - 3 * x^4 + 2 * x^3 + x + 3';
x = fzero(f,0)
```

结果：

```
x =  -0.7693
```

因为 $f(x)$是一个多项式，所以可以使用 roots 命令求出相同的实数零点和复共轭零点，即

```
p = [1  -3 2 0 1 3]; x = roots(p)
x   =
      1.8846  +0.58974i
      1.8846 - 0.58974i
      2.4286e - 16 + 1i
      2.4286e - 16  - 1i
    - 0.76929  + 0i
```

【例 3-15】 求正弦函数在 3 附近的零点，并求余弦函数在区间[1,2]的零点。

程序命令：

```
fun = @sin;
fun1 = @cos;
x = fzero(fun,3)
x1 = fzero(fun1,[1 2])
```

结果：

```
x  = 3.1416
x1  = 1.5708
```

3.7.2 求极值

fminbnd(f,a,b)函数是对 $f(x)$在[a,b]上求得极小值,求$-f(x)$的极小值时即可得到 $f(x)$的极大值。当不知道极值所在的范围时,可画出该函数的图形,估计极值范围再使用该命令求取。

语法格式:

```
[x,min] = fminbnd(f,a,b)          % x 为取得极小值的点,min 为极小值; f 表示函数名,a,b 表示
                                  % 求取极值的范围
```

【例 3-16】 求表达式 $f=2e^{-x}\sin(x)$在[0,8]的极值点。

程序命令:

```
syms x;
f = '2 * exp( - x) * sin(x)';
[x,min1] = fminbnd(f,0,8)
[x,max1] = fminbnd(' - 2 * exp( - x) * sin(x)',0,8)
```

结果:

```
x  = 3.9270
min1  = - 0.0279                  % 极小值点
x  = 0.7854
max1  = - 0.6448
max = - max1 =  0.6448            % 极大值点
```

3.8 求方程的解

方程的求解一般包括对线性方程、符号代数方程和常微分方程的求解。对高次方程求解是比较复杂的计算过程,使用 MATLAB 函数可替代复杂的编程过程,只用一个函数即可完成方程求解。

3.8.1 线性方程组求解

(1) 直接使用左除法求解。

【例 3-17】 利用左除法求三元一次方程组的解。

$$\begin{cases} x+y+z=1 \\ 3x-y+6z=7 \\ y+3z=4 \end{cases}$$

程序命令：

```
A = sym('[1  1  1; 3  -1  6; 0  1  3]');
b = sym('[1; 7; 4]');
x = A\b
```

结果：

```
x =  -1/3
      0
     4/3
```

（2）使用 solve 函数求解。

【例 3-18】 使用 solve 函数求解三元一次方程组。

$$\begin{cases} 2x+3y-z=2 \\ 8x+2y+3z=4 \\ 45x+3y+9z=23 \end{cases}$$

程序命令：

```
syms x,y,z;                          % 建立符号变量
[x,y,z] = solve(2 * x + 3 * y - z - 2, 8 * x + 2 * y + 3 * z - 4, 45 * x + 3 * y + 9 * z - 23)
                                     % 使用 solve 函数求解
```

结果：

```
x = 151/273
y = 8/39
z = -76/273
```

3.8.2 符号代数方程求解

线性方程组的符号解也可用 solve 函数求解，当方程组不存在符号解又无其他自由参数时，则给出数值解。

语法格式：

```
solve(f,'v')                         % 求一个方程的解
solve(f1,f2,…,fn)                    % 求 n 个方程的解
```

说明：f 既可以是含等号、符号表达式的方程，也可以不含等号或符号表达式，但所求的解均是令 f=0 的方程。当参数 v 省略时，默认为方程中的自由变量；其输出结果为结构数组类型。

【例 3-19】 已知方程为 $ax^2+bx+c=0$，$x^2-x-30=0$，求符号方程解及数值解。

程序命令：

```
syms a,b,c,x;
f1 = a * x^2 + b * x + c;
f2 = x^2 - x - 30;
Fx = solve(f1,x)                    % 对默认变量 x 求解
Fb = solve(f1,b )                   % 对指定变量 b 求解
F2 = solve(f2,x)
```

结果：

```
Fx = -(b + (b^2 - 4 * a * c)^(1/2))/(2 * a)
     -(b - (b^2 - 4 * a * c)^(1/2))/(2 * a)
Fb = -(a * x^2 + c)/x
F2 = -5
      6
```

3.8.3 常微分方程(组)求解

在 MATLAB 中，用 diff(y,x)表示 y′，diff(y,x,2)表示 y″，初始条件 Dy(0)=5 表示 y′(0)=5。解方程使用 dsolve( )函数。

语法格式：

```
dsolve( )                           % 求解符号常微分方程的解
dsolve(e,c,v)                       % 求解常微分方程 e 在初值条件 c 下的特解
```

说明：参数 v 描述方程中的自变量，省略时默认自变量是 t。若没有给出初值条件 c，则求的是方程的通解。

dsolve 在求常微分方程组时的调用格式为

$$dsolve(e1,e2,\cdots,en,c1,\cdots,cn,v1,\cdots,vn)$$

该函数求解常微分方程组 e1,…,en 在初值条件 c1,…,cn 下的特解。若不给出初值条件，则求方程组的通解，v1,…,vn 给出求解变量，若省略自变量，则默认自变量为 t。若找不到解析解，则返回其积分形式。

【例 3-20】 求微分方程 $\frac{dy}{dx}+2xy=xe^{-x^2}$ 的通解。

程序命令：

```
syms y(x);
equ1 = diff(y,x) == -2 * x * y + x * exp(-x^2) ;
Y = dsolve(equ1)
```

结果：

```
y = C1 * exp(-x^2) + (x^2 * exp(-x^2))/2
```

【例 3-21】 建立脚本程序，求微分方程 $xy''=\sqrt{(1+y'^2)}/2$ （$y_{100}=0, y'_{100}=0$）的解。

程序命令：

```
equ = diff(y,x,2) == sqrt(1 + diff(y,x)^2)/(2 * x);
cond = [y(100) == 0, Dy(100) == 0];
Y = dsolve(equ,cond)
```

结果：

```
Y =  (10 * x^(1/2) * (x/100 - 3))/3 + 200/3
   - (10 * x^(1/2) * (x/100 - 3))/3 - 200/3
```

【例 3-22】 求下列微分方程组的通解。

$$\begin{cases}\dfrac{d^2x}{dt}+2\dfrac{dx}{dt}=x(t)+2y(t)-e^{-t}\\[2ex]\dfrac{dy}{dt}=4x(t)+3y(t)+4e^{-t}\end{cases}$$

程序命令：

```
syms x(t)  y(t);
eqns = [diff(x,t,2) == x(t) + 2 * y(t) - exp( - t) - 2 * diff(x,t), diff(y,t) == 4 * x(t) + 3 * y
(t) + 4 * exp( - t)];
[x,y] = dsolve(eqns)
```

结果：

