函数绘图

数据图形能使视觉感官直接感受到数据的许多内在本质，发现数据的内在联系。MATLAB 可以表达出数据的二维，三维，甚至四维的图形。

第 1 节　二维图形

一、基本平面图形绘制命令：plot

功能　线性二维图。

格式　**plot(X,'s') X** 为实向量的时候，以该向量元素的下标为横坐标，元素值为纵坐标，绘出一条连续曲线。

plot(X,Y) X,Y 为同维向量时，绘制以 **X**、**Y** 元素为横、纵坐标的曲线。

X 为向量，**Y** 为一维或多维矩阵时，绘出多条不同颜色的曲线。**X** 为这些曲线共同的横坐标。

plot(Y)：**Y** 的维数为 **m**，则 **plot(Y)**等价于 **plot(x,Y)**，其中 **x=1**：**m**

plot(X1,Y1,X2,Y2,…)，其中 **Xi** 与 **Yi** 成对出现

plot(X1,Y1,LineSpec1,X2,Y2,LineSpec2…) 将按顺序分别画出由三参数定义 **Xi,Yi,LineSpeci** 的线条。其中参数 **LineSpeci** 指明了线条的类型，标记符号，和画线用的颜色。

可混合使用三参数和二参数的形式：**plot(X1,Y1,LineSpec1,X2,Y2,X3,Y3,LineSpec3)**

plot(…,'PropertyName',PropertyValue,…) 对图形对象中指定的属性进行设置。

h = plot(…) 返回 **line** 图形对象句柄的一列向量，一线条对应一句柄值。

允许用户对线条定义的属性

1．线型

定义符	-	--	:	-.
线型	实线（缺省值）	划线	点线	点划线

2．线条宽度 **'LineWidth'**

指定线条的宽度，取值为整数（单位为像素点）

3．颜色

定义符	**r**（red）	**g(green)**	**b(blue)**	**c(cyan)**
颜色	**红色**	**绿色**	**兰色**	**青色**
定义符	**m(magenta)**	**y(yellow)**	**k(black)**	**w(white)**
颜色	**品红**	**黄色**	**黑色**	**白色**

4．标记类型

定义符	+	**o(字母)**	*	.	**x**
标记类型	**加号**	**小圆圈**	**星号**	**实点**	**交叉号**
定义符	**d**	^	**v**	>	<

标记类型	棱形	向上三角形	向下三角形	向右三角形	向左三角形
定义符	s	h	p		
标记类型	正方形	正六角星	正五角星		

5．标记大小：'MarkerSize’

指定标记符号的大小尺寸，取值为整数（单位为像素）

6．标记面填充颜色'MarkerFaceColor'

指定用于填充标记符面的颜色。取值见上表。

7．标记周边颜色 'MarkerEdgeColor'

指定标记符颜色或者是标记符（小圆圈、正方形、棱形、正五角星、正六角星和四个方向的三角形）周边线条的颜色。取值见上表。

坐标轴的设置

创建图形时，用户可以制定坐标的范围、数据间隔及坐标名称。用命令 **axis** 可以控制坐标轴的刻度及形式。

axis[Xmin ,Xmax,Ymin,Ymax]

直角坐标图形的纵横比在默认的情况下与窗口纵横比相同，用 **axis** 可以控制图形纵横比的格式如下：

axis aquare :将两个轴的长度设置为相等；

axis equal :将坐标轴的标记间距设置为相等；

axis equal tight :将图形以紧缩的方式显示。

例：

```
t=(pi*(0:1000)/1000);
y1=sin(t);y2=sin(10*t);y12=sin(t).*sin(10*t);
plot(t,y1);axis([0,pi,-1,1])
```

图形标志

图形标志包括图名、坐标轴名、图形注释和图例，常用格式如下：

title(s) :书写图名；

xlabel (s) :横坐标轴名；

ylabel (s) :纵坐标轴名；

legend(s1,s2,…) :绘制曲线所用线型、色彩或数据点形图例；

text(xt,yt,s) :在图面（**xt , yt**）坐标处书写字符注释。

多子图

Matlab 允许用户在同一图形框内布置几幅独立的子图。

subplot(m,n,k) %使 **m*n** 幅子图中的第 **k** 幅成为当前图。

图形中有 **m*n** 幅图，**k** 是子图的编号。子图的序号原则是：左上方为第一幅，向右、下依次排号。**subplot** 产生的子图相互独立，所有绘图指令均可以在子图中应用。

参数 **LineSpeci** 的说明：

参数 **LineSpeci** 可以定义线条的三个属性：线型、标记符号和颜色。对线条的上述属性的定义用字符串来定义，如：**plot(x,y,'-.or')**，表示画点划线（**-.**），在数据点（**x**，**y**）处画出小圆圈（**o**），线和标记都用红色画出。注：字符串中的字母、符号可任意组合。若仅仅指定了标记符，而没有指定非线型，则 **plot** 只在数据点画出标记符，如：**plot(x,y,’d’)**。

