

Matlab 软件目录

| | |
|----------------------------------|----|
| Matlab 简介 | 5 |
| 0、前言 | 7 |
| 1、Matlab 帮助的使用 | 18 |
| 1.1 help | 18 |
| 1.2 lookfor 指令 | 18 |
| 1.3 超文本格式的帮助用户 | 18 |
| 1.4 demos | 18 |
| 1.5 pdf 帮助文件 | 18 |
| 2、Matlab 数据输入与类型 | 19 |
| 2.1 Matlab 中的数值型数据 | 19 |
| 2.2 Matlab 中的变量 | 20 |
| 2.3 向量及其运算 | 22 |
| 2.4 矩阵及其运算 | 23 |
| 2.5 数组及其运算 | 28 |
| 2.6 元胞（cell）数组（单元型变量） | 30 |
| 2.7 结构体变量 | 31 |
| 3、Matlab 数据处理 | 32 |
| 3.1 Matlab 中的默认数据文件 mat 文件 | 32 |
| 3.2 纯文本文件 | 32 |
| 3.3 Excel 文件 | 34 |
| 3.4 字符串数据 | 34 |
| 3.5 Matlab 与文件相关的操作函数： | 35 |
| 3.6 图像文件 | 38 |
| 3.7 数据的标准化 | 39 |
| 4、Matlab 编程 | 44 |
| 4.1 Matlab 中流程控制语句 | 44 |

| | |
|--|------------|
| 4.2 脚本文件..... | 46 |
| 4.3 函数文件..... | 47 |
| 4.4 函数参数的可调性..... | 50 |
| 4.5 程序调试..... | 51 |
| 4.6 匿名函数与 inline 函数..... | 51 |
| 4.7 伪代码与代码保密处理..... | 52 |
| 4.8 其它编程说明..... | 52 |
| 5、Matlab 绘图命令..... | 55 |
| 5.1 二维绘图命令..... | 55 |
| 5.2 复数绘图..... | 60 |
| 5.3 显函数，符号函数或隐函数的绘图..... | 61 |
| 5.4 三维图形..... | 63 |
| 5.5 特殊的二维/三维图形..... | 72 |
| 5.6 颜色和光照设置命令..... | 76 |
| 5.7 动态可视化图形..... | 78 |
| 5.8 Matlab 的图形句柄..... | 81 |
| 5.9 图形用户接口（Graphics User Interface）..... | 84 |
| 6、Matlab 在数值模拟中的几个应用..... | 88 |
| 6.1 蒙特卡洛方法..... | 88 |
| 6.2 微分方程组的数值模拟..... | 93 |
| 6.3 服从概率分布的随机模拟..... | 95 |
| 6.4 随机型动态系统仿真..... | 98 |
| 7、Matlab 在高等数学中的应用..... | 105 |
| 7.0 Matlab 中的符号表示..... | 105 |
| 7.1 求极限..... | 108 |
| 7.2 求导数..... | 109 |
| 7.3 求一元函数极值..... | 110 |
| 7.4 求积分..... | 110 |
| 7.5 求解微分方程（组）..... | 113 |

| | |
|--------------------------------|------------|
| 7.6 级数求和 | 120 |
| 7.7 Matlab 求解优化问题..... | 121 |
| 7.8 代数方程的求解..... | 127 |
| 7.8.1 代数方程的图解法 | 127 |
| 7.8.2 多项式型方程的准解析解法..... | 128 |
| 7.9 Matlab 求解插值和拟合问题..... | 129 |
| 8、Matlab 在线性代数中的应用..... | 141 |
| 8.1 向量组的线性相关性 | 141 |
| 8.2 齐次线性方程组 | 142 |
| 8.3 非齐次线性方程组 | 142 |
| 8.4 相似矩阵及二次型 | 145 |
| 8.5 线性代数中的其它应用 | 146 |
| 9、Matlab 在概率统计中的应用..... | 149 |
| 9.1 相关概率函数 | 149 |
| 9.2 常见的概率分布密度函数..... | 150 |
| 9.3 一类概率问题的求解..... | 153 |
| 9.4 概率统计作图 | 154 |
| 10、Matlab 图论工具箱..... | 157 |
| 10.1 图的矩阵表示与绘制 | 157 |
| 10.2 几个函数的简要介绍 | 158 |
| 10.3 最短路径问题 | 161 |
| 10.4 最小生成树问题 | 161 |
| 10.5 最大流问题 | 162 |
| 11、评价方法 | 163 |
| 11.1 理想解法 | 163 |
| 11.2 层次分析法 | 167 |
| 11.3 灰色关联分析法..... | 171 |
| 11.4 主成分分析法 | 172 |

| | |
|----------------------------|------------|
| 11.5 秩和比法 (RSR 法) | 174 |
| 12、预测方法 | 178 |
| 12.1 多项式回归分析 | 178 |
| 12.2 非线性回归 | 182 |
| 12.3 GM(1,1)预测模型 | 185 |
| 12.4 预测方法及其适用的范围小结 | 189 |
| 13、智能算法 | 191 |
| 13.1 人工神经网络 | 191 |
| 13.2 Matlab 神经网络工具箱 | 191 |
| 13.3 BP 神经网络的 Matlab 相关函数 | 192 |
| 13.4 RBF 神经网络的 Matlab 相关函数 | 195 |
| 13.5 遗传算法 | 196 |
| 14、Simulink 初步 | 198 |
| 14.1 什么是 Simulink | 198 |
| 14.2 Simulink 仿真模型的组成 | 199 |
| 14.3 Simulink 主要模块介绍 | 204 |
| 14.4 仿真示例 | 208 |
| 15、混沌与分形 | 211 |
| 15.1 混沌 | 211 |
| 15.2 分形 | 214 |
| 参考文献 | 221 |