```
x =
exp(t * (6^(1/2) + 1)) * (6^(1/2)/5 - 1/5) * (C2 + exp( - 2 * t - 6^(1/2) * t) * ((11 * 6^(1/
2))/3 - 37/4)) - exp( - t) * (C1 + 6 * t) - exp( - t * (6^(1/2) - 1)) * (6^(1/2)/5 + 1/5) *
(C3 - exp(6^(1/2) * t - 2 * t) * ((11 * 6^(1/2))/3 + 37/4))
y =
exp( - t) * (C1 + 6 * t) + exp(t * (6^(1/2) + 1)) * ((2 * 6^(1/2))/5 + 8/5) * (C2 + exp( - 2
 * t - 6^(1/2) * t) * ((11 * 6^(1/2))/3 - 37/4)) - exp( - t * (6^(1/2) - 1)) * ((2 * 6^(1/2))/
5 - 8/5) * (C3 - exp(6^(1/2) * t - 2 * t) * ((11 * 6^(1/2))/3 + 37/4))
```

3.9 级数

级数指将数列的项依次用加号连接起来的函数，级数操作一般包括级数求和与级数展开，它是数学基础分析的一个分支。

3.9.1 级数求和

级数求和运算是数学中常见的一种运算。例如：

$$s=a_0+a_1x+a_2x^2+a_3x^3+\cdots+a_nx^n$$

函数 symsum 用于通式项表达式 s 的求和运算。当确定变量个数为 n，变量变化范围为 a 到 b 时，可进行级数求和。

语法格式：

```
(1) symsum(n)                 %求自变量 n 从 0 到 n-1 的前 n 项的和
(2) symsum(s,n)               %s 为级数通式项,求自变量 n 从 0 到 n-1 的前 n 项的和
(3) symsum(s,a,b)             %s 为级数通式项,默认变量为 n,求从 a 到 b 的和
(4) symsum(s,n,a,b)           %s 为级数通式项,求自变量 n 从 a 到 b 的和
```

当默认变量不变时，前两种用法基本相同，后两种用法基本相同。

【例 3-23】 求级数 $S=\sum_{k=1}^{\infty}\frac{1}{(n+1)^2}$ 的前 n 项和、从 1 到无穷及其前 10 项和。

程序命令：

```
clc; syms n;
s = 1/(n+1)^2
S1 = symsum(s)
S2 = symsum(s,n)
S10 = symsum(s,1,10)
S20 = symsum(s,n,1,10)
```

结果：

```
s = 1/(n + 1)^2
S1 = piecewise( -1 < n, -psi(1,n + 1),n <=  -1,psi(1, -n))
S2 = piecewise( -1 < n, -psi(1,n + 1),n <=  -1,psi(1, -n))
S10 = 85758209/153679680
S20 = 85758209/153679680
```

说明：

(1) piecewise 表示分段函数，n＞－1 时，结果为－psi(1,n＋1)；n≤－1 时，结果为 psi(1,－n)。

(2) psi(X)为数组 X 的每个元素计算 psi 函数。X 必须是非负实数。psi 函数也称为双 γ 函数，是 γ 函数的对数导数。若 psi(k,X)为在 X 的元素中计算 psi 函数的第 k 个导数。psi(0,X)是双 γ 函数，psi(1,X) 是三 γ 函数，psi(2,X)是四 γ 函数，以此类推。

3.9.2 一元函数的泰勒级数展开

语法格式：

```
taylor(f)                     %求 f 关于默认变量的 5 阶泰勒级数展开
taylor(f,n)                   %求 f 关于默认变量的 n-1 阶泰勒级数展开
taylor(f,n,v)                 %求 f 关于变量 v 的 n-1 阶泰勒级数展开
taylor(f,n,v,a)               %求 f 在 v=a 处的 n-1 阶泰勒级数展开
```

【例 3-24】 已知函数表达式 $e^x=\sum_{n=0}^{\infty}\frac{x^n}{n!}$，$\sin x=\sum_{n=0}^{\infty}\frac{(-1)^n}{(2n+1)!}x^{2n+1}$，$\cos x=\sum_{n=0}^{\infty}\frac{(-1)^n}{2n!}x^{2n}$，求三个函数表达式的泰勒级数展开式。

程序命令：

```
syms x;
f1 = exp(x); f2 = sin(x); f3 = cos(x);
taylorexpx = taylor(f1)
taylorsinx = taylor(f2)
taylorcosx = taylor(f3)
```

结果：

```
taylorexpx = x^5/120 + x^4/24 + x^3/6 + x^2/2 + x + 1
taylorsinx = x^5/120 - x^3/6 + x
taylorcosx = x^4/24 - x^2/2 + 1
```

3.10 常用绘图功能

MATLAB 具有丰富的绘图功能，提供了一系列的绘图函数，不仅包括常用的二维图，还包括三维函数、三维网格、三维网面及三维立体切片图等，使用系统绘图函数不需要过多编程，只需给出一些基本参数就能得到所需图形。此外，MATLAB 还提供了直接对图形句柄进行操作的低层绘图操作，包括图形元素，例如坐标轴、曲线、文字等，系统对每个对象分配了句柄，通过句柄即可对该图形元素进行操作，而不影响其他部分。

3.10.1 二维绘图

MATLAB 的 plot()函数是绘制二维图形最基本的函数，它是针对向量或矩阵列来绘制曲线的，绘制以 x 轴和 y 轴为线性尺度的直角坐标曲线。

1. 语法格式

```
plot(x1,y1,option1,x2,y2,option2,…)
```

说明：x1，y1，x2，y2 给出的数据分别为 x 轴和 y 轴坐标值，option 定义了图形曲线的颜色、字符和线型，它由一对单引号括起来。可以画一条或多条曲线。若 x1 和 y1 都是数组，按列取坐标数据绘图。

2. option 的含义

option 通常由颜色(见表 3.1)、字符(见表 3.2)和线型(见表 3.3)组成。

表 3.1　颜色表示

选项	含　义	选项	含　义	选项	含　义
'r'	红色	'w'	白色	'k'	黑色
'g'	绿色	'y'	黄色	'm'	锰紫色
'b'	蓝色	'c'	亮青色	—	—

表 3.2　字符表示

选项	含　义	选项	含　义	选项	含　义
'.'	画点号	'o'	画圈符	'd'	画菱形符
'*'	画星号	'+'	画十字符	'p'	画五角形符
'x'	画叉号	's'	画方块符	'h'	画六角形符
'^'	画上三角	'>'	画左三角	—	—
'v'	画下三角	'<'	画右三角	—	—

表 3.3　线型表示

选项	含　义	选项	含　义
'-'	画实线	'.-'	点画线
'--'	画虚线	':'	画点线

【例 3-25】 绘制表达式 $y=2e^{-0.5t}\sin(2\pi t)$的曲线。

程序命令：

