7-1 特殊图形的绘制

1 柱状图绘制

2 结题图形绘制

3 方向和速度矢量图

4 等值线绘制

5 三位旋转图形绘制

7-2 误差理论

1 误差的来源

2 误差表示法

3 误差的积累与传播

4 工程计算中应注意的问题

5 MATLAB数据精度控制

7-1 特殊图形的绘制

1 柱状图绘制

2 结题图形绘制

3 方向和速度矢量图

4 等值线绘制

5 三位旋转图形绘制

1 使用bar()函数绘制柱状图

1．绘制二维柱状图

函数bar()和barh()用于绘制二维柱状图，分别绘制纵向和横向图形。在默认情况下，bar()函数绘制的条形图将矩阵中的每个元素表示为“条形”，“条形”的高度表示元素的大小，横坐标上的位置表示不同的行。在图形中，每一行的元素会集中在一起。

MATLAB中主要有四个函数用于绘制条形图。

绘制条形图函数

函 数	说 明	函 数	说 明
bar	绘制纵向条形图	bar3	绘制三维纵向条形图
barh	绘制横向条形图	bar3h	绘制三维横向条形图

bar()函数的调用格式如下：

(1) bar(Y)：使用bar()函数水平或垂直显示、绘制向量或矩阵值，bar()函数不接受多变量。bar(Y)对Y绘制条形图。如果Y为矩阵，Y的每一行聚集在一起。横坐标表示矩阵的行数，纵坐标表示矩阵元素值的大小。

(2) bar(x,Y)：指定绘图的横坐标。x的元素可以非单调，但是x中不能包含相同的值。

(3) bar(...,width)：指定每个条形的相对宽度。条形的默认宽度为0.8。

(4) bar(...,‘style’)：指定条形的样式。style的取值为“grouped”或者“stacked”，如果不指定，则默认为“grouped”。两个取值的意义分别为：

- grouped：绘制的图形共有m组，其中**m为矩阵Y的行数，每一组有n个条形，n为矩阵Y的列数**，Y的每个元素对应一个条形。
- stacked：绘制的图形有m个条形，**每个条形为第m行的n个元素的和，每个条形由多个(n个)色彩构成**，每个色彩对应相应的元素。

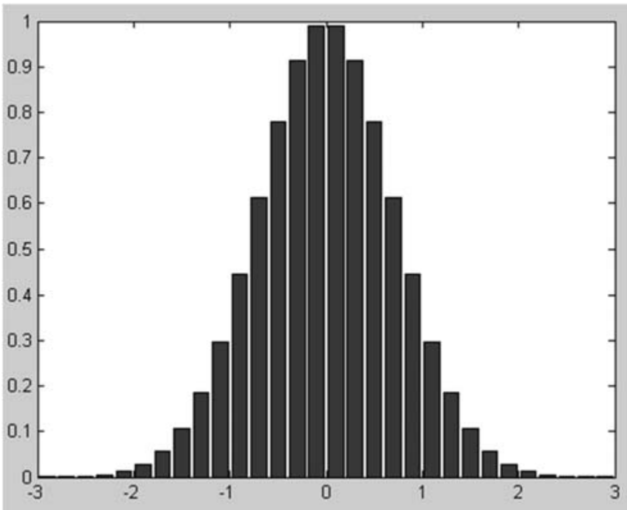
北京邮电大学

(5) bar(...,‘bar_color’)：指定绘图的色彩，所有条形的色彩由“bar_color”确定，“bar_color”的取值与plot绘图的色彩相同。

bar(x)显示x向量元素的条形图。输入下列命令：

```
x = -2.9:0.2:2.9;
bar(x,exp(-x.*x),'r')
```

绘制出二维条状图形。



北京邮电大学

2．绘制三维柱状图

bar3()和bar3h()用于绘制三维柱状图，分别绘制纵向图形和横向图形。这两个函数的用法相同，并且与函数bar()和barh()的用法类似，读者可以与bar()函数和barh()函数进行比较学习。下面以bar3()函数为例介绍这两个函数的用法。bar3()函数的调用格式如下：

(1) bar3(Y)：绘制三维条形图，Y的每个元素对应一个条形，如果Y为向量，则x轴的范围为[1:length(Y)]，如果Y为矩阵，则x轴的范围为[1:size(Y,2)]，即为矩阵Y的列数，图形中，矩阵每一行的元素聚集在相对集中的位置。

(2) bar3(x,Y)：指定绘制图形的行坐标，规则与bar函数相同。

(3) bar3(...,width)：指定条形的相对宽度，规则与bar函数相同。

(4) bar3(...,'style')：指定图形的类型，“style”的取值可以为“detached”、“grouped”或“stacked”，其意义分别为：

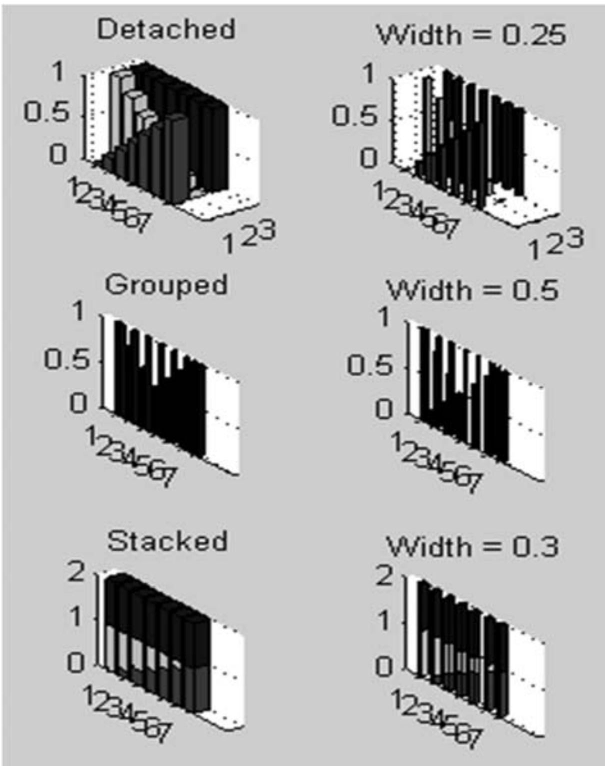
- detached：显示Y的每个元素，在x方向上，Y的每一行为一个相对集中的块；
- grouped：显示m组图形，每组图形包含n个条形，**m和n分别对应矩阵Y的行和列**；
- stacked：意义与bar中的参数相同，将Y的每一行显示为一个条形，每个条形包括不同的色彩，对应于该行的每个元素。