Matlab 简介

作为和 Mathematica、Maple 并列的三大数学软件。其强项就是其强大的矩阵计算以及仿真能力。要知道 Matlab 的由来就是 Matrix + Laboratory = Matlab，所以这个软件在国内也被称作《矩阵实验室》。每次 MathWorks 发布 Matlab 的同时也会发布仿真工具 Simulink。在欧美很多大公司在将产品投入实际使用之前都会进行仿真试验，他们所主要使用的仿真软件就是 Simulink。Matlab 提供了自己的编译器：全面兼容 C++ 以及 Fortran 两大语言。所以 Matlab 是工程师，科研工作者手上最好的语言，最好的工具和环境。Matlab 已经成为广大科研人员的最值得信赖的助手和朋友！

目前 MATLAB 产品族可以用来进行：

- 数值分析
- 数值和符号计算
- 工程与科学绘图
- 控制系统的设计与方针
- 电路、电子技术、电力技术等
- 数字图像处理
- 数字信号处理
- 通讯系统设计与仿真
- 财务与金融工程...

Simulink 是基于 MATLAB 的框图设计环境，可以用来对各种动态系统进行建模、分析和仿真，它的建模范围广泛，可以针对任何能够用数学来描述的系统进行建模，例如航空航天动力学系统、卫星控制制导系统、通讯系统、船舶及汽车等等，其中了包括连续、离散，条件执行，事件驱动，单速率、多速率和混杂系统等等。Simulink 提供了利用鼠标拖放的方法建立系统框图模型的图形界面，而且 Simulink 还提供了丰富的功能块以及不同的专业模块集合，利用 Simulink 几乎可以做到不书写一行代码完成整个动态系统的建模工作。

与其他语言比较，Matlab 语言的优势：（1）简洁高效；（2）科学运算功能；（3）绘图功能；（4）庞大的工具箱与模块集；（5）强大的动态系统仿真功能。

学好 Matlab 的简单方法：

要带着问题学，活学活用，学用结合，急用先学，立竿见影，在用字上下功夫

0、前言

为什么要学习计算机数学语言

- (1) 并不是所有问题都能够手工推导(并不是所有问题都有解析解);
- (2) 对于有些可以手工推导的问题, 使用计算机求解更加方便;
- (3) 常规编程软件具有很大的局限性;

例 1: 计算 $f(x) = \frac{\sin(x)}{x^2 + 4x + 3}$ 的四阶导数

解:

```
syms x;
```

```
f=sin(x)/(x^2+4*x+3);
```

```
y=diff(f,x,4)
```

例 2: 计算下面的多项式方程的根

$$s^6 + 9s^5 + \frac{135}{4}s^4 + \frac{135}{2}s^3 + \frac{1215}{16}s^2 + \frac{729}{16}s + 64 = 0$$

解: 数值解:

```
p=[1,9,135/4,135/2,1215/16,729/16,64];
```

```
roots(p)
```

解析解:

```
p1=poly2sym(p);
```

```
solve(p1)
```

例 3: 考虑如下 Hilbert 矩阵

$$H = \begin{bmatrix} 1 & 1/2 & 1/3 & \cdots & 1/n \\ 1/2 & 1/3 & 1/4 & \cdots & 1/(n+1) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1/n & 1/(n+1) & 1/(n+2) & \cdots & 1/(2n-1) \end{bmatrix}$$

计算 $n=20$ 时的行列式值

解:

```
H=sym(hilb(20));
```

```
det(H)
```

例 4： 计算 Van der Pol 微分方程

$$y'' + \mu(y^2 - 1)y' + y = 0$$

解：

mu=1000;

f=@(t,x)[x(2);-mu*(x(1)^2-1)*x(2)-x(1)];

[t,x]=ode15s(f,[0,3000],[-1,1]);

plot(t,x)

例 5： 最优化问题

(1) 求解 $f(x) = x^4 - x^2 + x - 1$ 在区间 $[-2,1]$ 上的极小值

解：

Method 1: [X, fval, exitflag, output]=fminbnd ('x^4-x^2+x-1',-2, 1);

Method 2: [x, fval, exitflag, output] =fminsearch ('x^4-x^2+x-1',0);

(2) 求解下面的优化问题

$$\min f(s,t) = s^4 - 4s - 8t + 15, s.t. \begin{cases} 9 - s^2 - t^2 \leq 0 \\ 2s + 3t \leq 2 \\ t - s \leq 5 \end{cases}$$