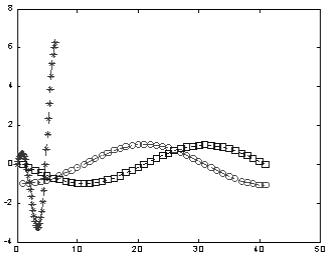
例 1

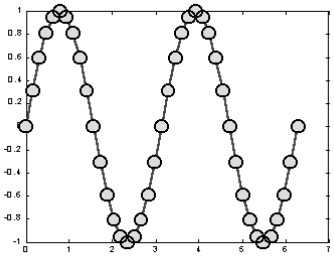
```
t = 0:pi/20:2*pi;
```

```
plot(t,t.*cos(t),'-.r*')
hold on
plot(exp(t/100).*sin(t-pi/2),'--mo')
plot(sin(t-pi),':bs')
hold off
```

例 2

```
>> t=1:0.1:10
plot(t,sin(2*t),'-mo','LineWidth',2,...
                      'MarkerEdgeColor','k',...
                      'MarkerFaceColor',[0.49,1,0.63],...
                      'MarkerSize',5)
```





二、一元函数 y=f(x)的绘图命令 fplot

fplot 采用自适应步长控制来画出函数 **function** 的示意图，在函数的变化激烈的区间，采用小的步长，否则采用大的步长。总之，使计算量与时间最小，图形尽可能精确。

注意：必须是函数，可以是一个 **m**-文件函数或者是一个包含符号变量的函数，如：**'sin(x)*exp(2*x)'**，**'[sin(x),cos(x)]'**。

格式：fplot('function',limits) 在指定的范围 **limits** 内画出一元函数图形。其中 **limits** 是一个指定 **x**-轴范围的向量**[xmin xmax]**或者是 **x** 轴和 **y** 轴的范围的向量**[xmin xmax ymin ymax]**。

fplot('function',limits,LineSpec) 用指定的线型 **LineSpec** 画出函数 **function**。

fplot('function',limits,tol) 用相对误差值为 **tol** 画出函数 **function**。相对误差的缺省值为 **2e-3**。

fplot('function',limits,tol,LineSpec) 用指定的相对误差值 **tol** 和指定的线型 **LineSpec** 画出函数 **function** 的图形。

fplot('function',limits,n) 当 **n>=1**，则至少画出 **n+1** 个点（即至少把范围 **limits** 分成 **n** 个小区间），最大步长不超过**(xmax-xmin)/n**。

fplot('function',lims,…) 允许可选参数 **tol**，**n** 和 **LineSpec** 以任意组合方式输入。

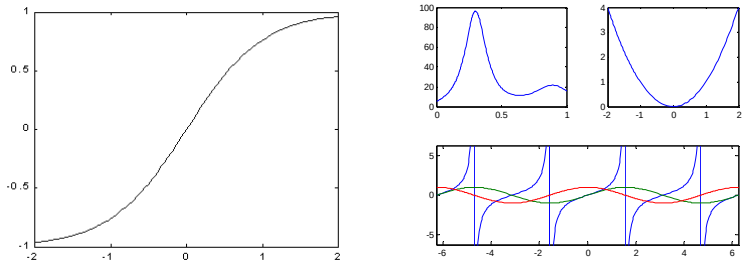
[X,Y] = fplot('function',limits,…) 返回横坐标与纵坐标的值给变量 **X** 和 **Y**，此时 **fplot** 不画出图形。若想画出，可用命令 **plot(X,Y)**。

[…] = plot('function',limits,tol,n,LineSpec,P1,P2,…) 允许用户直接给函数 **function** 输入参数 **P1**，**P2** 等，其中函数 **function** 的定义形式为

y = function(x,P1,P2,…)

若想用缺省的 **tol**，**n** 或 **LineSpec** 值，只需将空矩阵（**[ ]**）传递给函数即可。

```
>>fplot('tanh',[-2 2])
>>subplot(2,2,1);fplot('humps',[0 1])
  subplot(2,2,2);fplot('x^2',[-2 2])
  subplot(2,1,2);fplot('[tan(x),sin(x),cos(x)]',2*pi*[-1 1 -1 1])
```



绘制多条曲线：采用 **hold on** 或者 **Y** 为 二维向量

定义myfun函数

```
function Y = myfun(x)
Y(:,1) = 200*sin(x(:))./x(:);
Y(:,2) = x(:).^2;
```

```
>> fplot('myfun',[-20 20])
   title('myfun') , xlabel('x'), ylabel('y')
```

三、快速函数作图： ezplot（Easy to use function plotter）

例如

```
>> y='-16*x^2+64*x+96'    %    expression to plot
y=
  -16*x^2+64*x+96
>> ezplot(y)
```