```
t = 0: pi/100: 2 * pi; y1 = 2 * exp( - 0.5 * t). * sin(2 * pi * t);
y2 = sin(t); plot(t,y1,'b - ',t,y2,'r - o')
```

结果如图 3.2 所示。

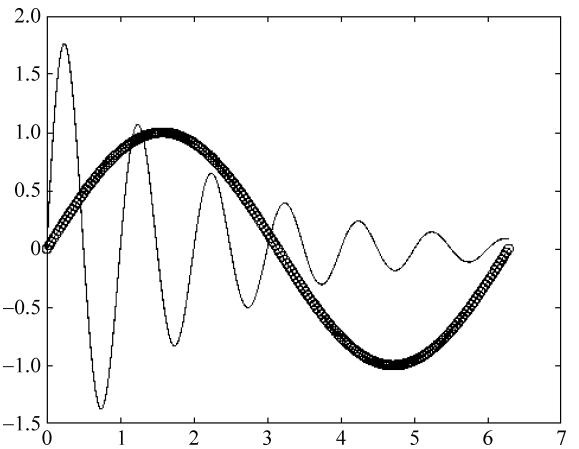

图 3.2　例 3-25 函数曲线

【例 3-26】 绘制表达式 $x=t\sin 3t, y=t\sin t\sin t$ 的曲线。

程序命令：

```
t = 0: 0.1: 2 * pi;    x = t. * sin(3 * t);    y = t. * sin(t). * sin(t);
plot(x,y,'r - p');
```

结果如图 3.3 所示。

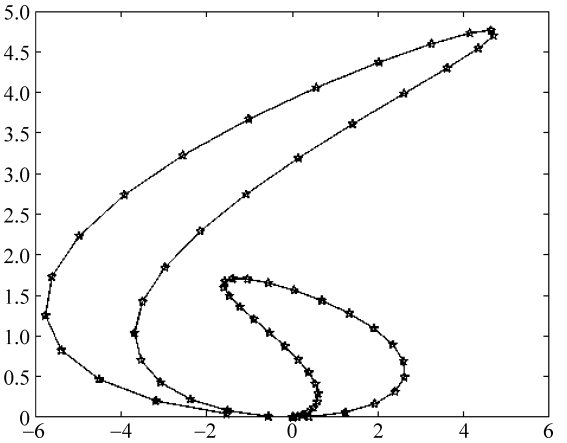

图 3.3 例 3-26 函数曲线

3. 图形屏幕控制命令

```
figure                    % 打开图形窗口
clf                       % 清除当前图形窗口的内容
hold on                   % 保持当前图形窗口的内容
hold off                  % 解除保持当前图形状态
grid on                   % 给图形加上栅格线
grid off                  % 删除栅格线
box on                    % 在当前坐标系中显示一个边框
box off                   % 去掉边框
close                     % 关闭当前图形窗口
close all                 % 关闭所有图形窗口
```

【例 3-27】 在不同窗口绘制 $y_1=\cos(t), y_2=\sin^2(t)$ 的波形图。

程序命令：

```
t = 0: pi/100: 2 * pi;    y1 = cos(t);    y2 = sin(t).^2;
figure(1); plot(t,y1,'g - p'); grid on; figure(2); plot(t,y2,'r - 0'); grid on;
```

结果如图 3.4 所示。

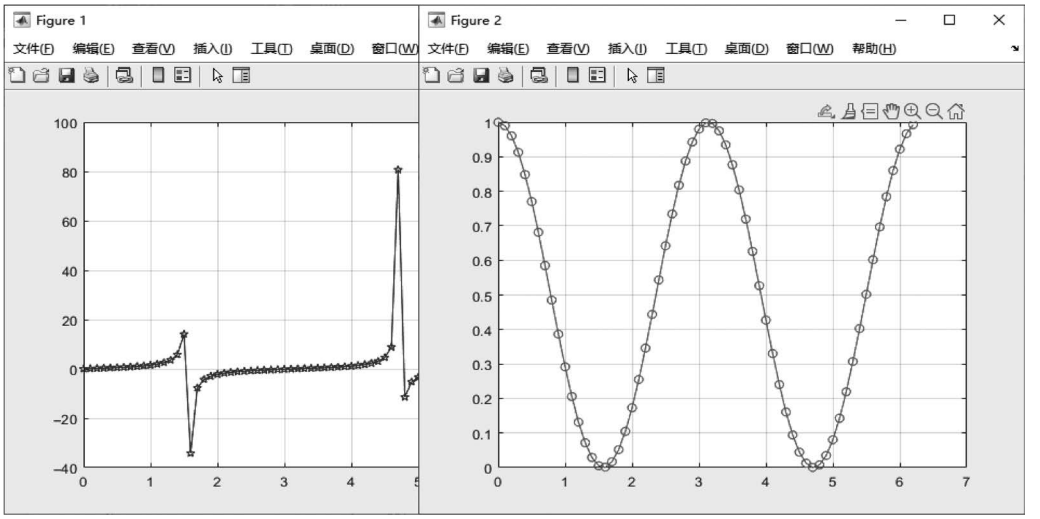

图 3.4　不同窗口绘图

4. 图形标注

```
title                          % 图题标注
xlabel                         % x 轴说明
ylabel                         % y 轴说明
zlabel                         % z 轴说明
legend              % 图例标注,legend 函数用于绘制曲线所用线型、颜色或数据点标记图例
legend('字符串 1','字符串 2',…)      % 按指定字符串顺序标记当前轴的图例
legend(句柄,'字符串 1','字符串 2',…)      % 指定字符串标记句柄图形对象图例
legend(M)           % 用字符 M 矩阵的每一行字符串作为图形对象标签标记图例
legend(句柄,M)      % 用字符 M 矩阵的每一行字符串作为指定句柄的图形对象标签标记图例
text                % 在图形中指定的位置(x,y)处显示字符串 string,格式: text (x,y,'string')
annotation % 线条、箭头和图框标注,例如,annotation('arrow',[0.1,0.45],[0.3,0.5])绘制箭头线
```

5. 字体属性

字体属性如表 3.4 所示。

表 3.4　字体属性

属　性　名	注　　释	属　性　名	注　　释
FontName	字体名称	FontWeight	字形
FontSize	字体大小	FontUnits	字体大小单位
FontAngle	字体角度	Rotation	文本旋转角度
BackgroundColor	背景色	HorizontalAlignment	文字水平方向对齐
EdgeColor	边框颜色	VerticalAlignment	文字垂直方向对齐

说明：

(1) FontName 属性定义名称，其取值是系统支持的一种字体名。

(2) FontSize 属性设置文本对象的大小，其单位由 FontUnits 属性决定，默认值为 10 磅。

(3) FontWeight 属性设置字体粗细，取值可以是 normal(默认值)、bold、light 或 demi。

(4) FontAngle 属性设置斜体文字模式，取值可以是 normal(默认值)、italic 或 oblique。

(5) Rotation 属性设置字体旋转角度，取值是数值量，默认值为 0，取正值时表示逆时针方向旋转，取负值时表示顺时针方向旋转。

(6) BackgroundColor 和 EdgeColor 属性设置文本对象的背景颜色和边框线的颜色，可取值为 none(默认值)或颜色字母。

(7) HorizontalAlignment 属性设置文本与指定点的相对位置，可取值为 left(默认值)、center 或 right。

6. 坐标轴 axis 的用法

语法格式：

axis([x_{min}, x_{max}, y_{min}, y_{max}]) 或 axis([x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max}])

说明：该函数用来标注输出图线的坐标范围。若给出 4 个参数则标注二维曲线最大值和最小值，给出 6 个参数则标注三维曲线最大值和最小值。其中：