解：

1) 首先定义目标函数文件

function y=OptimFun(x)

y=x(1)^4-4*x(1)-8*x(2)+15;

end

2) 定义非线性约束条件

function [c ceq]=ConFun(x)

c=9-x(1)^2-x(2)^2;

ceq=[];

end

3)定义线性约束条件

A=[2 3;1 -1]; b=[2;5];

4) 输入求解命令


```
X=fmincon(@OptimFun,[1,2],A,b,[],[],[],[],@ConFun)
```

例 6：积分变换、复变函数、偏微分方程、数据插值与拟合、概率论与数理统计、数值分析等

(1) Matlab 用于线性插值

```
x=0:2:24;  
y=[12,9,9,10,18,24,28,27,25,20,18,15,13];  
x1=0:0.5:24;  
y1=interp1(x,y,x1,'linear');  
plot(x,y,'bo',x1,y1,'r:');
```

(2) 二维插值

```
x=100:100:500;  
y=100:100:400;  
z=[636 697 624 478 450  
    698 712 630 478 420  
    680 674 598 412 400  
    662 626 552 334 310];  
p=100:1:500;  
q=100:1:400;  
q=q';%须为列向量  
z0=interp2(x,y,z,p,q);%分段线性插值  
z1=interp2(x,y,z,p,q,'spline');%三次线条插值  
subplot(2,1,1);  
mesh(p,q,z0);  
title('分段线性插值');  
subplot(2,1,2);  
mesh(p,q,z1);  
title('三次线条插值');  
  
(3) 样条插值  
x=100:100:500;  
y=100:100:400;
```

```

z=[636 697 624 478 450
    698 712 630 478 420
    680 674 598 412 400
    662 626 552 334 310];
p=100:1:500;
q=100:1:400;
q=q';
%三次线条插值
pp=csape({x,y},z');%注意跟 interp2 的区别,有个转置
z0=fnval(pp,{p,q});
mesh(p,q,z0);%注意跟 interp2 的区别,有个转置
title('三次线条插值');

```

(4) 散点插值并获得最大值

```

%散乱节点的二维插值
x=[129 140 103.5 88 185.5 195 105 157.5 107.5 77 81 162 162 117.5];
y=[7.5 141.5 23 147 22.5 137.5 85.5 -6.5 -81 3 56.5 -66.5 84 -33.5];
z=-[4 8 6 8 6 8 8 9 9 8 8 9 4 9];
x0=[75:1:200];
y0=[-85:1:145]';
z0=griddata(x,y,z,x0,y0,'cubic');%保凹凸性 3 次插值
mesh(x0,y0,z0);
%max(z0)返回一个行向量, 向量的第 i 个元素是矩阵 A 的第 i 列上的最大值
%find(A) 寻找矩阵 A 非零元素下标,返回矩阵 A 中非零元素所在位置
%[i,j,v]=find(A)返回矩阵 A 中非零元素所在的行 i,列 j,和元素的值 v(按所在位置先后顺序输出)

```

```

[p,q]=find(z0==max(max(z0)));
zmax=z0(p,q)

```

(5) 多项式拟合

```

x0=[1990 1991 1992 1993 1994 1995 1996];
y0=[70 122 144 152 174 196 202];

```

```
%画出散点图
plot(x0,y0,'ro');
hold on
%用线性拟合
p=polyfit(x0,y0,1);
z0=polyval(p,x0);
plot(x0,z0);
```

(6) 最小二乘优化

```
%拟合形如  $y=a+bx^2$  的函数
%采样点
x=[19 25 31 38 44]';
y=[19 32.3 49 73.3 97.8]';
r=[ones(5,1),x.^2];
ab=lsqlin(r,y)
x0=19:0.1:44;
y0=ab(1)+ab(2)*x0.^2;
plot(x,y,'o',x0,y0,'r')
```

例 7：新的数学分支，如人工神经网络与深度学习、机器学习、蚁群算法、粒子群算法、模糊集与粗糙集等

(1) 使用 BP 神经网络进行数据分类（newff 函数）

```
clear
%p1,p2 是训练数据
p1=[1.24,1.27;1.36,1.74;1.38,1.64;1.38,1.82;1.38,1.90;
1.40,1.70;1.48,1.82;1.54,1.82;1.56,2.08];
p2=[1.14,1.82;1.18,1.96;1.20,1.86;1.26,2.00
1.28,2.00;1.30,1.96];
p=[p1;p2]';
pr=minmax(p)
%goal 是训练数据 p 的标准输出结果
goal=[ones(1,9),zeros(1,6);zeros(1,9),ones(1,6)];
```

```

%plot(p1(:,1),p1(:,2),'h',p2(:,1),p2(:,2),'o')
%创建一个前向反馈后向传播神经网络-即 BP 神经网络
net=newff(pr,[3,2],{'logsig','logsig'});
%设置训练参数
net.trainParam.show = 10;
net.trainParam.lr = 0.05;
net.trainParam.goal = 1e-10;
net.trainParam.epochs = 50000;
%训练网络
net = train(net,p,goal);
x=[1.24 1.80;1.28 1.84;1.40 2.04]';
%测试训练结果
y1=sim(net,p1')
y2=sim(net,p2')
y=sim(net,x)