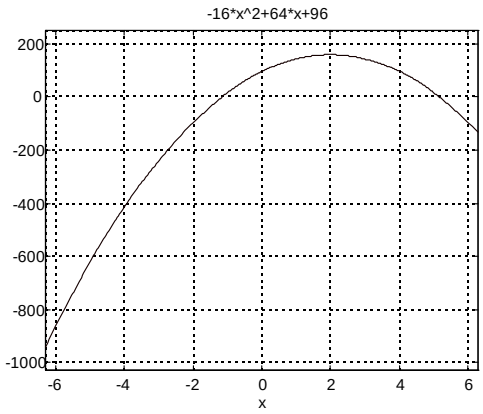


图3.1 符号函数16*x^2+64*x+96 （-2π≤x≤2π）

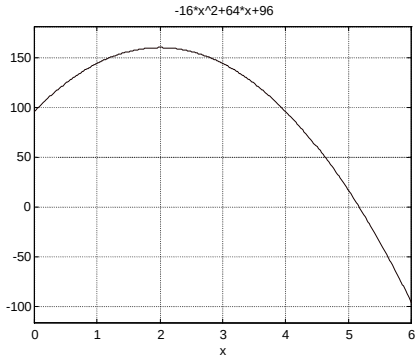


图3.2　符号函数16*xˆ2+64*x+96　　0≤x≤6

图3.1，ezplot绘制了定义域为-2π≤x≤2π的给定符号函数，如果感兴趣的时间是从0到6。需要指定范围，即

```
>> ezplot(y，[0 6])                    %   plot y for 0≤x≤6
```

第 2 节　　三维图形

一、三维图形等高线

命令 1　contour

功能　曲面的等高线图

用法　**contour(z)**　把矩阵 z 中的值作为一个二维函数的值，等高曲线是一个平面的曲线，平面的高度 v 是 Matlab 自动取的；

contour(x,y,z)　(x,y)是平面 z=0 上点的坐标矩阵，z 为相应点的高度值矩阵。效果同上；

contour(z,n)　画出 n 条等高线；

contour(x,y,z,n)　画出 n 条等高线；

contour(z,v)　在指定的高度 v 上画出等高线；

contour(x,y,z,v)　同上；

[c,h] = contour(…)　返回如同 contourc 命令描述的等高矩阵 c 和线句柄或块句柄列向量 h，这些可作为 clabel 命令的输入参量，每条线对应一个句柄，句柄中的 userdata 属性包含每条等高线的高度值；

contour(…,'linespec')　因为等高线是以当前的色图中的颜色画的，且是作为块对象处理的，即等高线是一般的线条，我们可象画普通线条一样，可以指定等高线的颜色或者线形。

命令 2　clabel

功能　在二维等高线图中添加高度标签。在下列形式中，若有 h 出现，则会对标签进行恰当的旋转，否则标签会竖直放置，且在恰当的位置显示个一个“+”号。

用法　**clabel(C,h)** 把标签旋转到恰当的角度，再插入到等高线中。只有等高线之间有足够的空间时才加入，当然这决定于等高线的尺度。

clabel(C,h,v) 在指定的高度 v 上显示标签 h，当然要对标签做恰当的处理。

clabel(C,h,'manual') 手动设置标签。用户用鼠标左键或空格键在最接近指定的位置上放置标签，用键盘上的回车键结束该操作。当然会对标签做恰当的处理。

clabel(C) 在从命令 contour 生成的等高线结构 c 的位置上添加标签。此时标签的放置的位置是随机的。

clabel(C,v) 在给定的位置 v 上显示标签

clabel(C,'manual') 允许用户通过鼠标来给等高线贴标签

例 7-27

```
>>[x,y] = meshgrid(-2:.2:2);
>>z = x.*y.*exp(-x.^2-y.^2);
>>[C,h] = contour(x,y,z);
>>clabel(C,h);
```

图形结果为图 7-27。

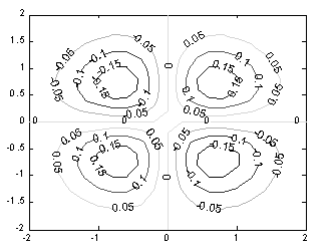


图 7-27

命令 4　contour3

功能　三维空间等高线图。该命令生成一个定义在矩形格栅上曲面的三维等高线图。

用法　**contour3(Z)** 画出三维空间角度观看矩阵 **z** 的等高线图，其中 **z** 的元素被认为是距离 **xy** 平面的高度，矩阵 **z** 至少为 **2*2** 阶的。等高线的条数与高度是自动选择的。若**[m，n]=size**（**z**），则 **x** 轴的范围为**[1：n]**，**y** 轴的范围为**[1：m]**。

contour3(Z,n) 画出由矩阵 **z** 确定的 **n** 条等高线的三维图。

contour3(Z,v) 在参量 **v** 指定的高度上画出三维等高线，当然等高线条数与向量 **v** 的维数相同；若想只画一条高度为 **h** 的等高线，输入：**contour3(Z,[h,h])**

contour3(X,Y,Z)、**contour3(X,Y,Z,n)**、**contour3(X,Y,Z,v)** 用 **X** 与 **Y** 定义 **x**-轴与 **y**-轴的范围。若 **X** 为矩阵，则 **X(1,:)**定义 **x**-轴的范围；若 **Y** 为矩阵，则 **Y(:,1)** 定义 **y**-轴的范围；若 **X** 与 **Y** 同时为矩阵，则它们必须同型。不论为哪种使用形式，所起的作用与命令 **surf** 相同。若 **X** 或 **Y** 有不规则的间距，**contour3** 还是使用规则的间距计算等高线，然后将数据转变给 **X** 或 **Y**。

contour3(…,LineSpec) 用参量 **LineSpec** 指定的线型与颜色画等高线。

[C,h] = contour3(…) 画出图形，同时返回与命令**contourc**中相同的等高线矩阵 **C**，包含所有图形对象的句柄向量**h**；除非没有指定**LineSpec**参数，**contour3** 将生成**patch**图形对象，且当前的**colormap**属性与**caxis**属性将控制颜色的显示。不论使用何种形式，该命令都生成**line**图形对象。