(5) bar3(...,LineSpec)：将所有的条形指定为相同的颜色，颜色的可选值与plot()函数的可选值相同。

例　绘制三维柱状图。

解 程序如下：

```
Y = cool(7);
subplot(3,2,1)
bar3(Y,'detached')
title('Detached')
subplot(3,2,2)
bar3(Y,0.25,'detached')
title('Width = 0.25')
subplot(3,2,3)
bar3(Y,'grouped')
title('Grouped')
subplot(3,2,4)
bar3(Y,0.5,'grouped')
title('Width = 0.5')
subplot(3,2,5)
bar3(Y,'stacked')
title('Stacked')
subplot(3,2,6)
bar3(Y,0.3,'stacked')
title('Width = 0.3')
colormap([1 0 0;0 1 0;0 0 1])
```

2 使用stairs()绘制阶梯图形

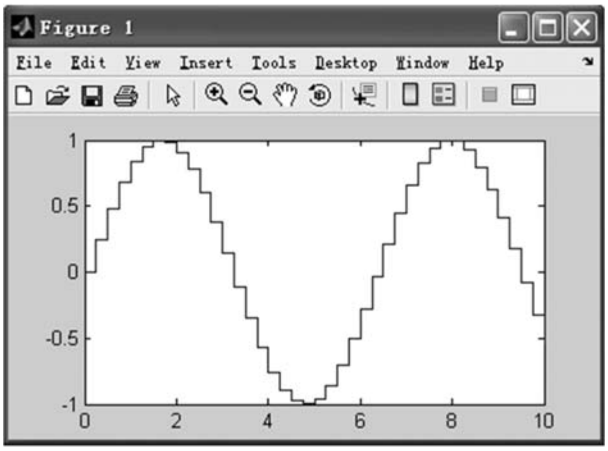
阶梯图主要用于绘制数字采样数据的时间关系曲线图，使用stairs()函数可以绘制阶梯状图形。stairs()函数的调用格式如下：

- stairs(Y)：绘制Y的元素的阶梯状图形。当Y是向量时，X轴的缩放范围是1～length(Y)，当Y是矩阵时，X轴的缩放范围是1～Y的行数。
- stairs(X,Y)：在X指定的位置绘制Y的元素的阶梯图形。X必须与Y的大小相同，当Y是矩阵时，X可以是行或列向量，例如：length(X) = size(Y,1)。

- stairs(...,LineSpec)：指定线型、符号和颜色等属性。

例如，输入下列命令：

```
x=0:0.25:10;
stairs(x,sin(x));
```



3 方向和速度矢量图形

MATLAB提供了一些函数用于绘制方向矢量和速度矢量图形，这些函数有compass()、feather()、quiver()和quiver3()。

绘制矢量图形的函数

函　　数	功 能 描 述
compass	罗盘图，绘制、显示极坐标图形中的极点发散出来的矢量图
feather	羽状图，绘制向量，显示从一条水平线上均匀间隔的点所发散出来的矢量图。向量起点位于与 x 轴平行的直线上，长度相等
quiver	二维矢量图，绘制二维空间中指定点的方向矢量，显示由(u,v)矢量特定的二维矢量图
quiver3	三维矢量图，绘制三维空间中指定点的方向矢量，显示由(u,v,w)矢量特定的三维矢量图

1．罗盘图的绘制

在MATLAB中，罗盘图由函数compass()绘制，该函数的调用格式如下：

(1) compass(U,V)：绘制罗盘图，数据的x分量和y分量分别由U和V指定；

(2) compass(Z)：绘制罗盘图，数据由Z指定；

(3) compass(...,LineSpec)：绘制罗盘图，指定线型；

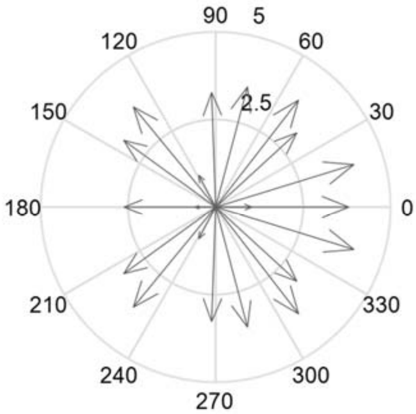
(4) compass(axes_handle,...)：在“axes_handle”指定的坐标系中绘制罗盘图；

(5) h = compass(...)：绘制罗盘图，同时返回图形句柄。

例 绘制罗盘图。

解 程序如下：

```
M = randn(20,20);
Z = eig(M);
compass(Z)
```



2．羽状图的绘制

羽状图由函数feather()绘制，该函数的调用格式如下：

(1) feather(U,V)：绘制由U和V指定的向量；

(2) feather(Z)：绘制由Z指定的向量；

(3) feather(...,LineSpec)：指定线型；

(4) feather(axes_handle,...)：在指定的坐标系中绘制羽状图；

(5) h = feather(...)：绘制羽状图，同时返回图像句柄。

```
theta = -pi/2:pi/16:pi/2;
r = 2*ones(size(theta));
[u,v] = pol2cart(theta,r);
feather(u,v)
```

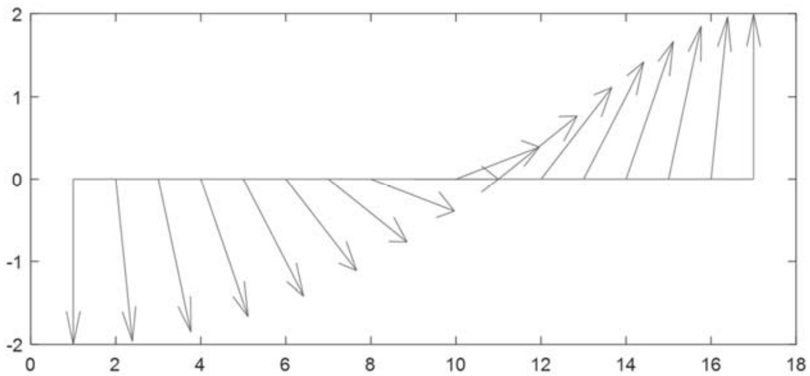


图5-58 绘制羽状图

3. 矢量图的绘制

矢量图在空间中指定点绘制矢量。矢量图通常绘制在其他图形中，显示数据的方向，如在梯度图中绘制矢量图用于显示梯度的方向。

MATLAB用于绘制二维矢量图和三维矢量图的函数，分别为quiver()和quiver3()，两个函数的调用格式基本相同。函数quiver()的主要调用格式如下：