```
axis equal              %将两坐标轴设为相等
axis on/off             %显示/关闭坐标轴的显示
axis auto               %将坐标轴设置为默认值
axis square             %产生两轴相等的正方形坐标系
```

7. 子图分割

语法格式：

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

其中，n 表示行数，m 表示列数，p 表示绘图序号，顺序是按从左至右、从上至下排列，它把图形窗口分为 n * m 个子图，在第 p 个子图处绘制图形。

【例 3-28】 利用子图绘制正弦和余弦图形。

程序命令：

```
t = 0: pi/100: 2 * pi; y1 = sin(t); y2 = cos(t); y3 = sin(t).^2; y4 = cos(t).^2;
subplot(2,2,1),plot(t,y1); title('sin(t)'); subplot(2,2,2),plot(t,y2,'g-p'); title('cos(t)')
subplot(2,2,3),plot(t,y3,'r-O'); title('sin^2(t)'); subplot(2,2,4),plot(t,y4,'k-h'); title
('cos^2(t) ')
```

结果如图 3.5 所示。

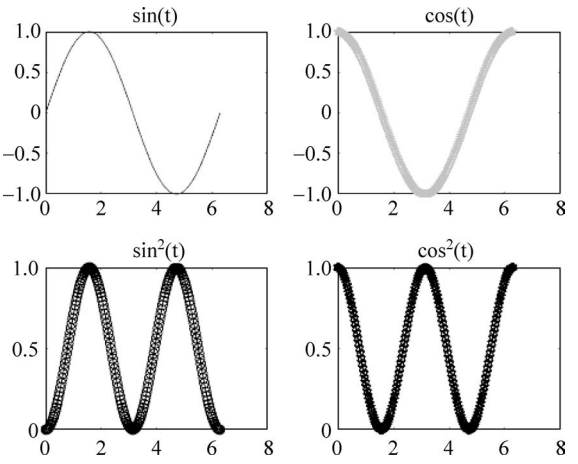

图 3.5　绘制子图

8. 绘制二维散点图(气泡图)

语法格式：

```
scatter(x,y)          % 在向量 x 和 y 指定的位置创建一个包含圆形的散点图
scatter(x,y,sz)       % sz 为标量是大小相等的圆圈,若 sz 为(x,y)长度不同向量是大小不等的圆
scatter(x,y,sz,c)     % c 指定为颜色名称或 RGB 三基色
scatter(,'filled')    % 'filled' 为填充圆形,与前面语法中的任何输入参数组合使用
scatter(,mkr)         % mkr 为指定标记,若'LineWidth',2 将轮廓宽度设置为 2 磅
scatter(ax,)          % ax 为指定的坐标区名称,可位于前面的语法中的任何输入参数组合之前
```

【例 3-29】 绘制 0～3π 的蓝色大小为 60 的散点图。

```
x = 0:pi/10:3 * pi
y = sin(x) + rand(1,31);
scatter(x,y,55,"blue")
```

结果如图 3.6 所示。

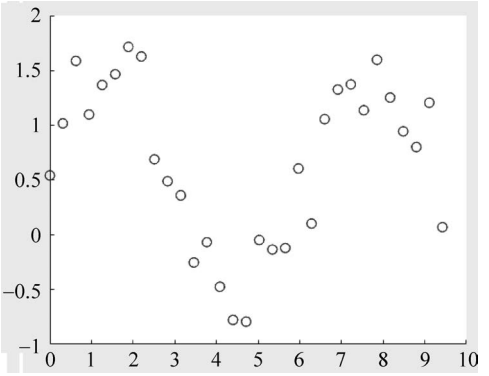

图 3.6　绘制二维散点图

3.10.2 三维绘图

1. 绘制三维空间曲线

与 plot()函数相类似,可以使用 plot3()函数来绘制一条三维空间的曲线。

语法格式:

```
plot3(x,y,z,option)  %绘制三维曲线
```

其中,x,y,z 以及选项 option 与 plot()函数中的 x,y 和选项相类似,多了一个 z 坐标轴。绘图方法可参考 plot()函数的使用方法。option 指定颜色、线形等。

【例 3-30】 已知函数,绘制[0,2π]区间三维函数曲线。

$$\begin{cases} x=(8+3\cos V)\cos U \\ y=(8+3\cos V)\sin U, \quad 0<U,V\leqslant 2 \\ z=3\sin V \end{cases}$$

程序命令:

```
r = linspace(0,2 * pi,60); [u,v] = meshgrid(r);
x = (8 + 3 * cos(v)). * cos(u); y = (8 + 3 * cos(v)). * sin(u); z = 3 * sin(v);
plot3(x,y,z); title('三维空间绘图'); xlabel('x 轴'); ylabel('y 轴'); zlabel('z 轴')
```

结果如图 3.7 所示。

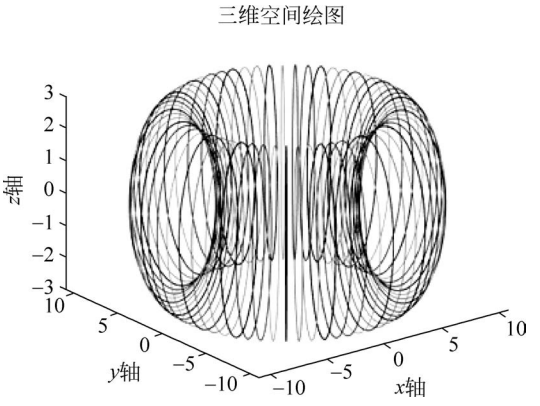

图 3.7 三维空间曲面图

2. 绘制三维散点图

语法格式:

```
scatter3(x,y,z)      %在向量 x、y 和 z 指定的位置显示圆圈
scatter3(x,y,z,sz)   %sz 同二维散点图一致
```

```
scatter3(x,y,z,sz,c) % c 指定的颜色同二维散点图一致
scatter3(,'filled')  % 使用前面的语法中的任何输入参数组合填充这些圆
scatter3( ,mkr)      % 指定标记类型,同二维散点图一致
scatter3(ax, )       % ax 指定的坐标区中
```

【例 3-31】 绘制指定大小和颜色的一个球形散点图。