例 7-28

```
>>[X,Y] = meshgrid([-2:.25:2]);
>>Z = X.*exp(-X.^2-Y.^2);
>>contour3(X,Y,Z,30)
```

图形结果为图 **7-28**。

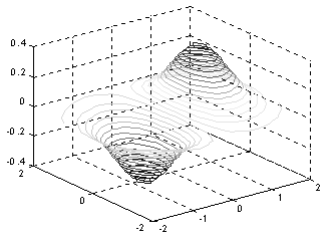


图 7-28

命令 6　pie3

功能　三维饼形图

用法　**pie3(X)** 用 **x** 中的数据画一个三维饼形图。**X** 中的每一个元素代表三维饼形图中的一部分。

pie3(X,explode) **x** 中的某一部分可以从三维饼形图中分离出来。**explode** 是一个与 **x** 同型的向量或矩阵，**explode** 中非零的元素对应 **x** 中从饼形图中分离出来的分量。

h = pie3(…) 返回一个分量为 **patch**，**surface** 和 **text** 图形句柄对象的向量。即每一块对应一个句柄。

注意：命令 **pie3** 将 **x** 的每一个元素在所有元素的总和中所占的比例表达出来。若 **x** 中的分量和小于 **1**（则所有元素小于 **1**），则认为 **x** 中的值指明三维饼形图的每一部分的大小。

例 7-30

```
>>x = [1 3 0.5 2.5 2]
>>ex = [0 1 0 0 0]
>>pie3(x,ex)
```

图形结果为图 **7-30**。

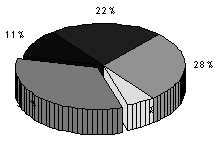


图 7-30

二、曲面与网格图命令

命令 1　mesh

功能　生成由 **X**，**Y** 和 **Z** 指定的网线面，由 **C** 指定的颜色的三维网格图。网格图是作为视点由 **view**（**3**）设定的 **surface** 图形对象。曲面的颜色与背景颜色相同（当要动画显示

不透明曲面时，这时可用命令 **hidden** 控制），或者当画一个标准的可透视的网线图时，曲面的颜色就没有（命令 **shading** 控制渲染模式）。当前的色图决定线的颜色。

用法 **mesh(X,Y,Z)** 画出颜色由 **c** 指定的三维网格图，所以和曲面的高度相匹配，

1．若 **X** 与 **Y** 均为向量，**length**（**X**）**=n**，**length**（**Y**）**=m**，而**[m**，**n]=size**（**Z**），空间中的点 **(X(j),Y(I),Z(I,j))** 为所画曲面网线的交点，分别地，**X** 对应于 **z** 的列，**Y** 对应于 **z** 的行。

2．若 **X** 与 **Y** 均为矩阵，则空间中的点 **(X(I,j),Y(I,j),Z(I,j))**为所画曲面的网线的交点。

mesh(Z) 由**[n**，**m] = size**（**Z**）得，**X =1**：**n** 与 **Y=1**：**m**，其中 **z** 为定义在矩形划分区域上的单值函数。

mesh(…,C) 用由矩阵 **c** 指定的颜色画网线网格图。**Matlab** 对矩阵 **c** 中的数据进行线性处理，以便从当前色图中获得有用的颜色。

mesh(…,PropertyName',PropertyValue, …) 对指定的属性 **PropertyName** 设置属性值 **PropertyValue,**可以在同一语句中对多个属性进行设置。

h = mesh(…) 返回 **surface** 图形对象句柄。

运算规则：

1．数据 **X**，**Y** 和 **Z** 的范围，或者是对当前轴的 **XLimMode**，**YLimMode** 和 **ZLimMode** 属性的设置决定坐标轴的范围。命令 **aXis** 可对这些属性进行设置。

2．参量 **c** 的范围，或者是对当前轴的 **Clim** 和 **ClimMode** 属性的设置（可用命令 **caxis** 进行设置），决定颜色的刻度化程度。刻度化颜色值作为引用当前色图的下标。

3．网格图显示命令生成由于把 **z** 的数据值用当前色图表现出来的颜色值。**Matlab** 会自动用最大值与最小值计算颜色的范围（可用命令 **caxis auto** 进行设置），最小值用色图中的第一个颜色表现，最大值用色图中的最后一个颜色表现。**Matlab** 会对数据的中间值执行一个线性变换，使数据能在当前的范围内显示出来。