(1) quiver(x,y,u,v)：绘制矢量图，参数x和y用于指定矢量的位置，u和v用于指定待绘制的矢量；

(2) quiver(u,v)：绘制矢量图，矢量的位置采用默认值。

函数quiver3()的主要调用格式如下：

(3) quiver3(x,y,z,u,v,w)：函数quiver3()使用元素(u,v,w)在点(x,y,z)绘制三维矢量图，u,v,w,x,y和z都是实数值，不是复数，并且大小相同。

(4) quiver3(z,u,v,w)：在矩阵z指定的等距离表面的点绘制三维矢量图，quiver3()根据它们之间的距离自动缩放，以防止它们重叠。

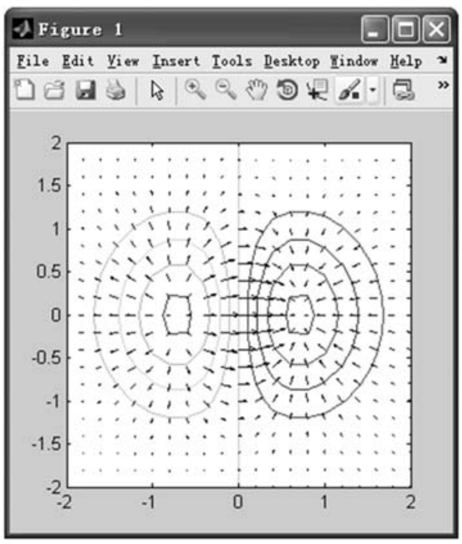
(5) quiver3(...,scale)：按照缩放系数scale自动缩放，以防止它们重叠。scale = 2 时，长度放大一倍；cale = 0.5时，长度缩小一倍；scale = 0 时，无缩放。

(6) quiver3(...,LineSpec)：LineSpec指定线型和颜色。

例 绘制函数的梯度场。

解 (1) 使用下列程序绘制二维矢量图，如图5-59所示。

```
[X,Y] = meshgrid(-2:0.2:2);
Z = X.*exp(-X.^2 - Y.^2);
[DX,DY] = gradient(Z,.2,.2); %gradient
hold on
contour(X,Y,Z)
quiver(X,Y,DX,DY)
colormap hsv;
hold off
```



4 等值线的绘制

等值线函数

函数名	功能描述
clabel	使用等值矩阵生成标注，在二维等值线中添加高度值标注，并将标注显示在当前图形
contour	绘制、显示指定数据矩阵 Z 的二维等高线图
contour3	绘制、显示指定数据矩阵 Z 的三维等高线图
contourf	绘制、显示矩阵 Z 的二维等高线图，并在各等高线之间用实体颜色填充
contourc	用于计算等值线矩阵，通常由其他函数调用
meshc	创建一个与二维等高线图匹配的网格线图
surfc	创建一个与二维等高线图匹配的曲面图

1. 二维等值线

contour()、contour3()等函数用于绘制二维、三维等值线，其调用格式如下：

(1) contour(Z)：绘制矩阵Z的等值线，绘制时将Z在x-y平面上进行插值，等值线的数量和数值由系统根据Z自动确定；

(2) contour(Z,n)：绘制矩阵Z的等值线，等值线数目为n；

(3) contour(Z,v)：绘制矩阵Z的等值线，等值线的值由向量v确定；

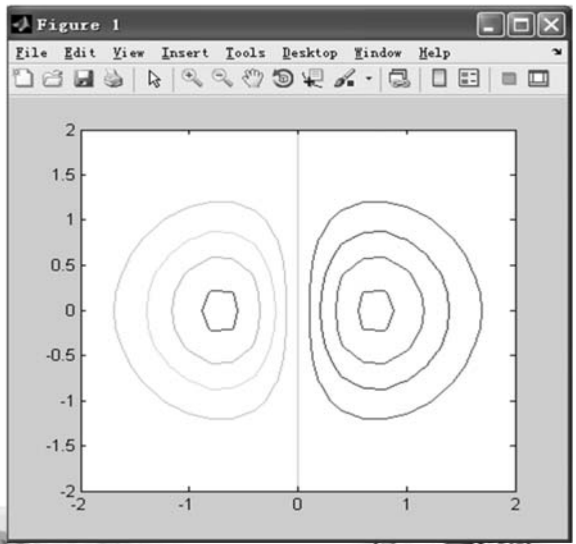
(4) contour(X,Y,Z)、contour(X,Y,Z,n)、contour(X,Y,Z,v)：绘制矩阵Z的等值线，坐标值由矩阵X和Y指定，矩阵X、Y、Z的维数必须相同；

(5) contour(...,LineSpec)：利用指定的线型绘制等值线；

(6) [C,h] = contour(...)：绘制等值线，同时返回等值线矩阵和图形句柄。

例如，上例的函数用contour()函数绘制二维等值线，如下图所示。

```
[X,Y] = meshgrid(-2:0.2:2);
Z = X.*exp(-X.^2 -Y.^2);
contour(X,Y,Z)
colormap hsv
```



2．三维等值线

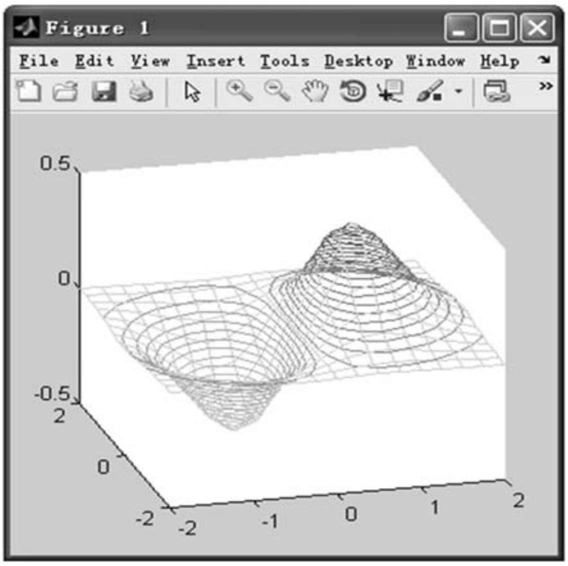
contour3()函数用于绘制三维等值线，其调用格式与contour()函数的基本相同。

(1) contour3(Z)：绘制矩阵Z的三维等值线，Z看做是相对于x-y平面的高度，Z最少是包含2个不同值的2 × 2的矩阵，contour号和值基于Z的最小和最大值自动选择，x、y轴的范围是[1:n]和[1:m]，[m,n] = size(Z)。

(2) contour3(Z,n)：根据n的值绘制矩阵Z的三维等值线。

例如，上例用contour3()函数绘制三维等值线。

```
[X,Y] = meshgrid([-2:.25:2]);
Z = X.*exp(-X.^2-Y.^2);
contour3(X,Y,Z,30)
surface(X,Y,Z,'EdgeColor’,
[.8 .8 .8],'FaceColor','none')
grid off
view(-15,25)
colormap cool
```