```
[x,y,z] = sphere(10);
scatter3(x,y,z,65,"magenta")
```

结果如图 3.8 所示。

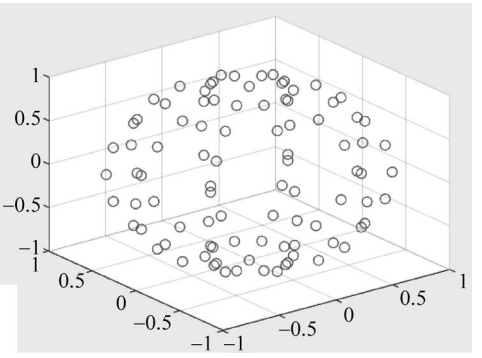

图 3.8　三维球形散点图

3. 网格矩阵的设置

meshgrid 函数产生二维阵列和三维阵列。用户需要知道各个四边形顶点的三维坐标值(x,y,z)。语法格式：

```
[X,Y] = meshgrid(x,y)     % 向量 x,y 分别指定 x 轴向和 y 轴向的数据点. 当 x 为 n 维向量,y 为
                          % m 维向量时,X,Y 均为 m * n 的矩阵. [X,Y] = meshgrid(x)等效于
                          % [X,Y] = meshgrid(x,x)
```

语法格式：

```
[X,Y,Z] = meshgrid(x,y,z)     % 产生 x 轴、y 轴和 z 轴的三维阵列
```

4. 绘制三维网格曲面图

语法格式：

```
mesh(x,y,z,c)
```

说明：

(1) 三维网格图是一些四边形相互连接在一起构成的一种曲面图。

(2) x,y,z 是维数相同的矩阵,x 和 y 是网格坐标矩阵,z 是网格点上的高度矩阵,c 用

于指定在不同高度下的颜色范围。

(3) c 省略时,c=z,即颜色的设定正比于图形的高度。

(4) 当 x 和 y 是向量时,要求 x 的长度必须等于矩阵 z 的列长度,y 的长度必须等于矩阵 z 的行长度,x 和 y 元素的组合构成网格点的 x 轴和 y 轴坐标,z 轴坐标则取自 z 矩阵,然后绘制三维曲线。

【例 3-32】 根据函数 $z=f(x,y)$的 x 轴和 y 轴坐标找出 z 的高度,绘制 $Z=x^2+y^2$ 的三维网格图形。

程序命令:

```
x = -5: 5; y = x; [X,Y] = meshgrid(x,y)
Z = X.^2 + Y.^2 ; mesh(X,Y,Z)
```

结果如图 3.9 所示。

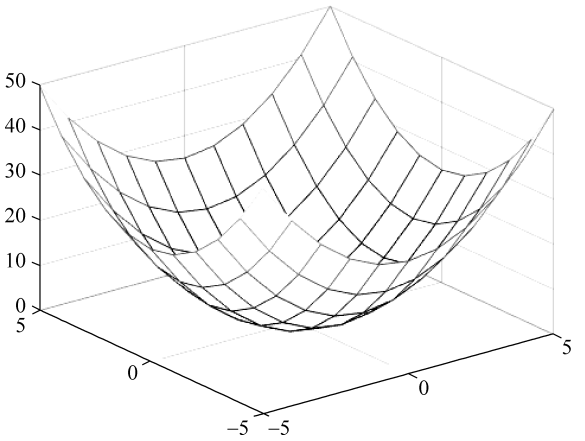

图 3.9 三维网格曲面图

5. 绘制三维曲面图

语法格式:

```
surf(x,y,z,c)
```

说明:参数 x,y,z,c 同 mesh 函数,它们均使用网格矩阵 meshgrid 函数产生坐标,自动着色,其三维阴影曲面四边形的表面颜色分布通过 shading 命令指定。

【例 3-33】 绘制马鞍函数 $z=f(x,y)=x^2-y^2$ 在[−10,10]区间的曲面图。

程序命令:

```
x = -10: 0.1: 10 ; [xx,yy] = meshgrid(x); zz  = xx .^2 - yy .^2;
surf(xx,yy,zz );   title('马鞍面'); xlabel('x 轴'); ylabel('y 轴') zlabel('z 轴'); grid on;
```

结果如图 3.10 所示。

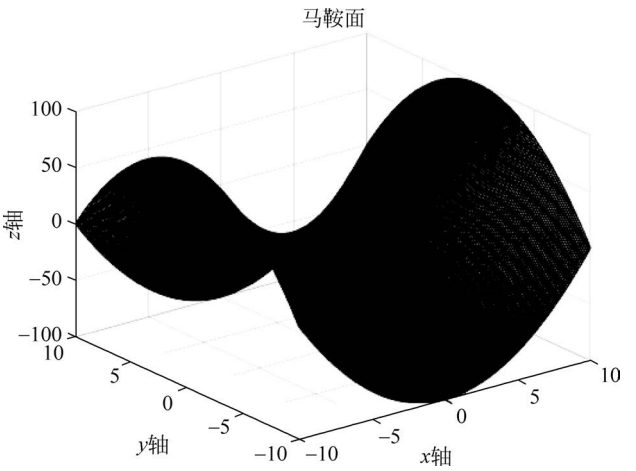

图 3.10　三维曲面图

6. 绘制特殊的三维立体图

1）球面图

MATLAB 提供了球面和柱面等标准的三维曲面绘制函数，使用户可以很方便地得到标准三维曲面图。

语法格式：

```
sphere(n)              % 画 n 等分球面,默认半径为 1,n = 20,n 表示球面绘制的精度
```

或

```
[x,y,z] = sphere(n)    % 获取球面在三维空间的坐标
```

2）柱面图

语法格式：

```
cylinder(R,n)          % R 为半径,n 为柱面圆周等分数
```

或

```
[x,y,z] = cylinder (R,n)   % x,y,z 代表空间坐标; 若在调用该函数时不带输出参数,则直接绘制
                           % 所需柱面; n 决定了柱面的圆滑程度,其默认值为 20; 若 n 值取得比
                           % 较小,则绘制出多面体的表面图
```

3）利用多峰函数绘图

多峰函数为

$$f(x,y)=3(1-x)^2\mathrm{e}^{-x^2-(y+1)^2}-10\left(\frac{x}{5}-x^3-y^5\right)\mathrm{e}^{-x^2-y^2}-\frac{1}{3}\mathrm{e}^{-(x+1)^2-y^2}$$

语法格式：

```
peaks(n)                % 输出 n * n 大小的矩阵峰值函数图形
```

或

```
[x,y,z] = peaks(n)      % x,y,z 代表空间坐标
```

【例 3-34】 使用子图分割分别绘制球面图、柱面图、多峰函数图和函数 $z=f(x,y)=\dfrac{\sin\sqrt{x^2+y^2}}{\sqrt{x^2+y^2}}$ 在[−10,10]区间的曲面图。

程序命令：