例 7-31

```
>>[X,Y] = meshgrid(-3:.125:3);
>>Z = peaks(X,Y);
>>mesh(X,Y,Z);
```

图形结果为图 **7-31**。

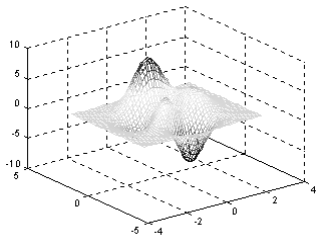


图 7-31

命令 2　surf

功能　在矩形区域内显示三维带阴影曲面图。

用法　**surf(Z)** 生成一个由矩阵 **z** 确定的三维带阴影的曲面图，其中 **[m**，**n] = size**（**Z**），而 **X = 1**：**n**，**Y = 1**：**m**。高度 **z** 为定义在一个几何矩形区域内的单值函数，**z** 同时指定曲面高度数据的颜色，所以颜色对于曲面高度是恰当的。

surf(X,Y,Z) 数据 **z** 同时为曲面高度，也是颜色数据。**X** 和 **Y** 为定义 **X** 坐标轴和 **Y** 坐标轴的曲面数据。若 **X** 与 **Y** 均为向量，**length**（**X**）**=n**，**length**（**Y**）**=m**，而 **[m,n]=size**（**Z**），在这种情况下，空间曲面上的节点为（**X(I),Y(j),Z(I,j)**）。

surf(X,Y,Z,C) 用指定的颜色 **c** 画出三维网格图。**Matlab** 会自动对矩阵 **c** 中的数据进行线性变换，以获得当前色图中可用的颜色。

surf(…,'PropertyName',PropertyValue) 对指定的属性 **PropertyName** 设置为属性值 **PropertyValue**

h = surf(…) 返回一个 **surface** 图形对象句柄给变量 **h**。

运算规则：

1．严格地讲，一个参数曲面是由两个独立的变量 **I**、**j** 来定义的，它们在一个矩形区域上连续变化。例如，**a<=I<=b,c<=j<=d**，三个变量 **X**，**Y**，**Z** 确定了曲面。曲面颜色由第四参数矩阵 **C** 确定。

2．矩形定义域上的点有如下关系：

```
                A(I-1,j)
                   |
B(I,j-1) ---- C(I,j) ---- D(I,j+1)
                   |
                E(I+1,j)
```

这个矩形坐标方格对应于曲面上的有四条边的块，在空间的点的坐标为[X(☺,Y(☺,Z)，每个矩形内部的点根据矩形的下标和相邻的四个点连接；曲面上的点只有相邻的三个点，曲面上四个角上的点只有两个相邻点，上面这些定义了一个四边形的网格图。

3．曲面颜色可以有两种方法来指定：指定每个节点的颜色或者是每一块的中心点颜色。在这种一般的设置中，曲面不一定为变量 **X** 和 **Y** 的单值函数，进一步而言，有四边的曲面块不一定为平面的，而可以用极坐标，柱面坐标和球面坐标定义曲面。

4．命令 **shading** 设置阴影模式。若模式为 **interp**，**C** 必须与 **X**，**Y**，**Z** 同型；它指定了每个节点的颜色，曲面块内的颜色由附近几个点的颜色用双线性函数计算出来的。若模式为 **facted**（缺省模式）或 **flat**，**c(I,j)**指定曲面块中的颜色：

```
A(I,j)----------- B (I,j+1)
   |      C(I,j)       |
C(I+1,j) --------- D(I+1,j)
```

在这种情形下，**C** 可以与 **X**，**Y**，和 **Z** 同型，且它的最后一行和最后一列将被忽略，换句话说，就是 **C** 的行数和列数可以比 **X**，**Y**，**Z** 少 **1**。

5．命令 **surf** 将指定图形视角为 **view**（**3**）。

6．数据 **X**，**Y**，**Z** 的范围或者通过对坐标轴的属性 **XlimMode**，**YlimMode** 和 **ZlimMode** 的当前设置（可以通过命令 **axis** 来设置），将决定坐标轴的标签。

7．参数 **C** 的范围或者通过对坐标轴的属性 **Clim** 和 **ClimMode** 的设置（可以通过命令 **caxis** 来设置），将决定颜色刻度化。刻度化的颜色值将作为引用当前色图的下标。

例 7-32

```
>>[X,Y,Z] = peaks(30);
>>surf(X,Y,Z)
>>colormap hsv
```

结果图形为图 **7-32**。

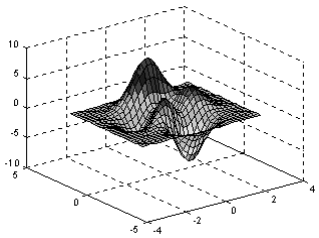


图 7-32

命令 3　surfc

功能　在矩形区域内显示三维带阴影曲面图，且在曲面下面画出等高线。

用法　**surfc(Z)**、**surfc(X,Y,Z)**、 **surfc(X,Y,Z,C)**、**surfc(…,'PropertyName',PropertyValue)**、**surfc(…)**、 **h = surfc(…)**