5 饼形图

饼状图是一种统计图形，用于显示每个元素占总体的百分比。在统计学中，人们经常用饼形图来表示各个统计量占总量的份额，饼形图可以显示向量或矩阵中的元素占所有元素总和的百分比。MATLAB提供了pie()函数和pie3()函数，分别用于绘制二维饼形图和三维饼形图。函数pie()的调用格式如下：

(1) pie(X)：绘制X的饼状图，X的每个元素占一个扇形，其顺序为从饼状图上方正中开始，逆时针为序，分别为X的各个元素，如果X为矩阵，则按照各列的顺序排列。

- 在绘制饼状图时，如果X的元素和超过1，则按照每个元素所占有的百分比绘制图形；
- 如果X的元素的和小于1，则按照每个元素的值绘制图形，绘制的图形不是一个完整的圆形。

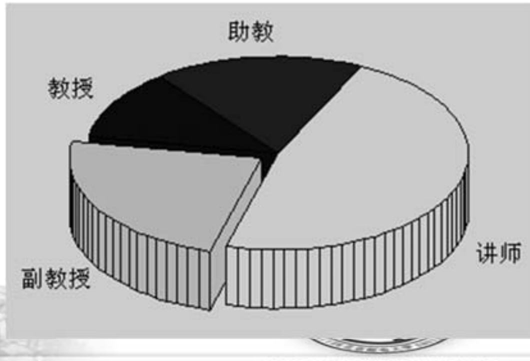
(2) pie(X,explode)：参数explode设置相应的扇形偏离整体图形，用于突出显示。explode是一个与X维数相同的向量或矩阵，其元素为0或者1，非0元素对应的扇形从图形中偏离。

(3) pie(...,labels)：标注图形，labels为元素为字符串的单元数组，元素个数必须与 X 的个数相同。

pie3()函数的调用方法与pie()函数相同。

例如：

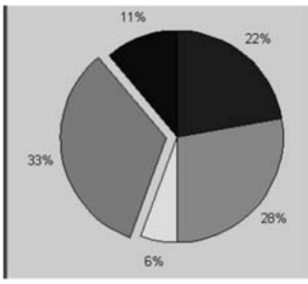
```
x=[2,4,8,3];
explode = [0 1 0 0 ];
labels={'教授','副教授','讲师','助教'};
pie3(x,explode,labels)
```



例 绘制二、三维饼状图。

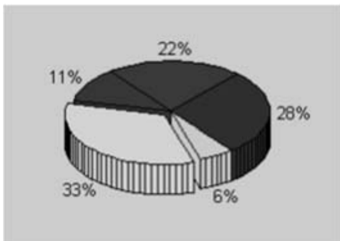
解 (1) 下列程序绘制二维饼状图。

```
x = [1 3 0.5 2.5 2];
explode = [0 1 0 0 0];
pie(x,explode)
colormap jet
```



(2) 下列程序绘制三维饼状图。

```
x = [1 3 0.5 2.5 2];
explode = [0 1 0 0 0];
pie3(x,explode)
colormap jet
```



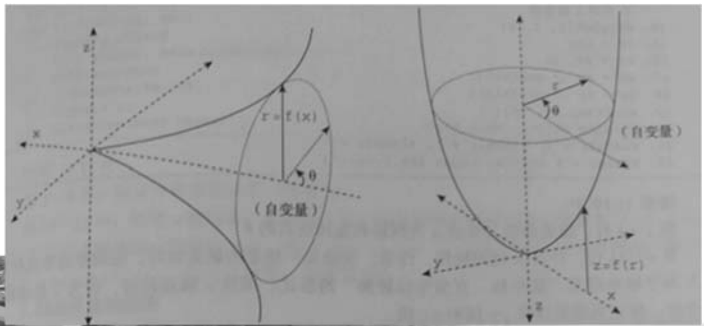
6 三维旋转体的绘制

1、旋转连续函数

考虑围绕x轴和z轴旋转连续函数v=f(u)

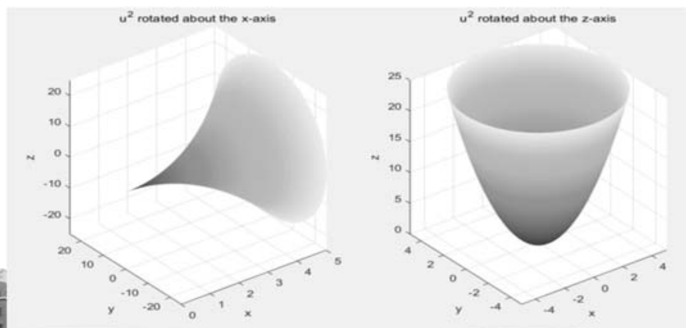
(1)围绕x轴旋转v=f(u)时，可以把方程看作r=f(x)，旋转的逻辑如图所示，x是自变量，y和z的值可通过“极坐标-直角坐标转换”得到： $y=r\cos(\theta); z=r\sin(\theta)$

（2）围绕z轴旋转v=f(u)时，可以把方程看作z=f(r)，则得到 $x=r\cos(\theta); y=r\sin(\theta)$



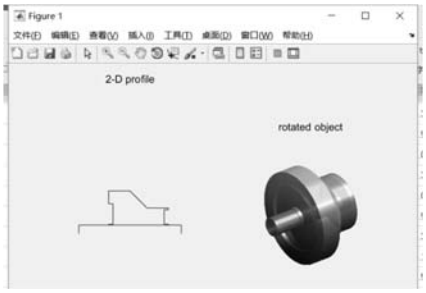
代码实现：

```
facets=100;
u=linspace(0,5,facets);
th=linspace(0,2*pi,facets);
[uu,tth]=meshgrid(u,th);
subplot(1,2,1);
rr = uu.^2;
xx=uu;
yy=rr.*cos(tth);
zz=rr.*sin(tth);
surf(xx,yy,zz,xx);
shading interp, axis tight
xlabel('x'),ylabel('y'),zlabel('z');
title('u^2 rotated about the x-axis')
subplot(1,2,2)
rr=uu;
zz=rr.^2;
xx=rr.*cos(tth);
yy=rr.*sin(tth);
surf(xx,yy,zz);
shading interp, axis tight
xlabel('x'),ylabel('y'),zlabel('z');
title('u^2 rotated about the z-axis')
```