```

(2) 使用 BP 神经网络进行数据分类 (feedforwardnet 函数)

```

clear
%p1,p2 是训练数据
p1=[1.24,1.27;1.36,1.74;1.38,1.64;1.38,1.82;1.38,1.90;
1.40,1.70;1.48,1.82;1.54,1.82;1.56,2.08];
p2=[1.14,1.82;1.18,1.96;1.20,1.86;1.26,2.00
1.28,2.00;1.30,1.96];
p=[p1;p2]';
%goal 是训练数据 p 的标准输出结果
goal=[ones(1,9),zeros(1,6);zeros(1,9),ones(1,6)];
%创建网络
net=feedforwardnet(8);
%训练网络
net = train(net,p,goal);
view(net);

```

```
x=[1.24 1.80;1.28 1.84;1.40 2.04]';
```

```
%测试训练结果
```

```
y1=net(p1')
```

```
y2=net(p2')
```

```
y=net(x)
```

（3）使用 BP 神经网络做预测

```
x=[54167
```

```
55196
```

```
56300
```

```
57482
```

```
58796
```

```
60266
```

```
61465
```

```
62828
```

```
64653
```

```
65994
```

```
67207
```

```
66207
```

```
65859
```

```
67295
```

```
69172
```

```
70499
```

```
72538
```

```
74542
```

```
76368
```

```
78534
```

```
80671
```

```
82992
```

```
85229
```

```
87177
```

89211
90859
92420
93717
94974
96259
97542
98705
100072
101654
103008
104357
105851
107507
109300
111026
112704
114333
115823
117171
118517
119850
121121
122389
123626
124761
125786
126743
127627
128453

```

129227
129988
130756
131448
132129
132802
134480
135030
135770
136460
137510]';
% 该脚本用来做 NAR 神经网络预测（非线性自回归 NN）
lag=3;    % 自回归阶数
iinput=x;    % x 为原始序列（行向量）
n=length(iinput);
%准备输入和输出数据
inputs=zeros(lag,n-lag);
for i=1:n-lag
    inputs(:,i)=iinput(i:i+lag-1)';
end
targets=x(lag+1:end);
%创建网络
hiddenLayerSize = 10; %隐藏层神经元个数
net = fitnet(hiddenLayerSize);
% 避免过拟合，划分训练，测试和验证数据的比例
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;
%训练网络
[net,tr] = train(net,inputs,targets);

```

```

%% 根据图表判断拟合好坏

yn=net(inputs);
errors=targets-yn;
figure, ploterrcorr(errors) %绘制误差的自相关情况 (20lags)
figure, parcorr(errors) %绘制偏相关情况
%[h,pValue,stat,cValue]= lbqtest(errors) %Ljung-Box Q 检验 (20lags)
figure, plotresponse(con2seq(targets),con2seq(yn)) %看预测的趋势与原趋势
figure, ploterrhist(errors) %误差直方图
figure, plotperform(tr) %误差下降线

%% 下面预测往后预测几个时间段
fn=7; %预测步数为 fn
f_in=input(n-lag+1:end)';
f_out=zeros(1,fn); %预测输出
% 多步预测时，用下面的循环将网络输出重新输入
for i=1:fn
    f_out(i)=net(f_in);
    f_in=[f_in(2:end);f_out(i)];
end
% 画出预测图
figure, plot(1949:2013,iinput,'b',2013:2020,[iinput(end),f_out], 'r')

例 8： 电路、电子技术、自动控制、电机与转动等
demo simulink 'simulink control design' %调出 Matlab 自带的 Demo

例 9： 计算 Fibonacci 数列

```

Matlab 语言：

```

a=[1 1];
for i=3:100
    a(i)=a(i-1)+a(i-2);
end
a

```


例 10. Matlab 用于回归分析

(1) 首先分析数据是否具有线性关系

```
x1=[2.23,2.57,3.87,3.10,3.39,2.83,3.02,2.14,3.04,3.26,3.39,2.35,2.76,3.90,3.16];
```

```
x2=[9.66,8.94,4.40,6.64,4.91,8.52,8.04,9.05,7.71,5.11,5.05,8.51,6.59,4.90,6.96];
```

```
y=[12.37,12.66,12.00,11.93,11.06,13.03,13.13,11.44,12.86,10.84,11.20,11.56,10.83,12.63,12.46];
```

```
corrcoef(x1,y)
```

```
corrcoef(x2,y)
```

```
plot3(x1,x2,y,'*')
```