上面各个使用形式的曲面效果与命令 **surf** 的相同，只不过是在曲面下面增加了曲面的等高线而已。

例 7-33

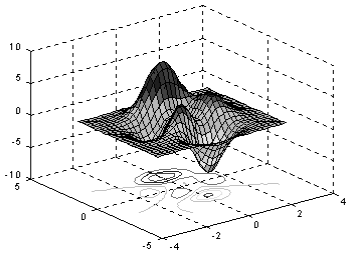


图 7-33

```
>>[X,Y,Z] = peaks(30);
>>surfc(X,Y,Z)
>>colormap hsv
```

图形结果为图 7-33。

命令 4　surfl

功能　画带光照模式的三维曲面图。该命令显示一个带阴影的曲面，结合了周围的，散射的和镜面反射的光照模式。想获得较平滑的颜色过度，要使用有线性强度变化的色图（如：gray，copper，bone，pink 等）。参数 X，Y，Z 确定的点定义了参数曲面的“里面”和“外面”，若用户想曲面的“里面”有光照模式，只要使用：

surfl(X',Y',Z')

用法　**surfl(Z)** 以向量 z 的元素生成一个三维的带阴影的曲面，其中阴影模式中的光源的方位、光照系数为缺省值（见下面）。

surfl(X,Y,Z) 以矩阵 X，Y，Z 生成的一个三维的带阴影的曲面，其中阴影模式中的光源的方位、光照系数为缺省值（见下面）。

surfl(…,'light') 用一个 matlab 光照对象（light object）生成一个带颜色、带光照的曲面，这与用缺省光照模式产生的效果不同。

surfl(…,'cdata') 改变曲面颜色数据（color data），使曲面成为可反光的曲面。

surfl(…,s) 指定光源与曲面之间的方位 s，其中 s 为一个二维向量[azimuth，elevation]，或者三维向量[sx，sy，sz]。缺省光源方位为从当前视角开始，逆时针 45℉（度）。

surfl(X,Y,Z,s,k) 指定反射常系数 k，其中 k 为一个定义环境光（ambient light）系数（0<=ka<=1）、漫反射(diffuse reflection)系数（0〈=kb〈=1〉、镜面反射(specular reflection)系数（0〈=ks〈=1〉与镜面反射亮度（以相素为单位）等的四维向量[ka，kd，ks，shine]，缺省值为 k=[0.55 0.6 0.4 10]。

h = surfl(…) 返回一个曲面图形句柄向量 h。

例 7-34

```
>>[X,Y] = meshgrid(-3:1/8:3);
>>Z = peaks(X,Y);
>>surfl(X,Y,Z);
>>shading interp
>>colormap(gray);
```

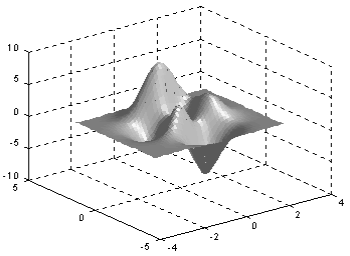


图 7-34

图形结果为图 7-34。

命令 5　waterfall

功能　瀑布图

用法　**waterfall(X,Y,Z)** 用所给参数 X、Y 与 Z 的数据画一“瀑布”效果图。若 X 与 Y 都是向量，则 X 与 Z 的列相对应，Y 与 Z 的行相对应，即 length(X)=Z 的列数，length(Y)=Z 的行数。参数 X 与 Y 定义了 x-轴与 y-轴，Z 定义了 z-轴的高度，Z 同时确定了颜色，所以颜色能恰当地反映曲面的高度。若想研究数据的列，可以输入：waterfall(Z')或 waterfall(X',Y',Z')

waterfall(Z) 画出一瀑布图，其中缺省地有：X=1:Z 的行数，Y=1:Z 的行数，且 Z 同时确定颜色，所以颜色能恰当地反映曲面高度。

waterfall(…,C) 用比例化的颜色值从当前色图中获得颜色，参量 C 决定颜色的比例，为此，必须与 Z 同型。系统使用一线性变换，从当前色图中获得颜色。

h = waterfall(…) 返回 patch 图形对象的句柄 h，可用于画出图形。

例 7-35

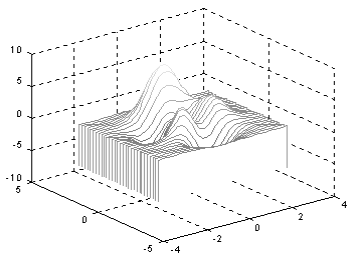


图 7-35

```
>>[X,Y,Z] = peaks(30);
>>waterfall(X,Y,Z)
```

图形结果为图 7-35。

命令 6　cylinder

功能　生成圆柱图形。该命令生成一单位圆柱体的 x-，y-，z-轴的坐标值。用户可以用命令 surf 或命令 mesh 画出圆柱形对象，或者用没有输出参量的形式而立即画出图形。