2、旋转离散函数

旋转体的轮廓不是只有连续函数，机器零件的二维轮廓和该轮廓围绕x轴旋转后形成的图行如下图所示。如何画出零件立体图形？



代码实现：

```
u = [0 0 3 3 1.75 1.75 2 2 1.75 1.75 3 4 ...
  5.25 5.25 5 5 5.25 5.25 3 3 6 6];
v = [0 .5 .5 .502 .502 .55 .55 1.75 1.75 ...
  2.5 2.5 1.5 1.5 1.4 1.4 ...
  .55 .55 .502 .502 .5 .5 0];
subplot(1, 2, 1)
plot(u, v, 'k')
axis ([-1 7 -1 3]), axis equal, axis off
title('2-D profile')
facets = 200;
subplot(1, 2, 2)
[xx tth] = meshgrid( u, linspace(0, 2*pi, facets) );
[rr tt] = meshgrid( v, 1:facets);
yy = rr .* cos(tth);
zz = rr .* sin(tth);
surf(xx, yy, zz);
shading interp
axis square, axis tight, axis off
colormap bone
lightangle(60, 45)
alpha(0.8)
title('rotated object')
```

3、围绕任意轴旋转

不是只有围绕x轴、y轴或z轴旋转才能生成旋转体。将z=f(x)围绕任意轴旋转的最简单的方法是：

（1）计算将旋转轴沿x轴放置的矩阵；

（2）通过旋转变换u和v；

（3）将变换后的u和v围绕x轴旋转；

（4）在结果曲面上变换回来；

感兴趣的同学可以课后研究一下

7-2 误差理论

1 误差的来源

2 误差表示法

3 误差的积累与传播

4 工程计算中应注意的问题

5 MATLAB数据精度控制

1 误差的来源

- 模型误差：将实际问题转化为数学问题时，必然要进行必要的抽象和简化，因此数学模型只是客观现象的一种近似、粗糙的描述，这种数学模型与实际问题之间出现的误差成为**模型误差**。
- 观测误差：在建立的数学模型中，往往有一些根据观测得到的物理量，如温度、长度、电压等，而观测不可避免会带来误差，这种误差称为**观测误差**。

- 截断误差：实际问题中，常常遇到只有通过无限过程才能得到结果的问题，在计算机上只能采用有限过程，于是产生了有限过程代替无限过程的误差，称为**截断误差**。

截断误差示例

例 计算定积分$I=\int_0^1 e^{-x^2}\,dx$的值.

结题思路：已知函数e^{-x^2}的原函数虽然存在但不能用初等函数表示，因此考虑用泰勒级数展开，这里利用6阶Taylor展开式近似求解改积分值：

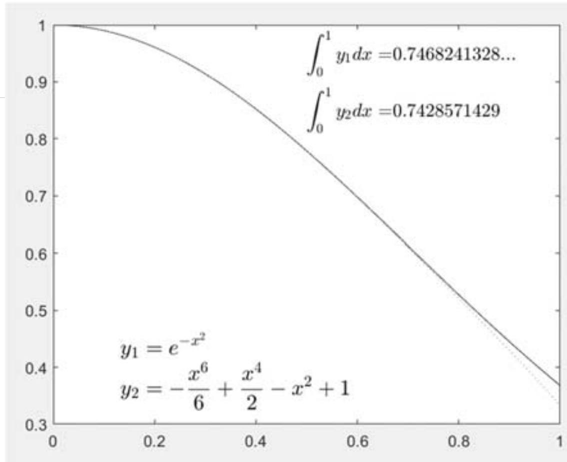
```
>> syms x
>> P = taylor(exp(-x^2),x,'order',7)
```

因此，$$I=\int_0^1 e^{-x^2}\,dx\approx\int_0^1\left(-\frac{x^6}{6}+\frac{x^4}{2}-x^2+1\right)dx=\left(-\frac{x^7}{42}+\frac{x^5}{10}-\frac{x^3}{3}+x\right)\Bigg|_0^1$$

$$=-\frac{1}{42}+\frac{1}{10}-\frac{1}{3}+1\approx0.742857$$

该结果与准确值I = 0.746824132812427025399467436…存在误差。

画图呈现截断误差




```
syms x;
P = taylor(exp(-x^2),x,'order',7)
x1 = linspace(0,1);
plot(x1,exp(-x1.^2),'k');
hold on
P1 = inline(P);
plot(x1,P1(x1),':');
legend({'$${y_1} = {e^{-{x^2}}}$$',...
  '$${y_2}=-\frac{{{x^6}}}{6}+\frac{{{x^4}}}{2}-{x^2}+1$$'},...
  'interpreter','latex','fontsize',14,'Location','southwest')
text(0.5,0.95,['$$\int_0^1{{y_1}dx} =$$',...
  char(vpa(int(exp(-x^2),0,1),10)),'...'],...
  'interpreter','latex','fontsize',12)
text(0.5,0.85,['$$\int_0^1{{y_2}dx} =$$',...
  char(vpa(int(P,x,0,1),10))],...
  'interpreter','latex','fontsize',12)
```

- 舍入误差：由于计算机的字长有限，在进行数值计算的过程中，对计算得到的中间结果数据要使用“四舍五入”或其他规则取近似值，因而使计算过程产生误差，这种误差成为**舍入误差**。

误差来源

- 模型误差
- 观测误差
- 截断误差
- 舍入误差

2 误差的表示

- 绝对误差的数学表达：$|e| = |x^* - x| < \varepsilon$，其中$x^*$为准确值，x为近似值，e为绝对误差。$\varepsilon$为近似值x的绝对误差限。
- 相对误差的数学表达： $e_r = \frac{x^*-x}{x^*}$，由于在实际计算中真值通常难以求得，常用下式代替： $\bar{e_r} = \frac{x^*-x}{x}$

例：有两个数值$x_1^* = x_1 \pm e_1 = 10 \pm 1, x_2^* = x_2 \pm e_2 = 1000 \pm 5$ ，显然，虽然x_2绝对误差较大，但是其近似程度更高。

3 有效数字

- 有效数字的表示：
  - 若近似值x的误差限是某一数位的半个单位，且该位到x的第一位非零数字共有n位，那么我们就说x具有n位有效数字。科学计数法表示为
  - $x = \pm 0.a_1 a_2 \cdots a_n \times 10^m$
  - $e = |x^* - x| < \frac{1}{2} \times 10^{m-n}$，有效位数为n位。