(2) 通过旋转图像可以观察到数据大多在一个平面上，因此可能满足关系 $y=a_0+a_1x_1+a_2x_2$ 使用 Matlab 中的回归函数

```
x1=[2.23,2.57,3.87,3.10,3.39,2.83,3.02,2.14,3.04,3.26,3.39,2.35,2.76,3.90,3.16]';
```

```
x2=[9.66,8.94,4.40,6.64,4.91,8.52,8.04,9.05,7.71,5.11,5.05,8.51,6.59,4.90,6.96]';
```

```
y=[12.37,12.66,12.00,11.93,11.06,13.03,13.13,11.44,12.86,10.84,11.20,11.56,10.83,12.63,12.46]';
```

```
e=ones(15,1);
```

```
x=[e,x1,x2];
```

```
[b,bint,r,rint,stats]=regress(y,x,0.05)
```

```
rcoplot(r,rint) %表示画出残差与残差区间的杠杆图
```

(3) 根据结果进行分析是否合理

没有发现高杠杆点，也就是说，数据中没有强影响点、异常观测点。

1、Matlab 帮助的使用

1.1 help

help ↵ %帮助总览

help elfun ↵ %关于基本数学函数的帮助信息

help exp ↵ %指数函数 exp 的详细信息

1.2 lookfor 指令

当要查找具有某种功能但又不知道准确名字的指令时,help 的能力就不够了,lookfor 可以根据用户提供的完整或不完整的关键词,去搜索出一组与之相关的指令。

lookfor integral ↵ %查找有关积分的指令

lookfor fourier ↵ %查找能进行傅里叶变换的指令

1.3 超文本格式的帮助文件

在 Matlab 中,关于一个函数的帮助信息可以用 doc 命令以超文本 (HTML) 的方式给出,如

doc ↵

doc ode45 ↵

doc eig ↵ %eig 求矩阵的特征值和特征向量

1.4 demos

Matlab 提供的用于各个领域的示例。

1.5 pdf 帮助文件

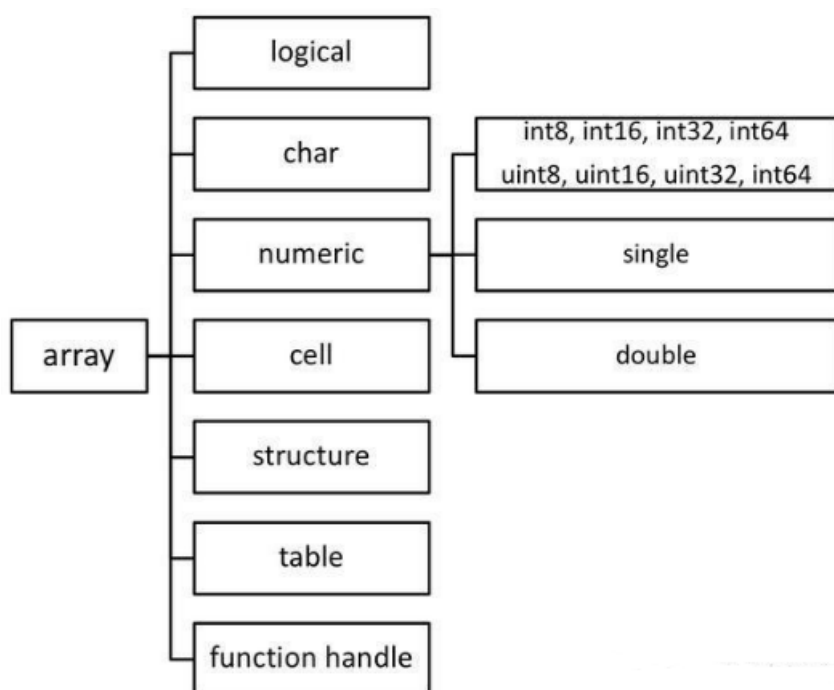
可从 MathWorks 网站上下载有关的 pdf 帮助文件。

网站地址: <http://www.mathworks.com/>

2、Matlab 数据输入与类型

2.1 Matlab 中的数值型数据

Matlab 中最常用的是数值量为双精度浮点型，占 8 个字节（64 位），遵从 IEEE 记数法，有 11 个指数位，52 位尾数和一个符号位，在 Matlab 中表示为 `double()`；考虑到一些特殊的应用，如图像处理，Matlab 也引入了无符号的 8 位整型数据 `uint8`，其值域为 0~255，这样可以大大节省存储空间，提高处理速度。类似的，还有一些其他数据类型，如 `int8`,`int16`,`int32`,`uint16`,`uint32` 等类型，用于不同的处理内容。



| 类型 | 子类型 | 符号 | 位数 | 用法 |
|-------|--------|--------|----|-----------------|
| 整型 | 有符号的整形 | int8 | 8 | a=int32(12) |
| | | int16 | 16 | |
| | | int32 | 32 | |
| | | int64 | 64 | |
| | 无符号的整形 | uint8 | 8 | a=uint32(12) |
| | | uint16 | 16 | |
| | | uint32 | 32 | |
| | | uint64 | 64 | |
| 单精度浮点 | | single | 32 | a=single(12.34) |
| 双精度浮点 | | double | 64 | a=12.23 |

注 1: 在定义指定数据类型预定义了初始空间的变量时, 可以使用下面两种方法获得相同的数据, 例如, `I1=uint8(zeros(100));I2=zeros(100,'uint8')`, 后一种方法的效率更高。