```
x = -10: 0.5: 10 ; [xx,yy] = meshgrid(x)
zz  = sin(sqrt(xx.^2 + yy.^2))./sqrt(xx.^2 + yy.^2)
subplot(2,2,1); surf(xx,yy,zz );   title('函数图'); xlabel('x 轴'); ylabel('y 轴'); zlabel('z
轴');
subplot(2,2,2); sphere(20);   title('函数图'); xlabel('x 轴'); ylabel('y 轴'); zlabel('z 轴');
subplot(2,2,3); cylinder(20);   title('函数图'); xlabel('x 轴'); ylabel('y 轴'); zlabel('z 轴');
subplot(2,2,4); peaks;   title('函数图'); xlabel('x 轴'); ylabel('y 轴'); zlabel('z 轴');
```

结果如图 3.11 所示。

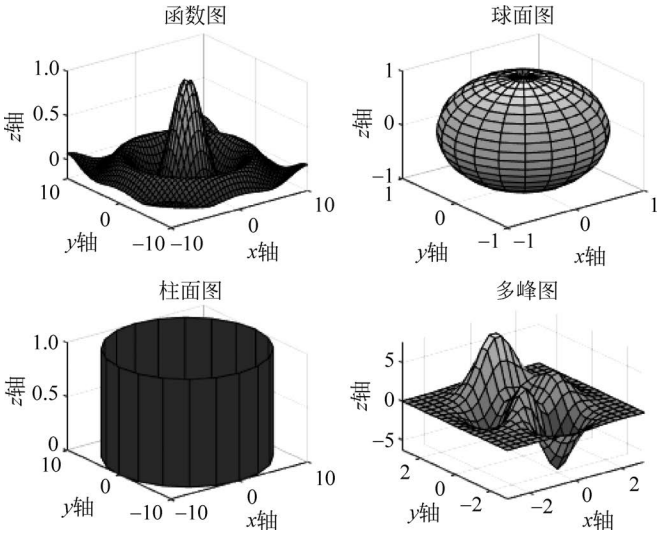

图 3.11　例 3-34 函数图像

7. 绘制三维条带图

语法格式：

```
ribbon(x)               % 使用 X = 1:size(x,1) 绘制 x 列宽度均匀的三维条带图,条带颜色可使
                        % 用 colormap
```

```
ribbon(x,y)          % x 和 y 是大小相同的向量或矩阵,Y 是包含 length(x) 行的矩阵.当 y 为矩阵
                     % 时,ribbon 将 y 中的每列绘制为位于对应 x 位置的条带
ribbon(x,y,width)    % 指定条带的宽度(默认为 0.75),若 width = 1,则各条带沿 z 轴向下紧密在
                     % 一起,若 width > 1,则条带相互重叠并可能相交
ribbon(ax,...)       % 指定的坐标区名称,可位于前面的语法中的任何输入参数组合之前
```

【例 3-35】

```
[x,y] = meshgrid( - 3:0.5:3);
z = peaks(x,y);
ribbon(y,z)
```

结果如图 3.12 所示。

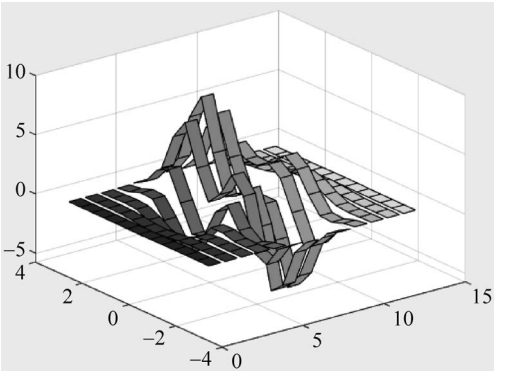

图 3.12　条带图

8. 立体切片图

语法格式：

```
slice(X,Y,Z,V,x0,y0,z0)   % 绘制向量 x1,y1,z1 中的点沿 x 轴、y 轴、z 轴方向的切片图,V 的
                          % 大小决定了每一点的颜色
slice(V,x0,y0,z0)         % 按数组 x1,y1,z1 的网格单位进行切片
```

说明：

(1) 该函数是显示三维函数 V=V(X,Y,Z)确定的超立体形在 x 轴、y 轴与 z 轴方向上的若干点切片图。在 V=V(X,Y,Z)中的变量 X 取一定值 X_1，则函数 V=V(X_1,Y,Z)变成带颜色的立体曲面切片图，各点的坐标由向量 x_0,y_0,z_0 指定。参量 X,Y,Z 为三维数组，用于指定立方体 V 的坐标。参量 X、Y 与 Z 必须有单调的正交间隔，如同用命令 meshgrid 生成的一样，在每一点上的颜色由对超立体 V 的三维内插值确定。

(2) slice(V,x0,y0,z0)省略了 X,Y,Z，默认取值为 X =1：m,Y=1：n,Z=1：p。m,n,p 为 V 的三维数组(阶数)。

【例 3-36】 绘制三维函数 $V=f(x,y,z)=xe^{-x^2-y^2-z^2}$ 的立体切片图，要求切片的坐

标为([－0.5,0.8,1.5],0.5,[])。

程序命令：

```
[x,y,z] = meshgrid( - 2: 0.2: 2, - 2: 0.25: 2, - 2: 0.2: 2)
V = x. * exp( - x.^2 - y.^2 - z.^2);
slice(x,y,z,V,[ - 0.5,0.8,1.5],0.5,[]);
```

绘图的结果如图 3.13 所示。

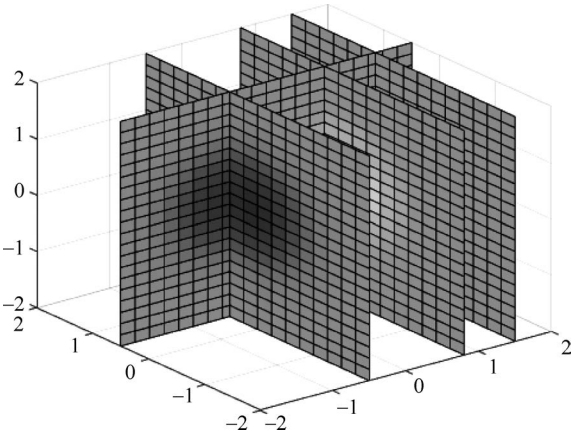

图 3.13　三维切片图

3.11　函数插值

已知函数表达式的情况下，插值就是在已知的数据点之间利用某种算法寻找估计值的过程，即根据一元线性函数表达式 $f(x)$中的两点，找出 $f(x)$在中间点的数值。插值运算可大大减少编程语句，使得程序简洁清晰。

3.11.1　一维插值

MATLAB 提供的一维插值函数为 interp1( )，定义如下：

若已知在点集 $\boldsymbol{x}$ 上的函数值 $\boldsymbol{y}$，构造一个解析函数曲线，通过曲线上的点求出它们之间的值，这一过程称为一维插值。

语法格式：