例：若π的近似取值为π =3.1415，问π有几位有效数字？
解：因为π =3.1415 = 0.31415*10^1, 且

$$|\pi^* - \pi| < \frac{1}{2} \times 10^{-3} = \frac{1}{2} \times 10^{1-4}$$

因此，有4位有效数字，精确到小数点后第3位。

- 根据相对误差推定有效数字位数
- 由$e = |x^* - x| < \frac{1}{2} \times 10^{m-n}$知，在m相同情况下，n越大则误差越小，且相对误差可表示为
- $e_r = \left|\frac{x^* - x}{x^*}\right| < \frac{\frac{1}{2} \times 10^{m-n}}{(a_1 \times 10^{-1} + a_2 \times 10^{-2} \cdots a_n \times 10^{-n}) \times 10^m} <$

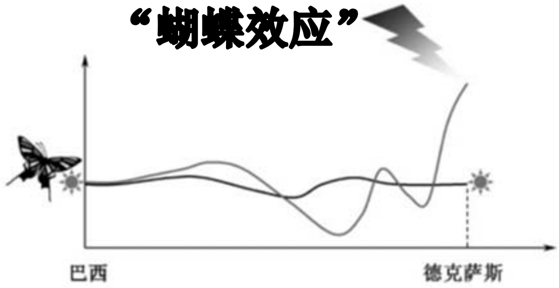
$\frac{10^{m-n}}{2a_1 \times 10^{m-1}} = \frac{1}{2a_1} \times 10^{1-n}$

- 可见：一个近似值的有效位数越多，其相对误差限越小；反过来，可根据x的误差限推导x至少有n位有效数字。

4 误差的积累与传播

4.1 误差的积累

数值计算方法中存在“蝴蝶效应”，极其微小的误差在计算中不断积累与传播，最终导致计算结果与理论结果严重偏差，使得计算方法完全失效。这种现象称为计算方法的病态性。



1979年12月，Lorenz在华盛顿的美国科学促进会的一次讲演中提出，一只蝴蝶在巴西扇动翅膀，有可能会在美国的德克萨斯引起一场龙卷风。

误差积累实例

例：计算定积分$I_n=\frac{1}{e}\int_0^1 x^n e^x\,dx$, n = 0,1,2,…

解法1：由分部积分法可得递推式：$I_n=1-nI_{n-1}$，

$$I_0=\frac{1}{e}\int_0^1 e^x\,dx=1-\frac{1}{e}\approx 0.63212056\rightarrow I_0^*$$

则初始误差为 $|\mathrm{E}_0|=|I_0-I_0^*|<0.5\times10^{-8}$

再由递推式可得

$I_1^*=1-I_0^*=0.36787944$ $I_2^*=1-2I_1^*=0.26424112$

$I_3^*=1-3I_2^*=0.20727664$ $I_4^*=1-4I_3^*=0.17089344$

$I_5^*=1-5I_4^*=0.14553280$ $I_6^*=1-6I_5^*=0.12680320$

$I_7^*=1-7I_6^*=0.11237760$ $I_8^*=1-8I_7^*=0.10097920$

$I_9^*=1-9I_8^*=0.09118720$ $I_{10}^*=1-10I_9^*=0.08812800$

$I_{11}^*=1-11I_{10}^*=0.03059200$ $I_{12}^*=1-12I_{11}^*=0.63289600$

$I_{13}^*=1-13I_{12}^*=-7.2276480$ $I_{14}^*=1-14I_{13}^*=102.18707$

$I_{15}^*=1-15I_{14}^*=-1531.8061$ 严重错误

原因解析

该题中 $|\mathrm{E}_n|=|I_n-I_n^*|=|(1-nI_{n-1})-(1-nI_{n-1}^*)|=n\,|\mathrm{E}_{n-1}|=\cdots=n!\,|\mathrm{E}_0|$

可见，此种算法中微小的误差能够快速积累，导致算法完全失效，该算法即称为不稳定的算法。

改进的算法

逆向推导方法：$I_{n-1} = \frac{1}{n} - \frac{I_n}{n}$，通过估计一个$I_N$，在反推 $I_n\ (n \ll N)$。

由 $\frac{1}{e}\int_0^1 x^n e^0 dx < I_n < \frac{1}{e}\int_0^1 x^n e^1 dx \Rightarrow 0 < \frac{1}{e(n+1)} < I_n < \frac{1}{n+1} < 1$ 可得

$I_N^* = \frac{1}{2}\left[\frac{1}{e(N+1)} + \frac{1}{N+1}\right] \approx I_N$，故取N=15，得到$I_{15}^* \approx$ 0.042746233，从而根据上述逆向提到公式得到

$I_{14}^* = \frac{1}{15} - \frac{I_{15}^*}{15} \approx 0.063816918$ $\quad I_{13}^* = \frac{1}{14} - \frac{I_{14}^*}{14} \approx 0.066870220$

$I_{12}^* = \frac{1}{13} - \frac{I_{13}^*}{13} \approx 0.071779214$ $\quad I_{11}^* = \frac{1}{12} - \frac{I_{12}^*}{12} \approx 0.077351732$

$I_{10}^* = \frac{1}{11} - \frac{I_{11}^*}{11} \approx 0.083877115$ $\quad I_{9}^* = \frac{1}{10} - \frac{I_{10}^*}{10} \approx 0.091612288$

误差积累公式：

$$|E_{N-1}| = |I_{N-1} - I_{N-1}^*| = \left|\frac{1}{N}(1-I_N) - \frac{1}{N}(1-I_N^*)\right| = \frac{1}{N}|E_N|$$

$I_8^* = \frac{1}{9} - \frac{I_9^*}{9} \approx 0.10093197$ $\quad I_7^* = \frac{1}{8} - \frac{I_8^*}{8} \approx 0.11238350$

$I_6^* = \frac{1}{7} - \frac{I_7^*}{7} \approx 0.12680236$ $\quad I_5^* = \frac{1}{6} - \frac{I_6^*}{6} \approx 0.14553294$

$I_4^* = \frac{1}{5} - \frac{I_5^*}{5} \approx 0.17089341$ $\quad I_3^* = \frac{1}{4} - \frac{I_4^*}{4} \approx 0.20727665$

$I_2^* = \frac{1}{3} - \frac{I_3^*}{3} \approx 0.26424112$ $\quad I_1^* = \frac{1}{2} - \frac{I_2^*}{2} \approx 0.36787944$

$I_0^* = \frac{1}{1} - \frac{I_1^*}{1} \approx 0.63212056$

4.2 误差的传播

- 数值计算中误差的传播情况非常复杂，参与运算的数据往往都是近似数，它们都带有误差，而这些数据的误差在各次运算中又会进行传播，使计算结果产生一定的误差。不同的计算方法会导致结果误差的不同。