注 2: 复数数据的输入: `c=1+2i`, 与 `c=1+2*i` 等价。

2.2 Matlab 中的变量

MATLAB 程序中的基本数据单元称为阵列(Array), 是一个分为行与列的数据集合。变量被看作是只有一行一列的阵列。MATLAB 语言不需要对变量进行事先声明, 也不需要指定变量类型, 它会自动根据所赋予变量的值或对变量所进行的操作来确定变量的类型。其命名规则为:

- (1) 变量名的**大小写是敏感**的。
- (2) 变量的**第一个字符必须为英文字母**, 而且不能超过 31 个字符。
- (3) 变量名可以包含**下划线、数字**, 但不能为空格符、标点。
- (4) 命名变量时可以取一个容易记忆并且能表达出其含义的名称, 如汇率, 可以定义为 `exchange_rate`。

对于变量作用域, 默认情况是**局部变量**, 使用 `global` 定义**全局变量**, 而且全局变量常用**大写的英文字母**表示, 全局变量再定义之后一般需要提供提供一个初始值, 在其他脚本文件中使用全局变量时需要使用 `global` 声明, **`global X;`**。初始化的时候声明一次, 用的时候再声明一次【在一个内存空间里声明 `global`, 在另一个内存空间里使用这个 `global` 的时候需要再次声明 `global`, 当然, 各内存空间里声明一次就可以了】。

Matlab 中也存在**静态变量 (persistent)**, 在函数中声明的变量, 当函数调用完之后就会释放。如果想保留这个变量的值 (供该函数下一次调用), 可以把这个变量声明为静态变量。静态变量不能在声明的时候赋值, 而且只能在 `function` 里声明, 且只有这个 `function` 才能认识它。声明方法: **`persistent x;`**声明后需要初始化。

MATLAB 预定义的变量如下表所示:

| 预设变量名 | 含义 |
|------------|----------------------------|
| ans | 预设的计算结果的变量名 |
| eps | MATLAB 定义的正的极小值=2.2204e-16 |
| pi | 内建的 π 值 |

| | |
|----------------|----------------------|
| inf | ∞ 值，无限大 |
| NaN | 无法定义一个数目 |
| i 或 j | 虚数单位 $i=j=\sqrt{-1}$ |
| nargin | 函数输入参数个数 |
| nargout | 函数输出参数个数 |
| realmax | 最大的正实数 2^{1023} |
| realmin | 最小的正实数 2^{-1022} |
| flops | 浮点运算次数 |

注 1: 在定义变量时要尽量与避免与这些名字相同，以免改变它们的值，如果已经改变，可以通过 **clear 变量名** 来恢复它的初始值，也可以通过重新启动 MATLAB 恢复这些值。

注 2: 数字的输入输出格式。所有数据均按 IEEE 浮点标准的长型格式存储。输入格式沿用了 C 语言的风格和规则；输出格式使用 **format 数据格式** 命令控制，只影响在屏幕上的显示结果，不影响内部的存储和运算。

注 3: 对于符号型数值可以通过变精度算法函数 **vpa()** 以指定的精度显示出来。调用方法为 **vpa(A)**，或者 **vpa(A,n)**，例如 **vpa(pi,200)**

| Type | Result |
|-----------|---|
| short | Scaled fixed point format, with 4 digits after the decimal point. For example, 3.1416. |
| long | Scaled fixed point format with 14 to 15 digits after the decimal point for double; and 7 digits after the decimal point for single. For example, 3.141592653589793. |
| short e | Floating point format, with 4 digits after the decimal point. For example, 3.1416e+000. |
| long e | Floating point format, with 14 to 15 digits after the decimal point for double; and 7 digits after the decimal point for single. For example, 3.141592653589793e+000. |
| short g | Best of fixed or floating point, with 4 digits after the decimal point. For example, 3.1416. |
| long g | Best of fixed or floating point, with 14 to 15 digits after the decimal point for double; and 7 digits after the decimal point for single. For example, 3.14159265358979. |
| short eng | Engineering format that has 4 digits after the decimal point, and a power that is a multiple of three. For example, 3.1416e+000. |
| long eng | Engineering format that has exactly 16 significant digits and a power that is a multiple of three. For example, 3.14159265358979e+000. |

2.3 向量及其运算

1、向量的生成

①命令窗口直接输入，使用[]，元素之间用**空格、逗号或者分号**隔开。

②使用冒号表达式，基本形式为 **x=x0:step:xn**，其中 xn 为尾元素数值限，而不一定是尾元素的值。当 step=1 时可省略步长。

③生成线性等分向量，使用 **linspace** 函数。Y=linspace(x1,x2,n) （在 x1 和 x2 之间分成 n 等份）

④生成对数等分向量，使用 **logspace** 函数。Y=logspace(x1,x2,n) （ 10^{x1} 和 10^{x2} 之间分成 n 等份）

2、向量的基本运算

数加（减）、数乘、点积(dot 函数)、叉积(cross 函数)、混合积 dot(a,cross(b,c))