用法　**[X,Y,Z] = cylinder** 返回一半径为 1、高度为 1 的圆柱体的 x-，y-，z-轴的坐标值，圆柱体的圆周有 20 个距离相同的点。

[X,Y,Z] = cylinder(r) 返回一半径为 r、高度为 1 的圆柱体的 x-，y-，z-轴的坐标值，圆柱体的圆周有 20 个距离相同的点。

[X,Y,Z] = cylinder(r,n) 返回一半径为 r、高度为 1 的圆柱体的 x-，y-，z-轴的坐标值，圆柱体的圆周有指定的 n 个距离相同的点。

cylinder(…) 没有任何的输出参量，直接画出圆柱体。

例 7-36

```
>>t = 0:pi/10:2*pi;
>>[X,Y,Z] = cylinder(2+(cos(t)).^2);
>>surf(X,Y,Z); axis square
```

图形结果为图 7-36。

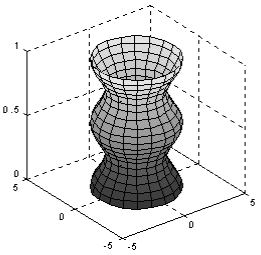


图 7-36

命令 7　sphere

功能　生成球体

用法　**sphere** 生成三维直角坐标系中的单位球体。该单位球体由 20*20 个面。

sphere(n) 在当前坐标系中画出有 n*n 个面的球体

[X,Y,Z] = sphere(n) 返回三个阶数为(n+1)*(n+1)的，直角坐标系中的坐标矩阵。该命令没有画图，只是返回矩阵。用户可以用命令 surf（X，Y，Z）或 mesh（X，Y，Z）画出球体。

例 7-37

```
>>[X,Y,Z]=sphere;
>>mesh(X,Y,Z)
>>hidden off
```

图形结果为图 7-37。

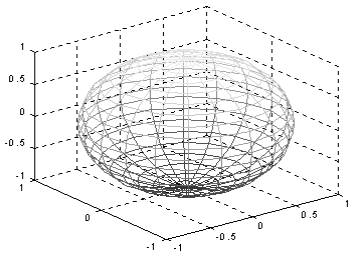


图 7-37

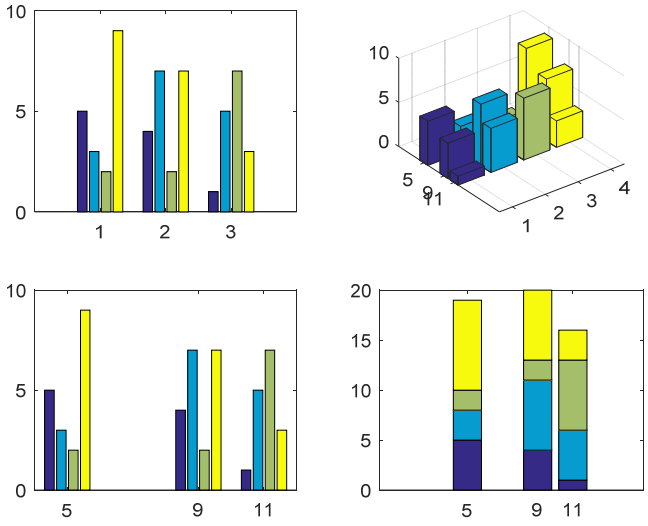
第 3 节　绘制直方图

matlab 中函数 bar 绘制直方图中的应用函数 bar(x)可以绘制直方图，这对统计或者数据采集非常直观实用。它共有四种形式：bar,bar3,barh 和 bar3h，其中 bar 和 bar3 分别用来绘制二维和三维竖直方图，barh 和 bar3h 分别用来绘制二维和三维水平直方图，调用格式是：

bar(x,y)其中 x 必须单调递增或递减，y 为 n×m 矩阵，可视化结果为 m 组，每组 n 个垂直柱，也就是把 y 的行画在一起，同一列的数据用相同的颜色表示；bar(x,y,width)（或 bar(y,width)）指定每个直方条的宽度，如 width>1，则直方条会重叠，默认值为 width=0.8；bar(…,'grouped')使同一组直方条紧紧靠在一起；bar(…,'stack')把同一组数据描述在一个直方条上。

例 5.3.2

```
y=[5 3 2 9;4 7 2 7;1 5 7 3];
subplot(2,2,1),bar(y)
x=[5 9 11];
subplot(2,2,2),bar3(x,y)
subplot(2,2,3),bar(x,y,'grouped')
subplot(2,2,4),bar(x,y,'stack')
```



第 4 节　求零点

正如人们对寻找函数的极点感兴趣一样，有时寻找函数过零或等于其它常数的点也非常重要。一般试图用解析的方法寻找这类点非常困难，而且很多时候是不可能的。在上述函数 humps 的图中(如图 13.3 所示)，该函数在 x=1.2 附近过零。