A. 四则运算中误差的传播

$$\begin{cases} e(x_1 \pm x_2) \approx e(x_1) \pm e(x_2) \\ e(x_1 x_2) \approx x_2 e(x_1) + x_1 e(x_2) \\ e\left(\frac{x_1}{x_2}\right) \approx \frac{1}{x_2} e(x_1) - \frac{x_1}{x_2^2} e(x_2) \quad (x_2 \neq 0) \\ e_r(x_1 \pm x_2) \approx \frac{x_1}{x_1 \pm x_2} e_r(x_1) \pm \frac{x_2}{x_1 \pm x_2} e_r(x_2) \\ e_r(x_1 x_2) \approx e_r(x_1) + e_r(x_2) \\ e_r\left(\frac{x_1}{x_2}\right) \approx e_r(x_1) - e_r(x_2) \quad (x_2 \neq 0) \end{cases}$$

四则运算误差传播实例

例：计算$\sqrt{2010}-\sqrt{2009}$的值，并分析计算结果具有几位有效数字。

方法1：$x_1^*=\sqrt{2010}\approx 44.833024$， $x_2^*=\sqrt{2009}\approx 44.821870$

则$x_1^*-x_2^*\approx x_1-x_2=44.833024\text{- }44.821870=0.011154$

$$|e(x_1-\ x_2)|\approx|e(x_1)-e(x_2)|<|e(x_1)|+|e(x_2)|<\frac{1}{2}\times 10^{-6}+\frac{1}{2}\times 10^{-6}<\frac{1}{2}\times 10^{-5}$$

因此，该方法所得结果至少具有4位有效数字。

方法2：

$$（2）\ x_1^*-x_2^*=\frac{1}{x_1^*+x_2^*}\approx\frac{1}{x_1+x_2}=\frac{1}{44.833024+44.821870}\approx 0.011153881$$

$$|e(x_1-x_2)|=e\left(\frac{1}{x_1+x_2}\right)\approx\left|-\frac{1}{(x_1+x_2)^2}e(x_1+x_2)\right|\approx\left|-\frac{1}{(x_1+x_2)^2}[e(x_1)+e(x_2)]\right|$$

$$\leq\frac{1}{(x_1+x_2)^2}(|e(x_1)|+|e(x_2)|)$$

$$\leq\frac{1}{(44.833024+44.821870)^2}\left(\frac{1}{2}\times 10^{-6}+\frac{1}{2}\times 10^{-6}\right)$$

$$\approx 0.12\times 10^{-9}<\frac{1}{2}\times 10^{-9}$$

这种方法所得结果至少具有8位有效数字。

说明：两个相近的数相减造成有效数字位数减少。

B. 函数运算中误差的传播

记有函数$y^* = f(x^*)$, 设x是x^*的近似值，相应的函数的近似值为y，若f(x)可微，则 $e(y) = y^* - y = f(x^*) - f(x) = f'(x)(x^* - x) + \frac{1}{2} f''(\xi)(x^* - x)^2$　　（ξ介于x与x^*之间）

从而有 $|f(x^*) - f(x)| < |f'(x)|e(x) + \frac{1}{2}|f''(\xi)|\big(e(x)\big)^2$

忽略高阶项，其误差限为 $\varepsilon\big(f(x)\big) \approx |f'(x)|e(x)$

相对误差限为 $\varepsilon_r\big(f(x)\big) = |f'(x)|\frac{e(x)}{f(x)}$

多元函数的误差传播分析

对多元函数$y^* = f(x_1^*, x_2^*, \cdots x_n^*)$，若$x_1, x_2, \cdots x_n$ 分别是$x_1^*, x_2^*, \cdots x_n^*$的近似值，则有

$$e\Big(f(x_1, x_2, \cdots x_n)\Big) \approx \sum_{k=1}^{n} \frac{\partial f(x_1, x_2, \cdots x_n)}{\partial x_k} e(x_k)$$

$$e_r\Big(f(x_1, x_2, \cdots x_n)\Big) \approx \sum_{k=1}^{n} \frac{\partial f(x_1, x_2, \cdots x_n)}{\partial x_k} \frac{e(x_k)}{f(x_1, x_2, \cdots x_n)}$$

$$\varepsilon_r\Big(f(x_1, x_2, \cdots x_n)\Big) = \sum_{k=1}^{n} \left|\frac{\partial f(x_1, x_2, \cdots x_n)}{\partial x_k}\right| \frac{\varepsilon(x_k)}{|f(x_1, x_2, \cdots x_n)|}$$

函数运算中的误差传播实例

例： $f(x,y)=\frac{\cos y}{x}, x=1.30\pm0.005, y=0.871\pm0.0005$ ，求根据x和y的值得到的函数值的有效数字位数。

解：
$$u=f(1.30,0.871)=\frac{\cos 0.871}{1.30}\approx 0.49543$$

由于
$$\frac{\partial f(x,y)}{\partial x}=-\frac{\cos y}{x^2},\quad \frac{\partial f(x,y)}{\partial y}=-\frac{\sin y}{x}$$

，则
$$e(u)=\left|-\frac{cos0.871}{1.30^2}\right|\times 0.005+\left|-\frac{sin0.871}{1.30}\right|\times 0.0005<\frac{1}{2}\times 10^{-2}$$

因此，近似函数值有2位有效数字。

5 工程计算中应注意的问题

A. 避免两个相近的数相减

设x_1 ,x_2 分别是x_1^*, x_2^*的近似值，则$y=x_1-x_2$是$y^*=x_1^*,x_2^*$的近似值，则

$$|e_r(x_1-x_2)|<\left|\frac{x_1}{x_1-x_2}\right||e_r(x_1)|+\left|\frac{x_2}{x_1-x_2}\right||e_r(x_2)|$$

可见，当x_1 ,x_2 很接近时，y的相对误差有可能很大。

B. 避免绝对值过小的数作为除数

设x_1 ,x_2 分别是x_1^*, x_2^*的近似值，则$y=\frac{x_1}{x_2}$是$y^*=\frac{x_1^*}{x_2^*}$的近似值，则

$$\left|e\left(\frac{x_1}{x_2}\right)\right| < \left|\frac{1}{x_2}\right||e\quad(x_1)| + \left|\frac{x_1}{x_2^2}\right||e\quad(x_2)|$$

可见，当x_2 太小时，可能导致商的绝对误差很大。

C. 避免大数吃掉小数

计算机上只能采用有限位数计算，因此，若参加运算的数量级差很大，则在运算中，绝对值小的数往往被绝对值大的数吃掉，造成计算结果失真。

例题：已知a=10^9,b=1,c=-a，使用单精度类型计算a+b+c
解：显然，人为直接计算即可得到准确结果为a+b+c=1
在计算机内，109被存储为10|0.1,1存为1|0.1，在做加法时，两个加数的指数先向大的指数对齐，再将浮点部分相加。
因此，有 $a+b=0.1\times10^{10}+0.0000000001\times10^{10}=0.10000000\times10^{10}=10^9$
计算结果为0，即b被a吃掉了。