示例：

（1）dot（a，b） 返回向量 a 和 b 的数量点积。a 和 b 必须同维。当 a 和 b 都为列向量时，dot（a，b） 同于 a.*b。

`dot(a, b, dim)` 返回 `a` 和 `b` 在维数为 `dim` 的点积。`dim=1` 按列点积，`dim=2` 对应按行点积，`dim>=3` 时逐个元素点积。

```
A=[1 2 3;4 5 6;7 8 9];
```

```
B=[7 8 9;1 2 3;4 5 6];
```

```
dot(A,B,1), dot(A,B,2), dot(A,B,3)
```

(2) `c = cross(a, b, dim)` 当 `a` 和 `b` 为 `n` 维数组时，则返回 `a` 和 `b` 的 `dim` 维向量的叉积。`a` 和 `b` 必须有相同的维数。且 `size(a, dim)` 和 `size(b, dim)` 必须为 **3**。`dim` 的含义和 `dot` 中的 `dim` 一样。

```
A=[1 2 3;4 5 6;7 8 9];
```

```
B=[7 8 9;1 2 3;4 5 6];
```

```
C1=cross(A,B,1), C2=cross(A,B,2)
```

注 1: 根据叉积的计算规则，当 `a,b` 是向量时对应的维数必须是 3 维的。

注 2: 当 `a` 和 `b` 是矩阵时，叉积所得结果实质是由多个叉积结果组合而成。如 `C1(:,1)` 由 `A(:,1)`和 `B(:,1)`的叉积计算而来。

2.4 矩阵及其运算

1、简单矩阵的输入

(1) 要直接输入矩阵时，矩阵一行中的元素用空格或逗号分隔；矩阵行与行之间用分号“`;`”隔离，整个矩阵放在方括号“`[]`”里。

```
A=[1,2,3;4,5,6;7,8,9] ↵
```

说明：指令执行后，矩阵 `A` 被保存在 `Matlab` 的工作空间中，以备后用。如果用户不用 `clear` 指令清除它，或对它进行重新赋值，那么该矩阵会一直保存在工作空间中，直到本次指令窗关闭为止。

(2) 矩阵的分行输入，此时回车键作为分行标志，

```
A=[1,2,3
```

```
4,5,6
```

```
7,8,9];
```

(3) 使用 `M` 文件创建大矩阵，当矩阵维数非常大时，可以创建 `m` 文件，在 `m` 文件中输入数据或者导出数据文件。

2、矩阵的基本运算

①矩阵的四则运算。其中乘法运算要注意相乘的双方有**相邻公共维**，除法分为**左除**“\”(A\B=inv(A)*B)和**右除**“/”(A/B=A*inv(B)) (需要计算逆矩阵)

②矩阵的逆运算：**inv** 函数。

③矩阵的幂运算：**^**，转置运算：**'**。

④矩阵的指数运算。**exp**(返回每个元素的指数值)，**expm**([V,D]=EIG(X)，**expm**(X)=V*diag(exp(diag(D)))/V)，**expm1**(exp(x)-1)

⑤矩阵的对数运算。**logm**，A=logm(B)/log(10)，B=10^A

⑥矩阵的特征值函数。**eig** 和 **eigs**(适合于大型稀疏方阵)

⑦矩阵的奇异值函数。**svd** ([U,S,V]=SVD(X)，X=U*S*V') 和 **svds**

⑧矩阵的条件数函数。**cond** (矩阵 A 的条件数等于 A 的范数与 A 的逆的范数的乘积，c=cond(A,p)等价于 norm(A,p)*norm(inv(A),p))，condest(1 范数的条件数的估计值)，**rcond**

⑨特征值的条件数函数。**condeig** ([V,D,s]=condeig(A) 等价于[V,D]=eig(A); s=condeig(A);)

⑩范数函数。**norm** (1-范数：即列范数，矩阵的各列绝对值之和的最大值；2-范数：所有元素的平方和开根号(默认)；无穷范数：即行范数，矩阵各行的绝对值之和的最大值)，**normest** (矩阵的 2 范数的估计值)

其他还有秩函数 **rank**，迹函数 **trace**，零空间函数 **null** (又称为核空间，X=null(A)，则 A*X=0，X'*X=I)，正交空间函数 **orth**(B=orth(A)，则 B'*B=eye(rank(A)))，伪逆函数 **pinv** 等。

3、特殊向量和特殊矩阵

(1) 特殊向量

t=[0:0.1:10] %产生从 0 到 10 的行向量，元素之间间隔为 0.1

t=linspace(n1,n2,n)

%产生 n1 和 n2 之间线性均匀分布的 n 个数 (缺省 n 时,产生 100 个数)

t=logspace(n1,n2,n) (缺省 n 时,产生 50 个数)

%在和之间按照对数距离等间距产生 n 个数。

(2) 特殊矩阵

i)单位矩阵

eye(m),

eye(m,n) 可得到一个可允许的最大单位矩阵而其余处补 0,

eye(size(a)) 可以得到与矩阵 a 同样大小的单位矩阵。

ii)所有元素为 1 的矩阵

ones(n), **ones(size(a))**, **ones(m, n)**。

iii) 所有元素为 0 的矩阵

zeros(n), **zeros(m,n)**。

iv) **空矩阵**是一个特殊矩阵, 这在线性代数中是不存在的。例如

q=[]