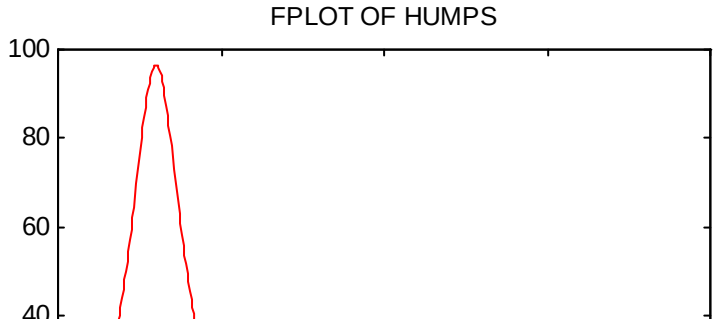


图 13.3 humps 函数的图形

MATLAB 再一次提供了该问题的数值解法。函数 fzero 寻找一维函数的零点。为了说明该函数的使用，让我们再运用 humps 例子。

```
>> xzero=fzero('humps', 1.2)      %   在 1.2 附近找 0 点
xzero=
     1.2995
>>yzero=humps(xzero)             %   evaluate at xzero
yzero=
    0
```

所以，humps 的零点接近于 1.3。如前所述，寻找零点的过程可能失败。如果 fzero 没有找到零点，它将停止运行并提供解释。

当调用函数 fzero 时，必须给出该函数的名称。但由于某种原因，它不能接受以 x 为自变量的字符串来描述的函数。

fzero 不仅能寻找零点，它还可以寻找函数等于任何常数值的点。仅仅要求一个简单的再定义。例如，为了寻找 f(x)=c 的点，定义函数 g(x)=f(x)-c，然后，在 fzero 中使用 g(x)，就会找出 g(x)为零的 x 值，它发生在 f(x)=c 时。

第 5 节　积分

一个函数的积分或面积也是它的另一个有用的属性。MATLAT 提供了在有限区间内，数值计算某函数下的面积的三种函数：trapz , quad 和 quadl。函数 trapz 通过计算若干梯形面积的和来近似某函数的积分，这些梯形如图 13.4 所示，是通过使用函数 humps 的数据点形成。

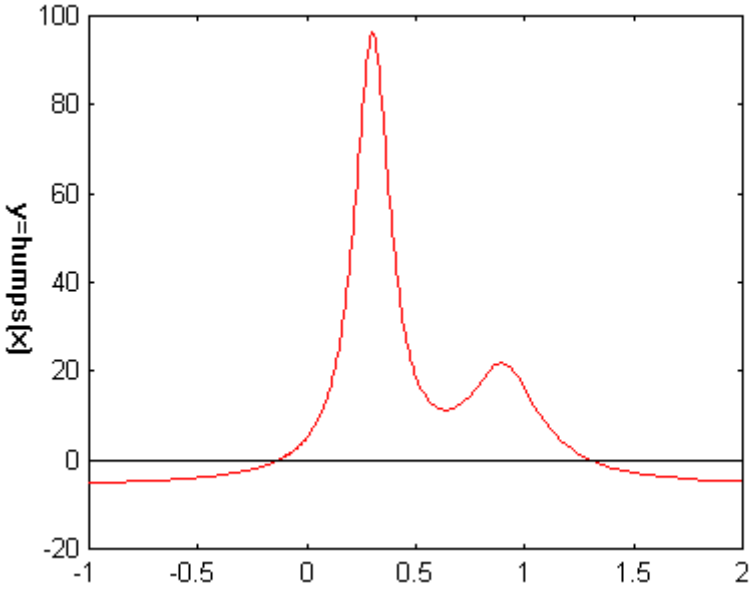


图 13.4 粗略的梯形逼近曲线下的面积示意图

从图中可明显地看出，单个梯形的面积在某一段欠估计了函数真正的面积，而在其它段又过估计了函数的真正面积。如同线性插值，当梯形数目越多时，函数的近似面积越准确。例如，在图 13.4 中，如果我们大致增加一倍数目的梯形，我们得到如下页（如图 13.5）

所示的更好的近似结果。

对如上所示的两个曲线，用 trapz 在区间-1<x<2 上计算 y=x^2 下面的面积：

```
>>x=-1 : 0.1 : 2;                    %   rough approximation
>> y=x.^2;
>>area=trapz(x , y)                  %   call trapz just like the plot command
area =
    3.0050
>>x=-1 : 0.01 : 2;                   %   better approximation
>> y=x.^2;
>>area=trapz(x , y)
area =
    3.0000
```

自然地，上述两个结果不同。基于对图形的观察，粗略近似可能低估了实际面积。除非特别精确，没有准则说明哪种近似效果更好。很明显，如果人们能够以某种方式改变单个梯形的宽度，以适应函数的特性，即当函数变化快时，使得梯形的宽度变窄，这样就能够得到更精确的结果。

例　求定积分。

(1) 建立被积函数文件 fesin.m。

```
function f=fesin(x)
f=exp(-0.5*x).*sin(x+pi/6);
```

(2) 调用数值积分函数 quad 求定积分。

```
[S,n]=quad('fesin',0,3*pi)
S =
    0.9008
n =
     77
```

例　求定积分。

(1) 被积函数文件 fx.m。

```
function f=fx(x)
f=x.*sin(x)./(1+cos(x).*cos(x));
```

(2) 调用函数 quadl 求定积分。

```
I= quadl('fx',0,pi)
I =
     2.4674
```

符号积分

```
f='x^2'
int(f,'x',0, 1)
```