```
yi = interp1(x,y,xi);            % x,y 为已知数据值,xi 为插值数据点;
y1 = interp1(x,y,xi,'method'); % x,y 为已知数据值,xi 为插值点,method 为设定插值方法
```

说明：method 常用的设置参数有 linear，nearest 和 spline，分别表示线性插值、最邻近插值和三次样条插值。linear 也称为分段线性插值(默认值)，spline 函数插值所形成的曲线

最平滑、效果最好。

(1) nearest(最邻近插值)：该方法将内插点设置成最接近于已知数据点的值，其特点是插值速度最快，但平滑性较差。

(2) linear(线性插值)：该方法连接已有数据点作线性逼近。它是 interp1()函数的默认方法，其特点是需要占用更多的内存，速度比 nearest 方法稍慢，但是平滑性优于 nearest 方法。

(3) spline(三次样条插值)：该方法利用一系列样条函数获得内插数据点，从而确定已有数据点之间的函数。其特点是处理速度慢，但占用内存少，可以产生最光滑的插值结果。

【例 3-37】 先绘制 0～2π 的正弦曲线，按照线性插值、最邻近插值和三次样条插值三种方法，每隔 0.5 进行插值，绘制插值后曲线并进行对比。

程序命令：

```
clc; x = 0:2 * pi; y = sin(x);
xx = 0:0.5:2 * pi;
subplot(2,2,1); plot(x,y); title('原函数图');
y1 = interp1(x,y,xx,'linear');
subplot(2,2,2); plot(x,y,'o',xx,y1,'r'); title('线性插值')
y2 = interp1(x,y,xx,'nearest');
subplot(2,2,3); plot(x,y,'o',xx,y2,'r'); title('最邻近插值')
y3 = interp1(x,y,xx,'spline');
subplot(2,2,4); plot(x,y,'o',xx,y3,'r'); title('三次样条插值')
```

三种插值与原函数插值结果如图 3.14 所示。

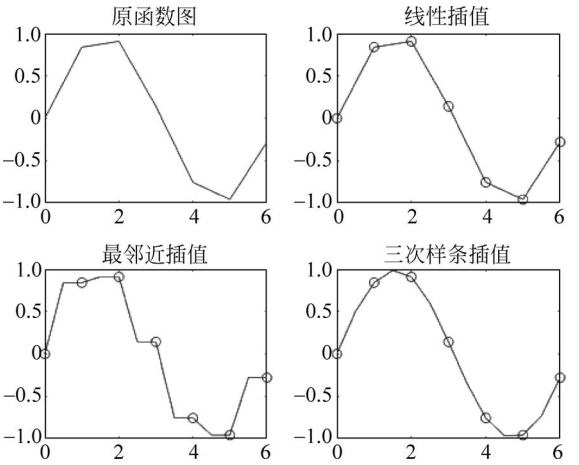

图 3.14　三种插值与原函数图的比较

结论：从线性插值、最邻近插值和三次样条插值三种方法可以看出，三次样条插值的曲线效果最平滑。

【例 3-38】 设某一天 24h 内，从零点开始每间隔 2h 测得的环境温度数据分别为 12，9，

9,10,18,24,28,25,20,16,12,11,11,请推测上午 9 时的温度。

程序命令：

```
x = 0:2:24;
y = [12,9 ,9,10,18,24,28,25,20 ,16,12,11,11];
x1 = 9;
y1 = interp1(x,y,x1,'spline');
plot(x,y,'b - p',x,y,x1,y1,'r - h');
title('插值点绘图');
text (x1 - 3,y1,'插值点');
```

结果曲线如图 3.15 所示。

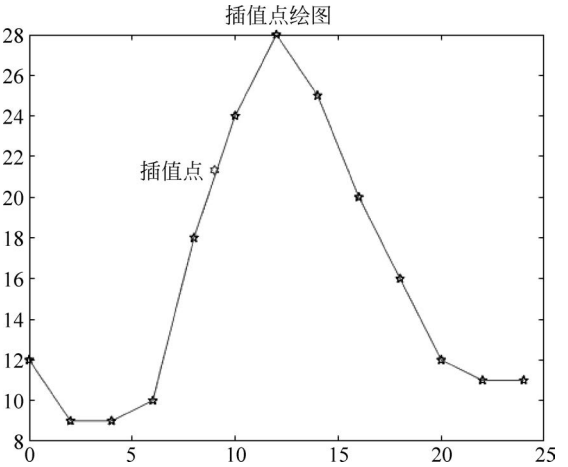

图 3.15　插值绘图

【例 3-39】 假设 2000—2020 年的产量每间隔 2 年数据分别为 90,105,123,131,150,179,203,226,249,256,267,试估计 2015 年的产量并绘图。

程序命令：

```
clear;
year = 2000:2:2020;
product = [ 90 105 123 131 150 179 203   226 249 256 267 ];
x  = 2000:1:2020;
y = interp1(year,product,x);
p2015 = interp1(year,product,2015)
plot(year,product,'b - O',x,y)
title('2000 - 2020 年的产量')
```

插值结果：

```
p2015 = 237.5000
```

默认插值曲线如图 3.16 所示。

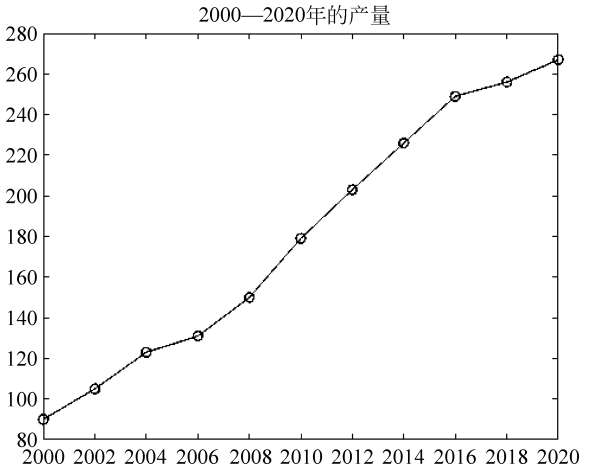

图 3.16　默认插值曲线

【例 3-40】 对离散分布在 $y=e^x \sin x$ 函数曲线上的数据，分别进行三次样条插值和线性插值计算，并绘制曲线。

程序命令：

```
clear;
x = [0 2 4 5 8 12 12.8 17.2 19.9 20];
y = exp(x). * sin(x);
xx = 0:.25:20;
yy = interp1(x,y,xx,'spline');
plot(x,y,'o',xx,yy); hold on
yy1 = interp1(x,y,xx,'linear');
plot(x,y,'o',xx,yy1); hold on
```

结果：

插值后曲线如图 3.17 所示。

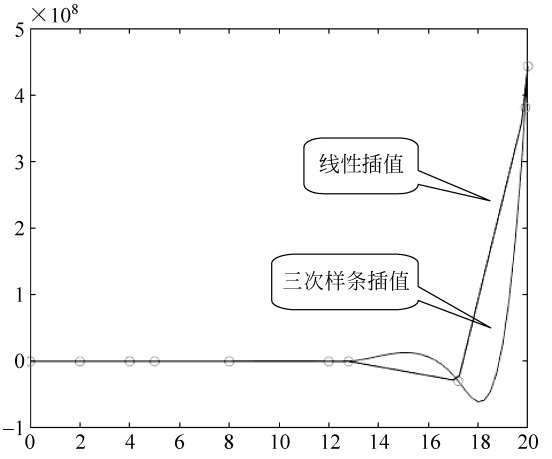

图 3.17　三次样条插值与线性插值绘图

3.11.2 二维插值

二维插值函数为二元函数。

语法格式：