D. 先化简再计算，减少计算步骤，避免误差积累

E. 采用数值稳定性好的算法

6 MATLAB的数据精度

6.1 浮点数误差限——eps()函数

MATLAB中存在一个用双精度表示的浮点相对误差限eps，定义为1与大于1的最小数之间的步进距离，用eps获得。

(1) eps：返回从1.0到下一个最大的双精度数的距离，eps = 2^(-52)。例如：

```
>> eps
ans =
     2.2204e-016
```

(2) eps(‘double’)：等同于eps或eps(1.0)。例如：

```
>> eps(‘double’)
ans =
      2.2204e-016
```

(3) eps('single')：等同于eps(single(1.0))或single(2^-23)。例如：

```
>> eps('single')
ans =
    1.1921e-007
```


double类型

S	e[52:62]	f[0:51]

数值计算公式:

当0<e<2047时，value=$(-1)^s \times 2^{e-1023} \times 1.f$;

当e=0,f≠0时，value =$(-1)^s \times 2^{e-1022} \times 0.f$;

当e=0,f=0时，value=$(-1)^s \times 0.0$;

当e=2047,f=0,s=0时，value = +inf;

当e=2047,f=0,s=1时，value = -inf;

当e=2047, f≠0时，value = NaN。

single类型

S	e[23:30]	f[0:22]

数值计算公式:

当$0<e<255$时，$value=(-1)^s \times 2^{e-127} \times 1.f$;

当$e=0,f\neq 0$时，$value =(-1)^s \times 2^{e-126} \times 0.f$;

当$e=0,f=0$时，$value=(-1)^s \times 0.0$;

当$e=127,f=0,s=0$时，$value = +inf$;

当$e=127,f=0,s=1$时，$value = -inf$;

当$e=127, f\neq 0$时，$value = NaN$。

有限精度产生的结果

MATLAB的有限精度的局限性往往会产生不寻常的结果。例如：

```
>> 0.42-0.5+0.08
ans =
  -1.387778780781446e-017
```

出现上述现象的原因是**并不是所有的数字都可以用双精度数精确地表示**。当出现这种情况时，MATLAB会用一个尽可能精确的数字表示，这将会出现误差。实际上，这种误差常常是很小的，并且通常是在比较两个数是否相等时才会体现。

MATLAB有限精度局限性的第二个后果出现在函数运算中。例如：

```
>> sin(0)
ans =
   0
>> sin(pi)
ans =
  1.224646799147353e-016
```

从数学角度来讲，上述两个式子的结果都应该是0，但实际计算结果并非如此，sin(pi)并不为0。

上述两种情况出现的误差都是很小的，都小于eps。

因此，当判断一个运算式的结果是否等于某个数时，不可以直接用逻辑判断符 ==，而应该用两数的差的绝对值小于一个极小数来判断。

例：

```
>>A=0.5-0.48;
>>e=1e-10
>>A==0.02
ans = 0
>>abs(A-0.02)<e
ans = 1
```

观察： 不同数量级的数相减时所产生的误差。

1.4-1-0.4= -1.1102e-16

10.4-10-0.4=3.3307e-16

100.4-100-0.4= 5.6621e-15

1000.4-1000-0.4= -2.2760e-14

10000.4-10000-0.4= -3.6382e-13

100000.4-100000-0.4= -5.8208e-12

1000000.4-1000000-0.4= 2.3283e-11

10000000.4-10000000-0.4= 3.7253e-10

100000000.4-100000000-0.4= 5.9605e-09

1000000000.4-1000000000-0.4= -2.3842e-08

100000000000000.4-100000000000000-0.4= 0.0062

1000000000000000.4-1000000000000000-0.4= -0.0250

10000000000000000.4-10000000000000000-0.4= -0.4000

6.2 MATLAB中运算精度的控制

1. 数值算法：将每个数值都取为16位有效数值，它是运算速度最快的一种算法；
2. 符号算法：把每个数据都变成符号量，这种算法可以得出精确的结果，但是它占据空间多，运算速度也比较慢；
3. 可控精度算法：利用控制精度指令 digits(n) 使此后的运算均以n位有效数字进行，配合vpa()函数使用。

示例

```
>> clear
>> a=1/3;b=1/8;
>> tic, a1=a+b, toc
a1 =
  0.458333333333333
历时 0.001776 秒。
>> tic, a2 = sym(a+b),toc
a2 =
11/24
历时 0.064420 秒。
>> digits(2),tic, a3 = vpa(a+b),toc
a3 =
0.46
历时 0.064978 秒。
```

Homework

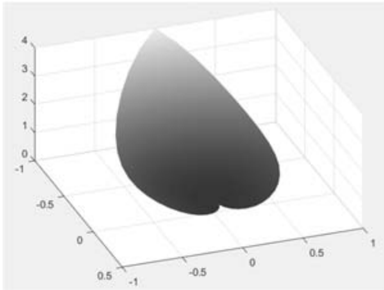
1、一个学校想要建一个钟楼，需要对其进行建模，将方程 $z=1/(x^2+y^2)$ 作为模型，编写一个绘制钟楼的脚本。x、y的取值范围是：-0.75<=x<=0.75，数据间隔为0.05。设置坐标轴，使x、y的所有区域可见，并且z的范围在0到300之间。使用surf（）绘制图像。

[x y]=meshgrid(-.75:.05:.75)

[x y] = meshgrid(1:3)

2、画一颗爱心，绘制方法如下：

1）使用x值（范围是0到2π，间隔是0.05π）和y值（范围是0到1，间隔是0.05）创建一个网格[xx,yy]。



2）定义如下变量

c=[0.1+0.9*(π-abs(xx-π))/π].*yy

aa=c.*cos(xx)

bb=c.*sin(xx)

zz=(-2)*aa.^3+(3/2)*c.^2+0.5

3)使用surf（）函数绘制图形zz vs. aa和zz vs. bb，对色彩进行插值处理

3、编写程序讨论下面两张数列生成算法的稳定性

数列 $p_n = \left\{1, \frac{1}{3}, \frac{1}{9}, \frac{1}{27}, \ldots\right\}$

(1)设 $p_0 = 1$，递推公式$p_n = \frac{1}{3} p_{n-1}$

(2)设$p_0 = 1$， $p_1 = \frac{1}{3}$，递推公式$p_n = \frac{10}{3} p_{n-1} - p_{n-2}$