```
ZZ = interp2(X,Y,Z,X1,Y1)    % X 和 Y 分别是 m 维和 n 维向量,表示节点; Z 为 n * m 的矩阵,表示
                             % 节点值; X1(行向量)和 Y1(列向量) 是插值点的一维数组,为插值
                             % 范围,若插值在范围外的点,则返回 NaN(数值为空)
ZZ = interp2(Z,X1,Y1)        % 表示 X1 = 1:n,Y1 = 1:m,其中[m,n] = size(Z)
ZZ = interp2(X,Y,Z,X1,Y1,method)   % 用指定的方法 method 计算二维插值,method 可以取值为
                                   % linear(双线性插值算法,默认值)、nearest(最邻近插值)、
                                   % spline(三次样条插值)或 cubic(双三次插值)
```

说明：interp2()函数能够较好地进行二维插值运算，但是它只能处理以网格形式给出的数据。

【例 3-41】 已知工人的平均工资从 1980 年到 2020 年逐年提升，计算在 2000 年工作了 12 年的员工的平均工资。

程序命令：

```
years = 1980:10:2020;
times =  10:10:30;
salary = [1500 1990 2000 3010 3500 4000 4100 4200 4500 5600 7000 8000 9500 10000 12000];
S =  interp2(service,years,salary,12,2000)
```

结果：

```
S =   4120
```

【例 3-42】 对函数 $z=f(x,y)=\dfrac{\sin\sqrt{x^2+y^2}}{\sqrt{x^2+y^2}}$ 进行不同插值拟合曲面，并比较拟合结果。

程序命令：

```
[x,y] = meshgrid( - 8:0.8:8);
z = sin(sqrt(x.^2 + y.^2))./sqrt(x.^2 + y.^2);
subplot(2,2,1); surf(x,y,z); title('原图');
  [x1,y1] = meshgrid( - 8:0.5:8);
z1 = interp2(x,y,z,x1,y1);
subplot(2,2,2); surf(x1,y1,z1); title('linear');
z2 = interp2(x,y,z,x1,y1,'cubic');
subplot(2,2,3); surf(x1,y1,z2); title('cubic');
z3 = interp2(x,y,z,x1,y1,'spline');
subplot(2,2,4); surf(x1,y1,z3); title('spline');
```

程序运行结果如图 3.18 所示。

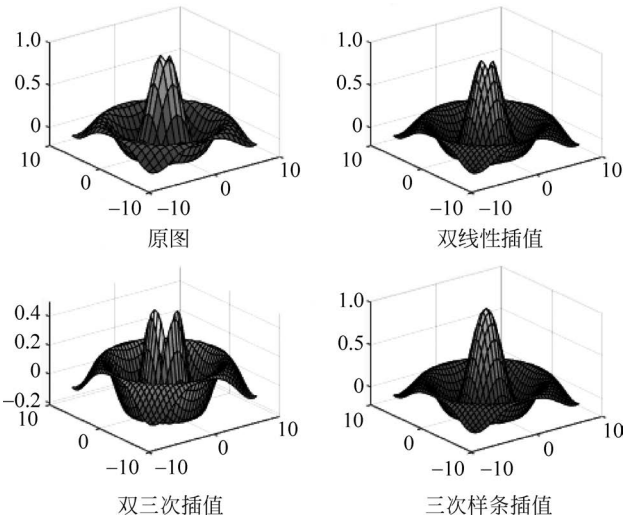

图 3.18　四种插值曲面拟合结果

3.11.3　三维插值

三维插值函数 interp3()和 n 维网格插值函数 interpn()的调用格式与函数 interp1()和 interp2()一致，需要使用三维网格生成函数实现，即[$\boldsymbol{X}$,$\boldsymbol{Y}$,$\boldsymbol{Z}$]＝meshgrid($\boldsymbol{X}$1,$\boldsymbol{Y}$1,$\boldsymbol{Z}$1)，其中 $\boldsymbol{X}$1,$\boldsymbol{Y}$1,$\boldsymbol{Z}$1 为三维所需要的分割形式，以向量形式给出三维数组，目的是返回 $\boldsymbol{X}$,$\boldsymbol{Y}$,$\boldsymbol{Z}$ 的网格数据。

语法格式：

```
interp3(X,Y,Z,V,X1,Y1,Z1,method )
```

说明：V 表示函数，使用方法与 interp2()函数一致。

【例 3-43】 已知三维函数 $V(x,y,z)=x^2+y^2+z^2$，通过函数生成网格型样本点，根据样本点进行拟合，并使用 slice 函数绘制拟合图。

程序命令：

```
clc;
[x,y,z] = meshgrid( - 1:0.1:1);
[x1,y1,z1] = meshgrid( - 1:0.1:1);
V = x.^2 + y.^2 + z.^2;                                    % 拟合函数
V1 = interp3(x,y,z,V,x1,y1,z1,'spline');                   % 三维拟合
x0 = [ - 0.5,0.5]; y0 = [0.2, - 0.2]; z0 = [ - 1, - 0.5,0.5];   % 图形位置
slice(x1,y1,z1,V1,x0,y0,z0); title('三维拟合');
```

程序运行结果如图 3.19 所示。

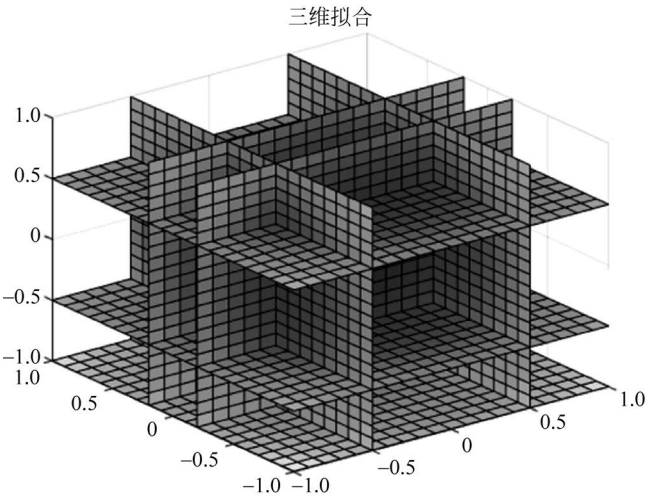

图 3.19　三维插值图