矩阵 q 在工作空间之中, 但它的大小为零。通过空矩阵的办法可以删除矩阵的行与列。例如

a(:,3)=[]

表示删除矩阵 a 的第 3 列。

v) 随机数矩阵

rand(m,n) 产生 $m \times n$ 矩阵, 其中的元素是服从[0,1]上均匀分布的随机数。

randint(m,n,[min,max]) 产生 $m \times n$ 矩阵, 其中的元素是[min,max]上的随机整数。

normrnd(mu,sigma,m,n)产生 $m \times n$ 矩阵, 其中的元素是服从均值为 mu, 标准差为 sigma 的正态分布的随机数。

exprnd(mu,m,n) 产生 $m \times n$ 矩阵, 其中的元素是服从均值为 mu 的指数分布的随机数。

poissrnd(mu,m,n) 产生 $m \times n$ 矩阵, 其中的元素是服从均值为 mu 的泊松 (Poisson) 分布的随机数。

unifrnd(a,b,m,n) 产生 $m \times n$ 矩阵, 其中的元素是服从区间[a,b]上均匀分布的随机数。

r = mvnrnd(MU,SIGMA,cases) 产生 cases 对均值向量为 MU, 协方差阵为 SIGMA 的多维正态分布的随机数。

vi) 随机置换

randperm(n)产生 1 到 n 的一个随机全排列。

perms([1:n])产生 1 到 n 的所有全排列。

vii)稀疏矩阵

稀疏矩阵是指矩阵中零元素很多, 非零元素很少的矩阵。对于稀疏矩阵, 只要存放非零元素的行标、列标、非零元素的值即可, 可以按如下方式存储

（非零元素的行地址，非零元素的列地址），非零元素的值。

在 Matlab 中无向图和有向图邻接矩阵的使用上有很大差异。

对于有向图，只要写出邻接矩阵，直接使用 Matlab 的命令 **sparse** 命令，就可以把邻接矩阵转化为稀疏矩阵的表示方式。

对于无向图，由于邻接矩阵是对称阵，Matlab 中只需使用邻接矩阵的下三角元素，即 Matlab 只存储邻接矩阵下三角元素中的非零元素。

稀疏矩阵只是一种存储格式。Matlab 中，普通矩阵使用 **sparse** 命令变成稀疏矩阵，稀疏矩阵使用 **full** 命令变成普通矩阵。

示例：

```
a=zeros(5);
```

```
a(1,[2,4])=[3,4];
```

```
a(3,[2:4])=[1 3 8];
```

```
a(5,[1,5])=[6,7]
```

```
b=sparse(a) % 普通矩阵转化成稀疏矩阵
```

```
c=full(b) % 稀疏矩阵转化成普通矩阵
```

其他一些特殊矩阵如下表所示：

| 函数 | 功能 | 函数 | 功能 |
|----------|--------------|-----------|---------------------|
| compan | 伴随阵 | magic | 魔方阵 |
| gallery | Higham 测试阵 | rosser | 经典对称特征值测试阵 |
| hadamard | Hardamard 矩阵 | toeplitz | Toeplitz 矩阵 |
| hankel | Hankel 矩阵 | pascal | Pascal 矩阵 |
| hilb | Hilbert 矩阵 | vander | 范德蒙矩阵 |
| invhilb | 反 Hilbert 矩阵 | wilkinson | Wilkinson's 特征值测试矩阵 |

4、矩阵的特殊操作

①变维。 有两种方法，使用冒号（**:**）和使用函数 **reshape**

使用 “**:**”表达式对两个矩阵进行变维操作，需要预先定义两个矩阵的维数（例如

A(:,3)=[]); reshape 有两种形式，分别为 reshape(X,M,N)和 reshape(X,M,N,P...)

②变向 主要函数如下表所示：

| 函数 | 功能 | 函数 | 功能 |
|--------|-------------|------|----------|
| fipplr | 矩阵左右翻转 | diag | 产生或提取对角阵 |
| fipud | 矩阵上下翻转 | tril | 产生下三角 |
| fipdim | 矩阵特定维翻转 | triu | 产生上三角 |
| Rot90 | 矩阵反时针 90 翻转 | | |

③矩阵的抽取

提取子矩阵的具体方法： $B=A(v1,v2)$,其中 $v1$ 向量表示子矩阵要保留的行号构成的向量， $v2$ 表示要保留的列号构成的向量，这样就可以从 A 中提取有关的行和列，即可构成子矩阵 B 。

对角线元素抽取函数 **diag(X,k)/diag(v,k)**，抽取矩阵 X 的第 k 条对角线的元素向量/使得向量 v 为所得矩阵的第 k 条对角线元素。

上三角元素抽取 **tril(X,k)**和下三角元素抽取 **triu(X,k)**

④**扩展** 两种方法：利用对矩阵标示块的赋值命令 $X(m1:m2,n1:n2)=a$ 生成大矩阵，其中 $m2-m1+1$ 必须等于 a 的行维数， $n2-n1+1$ 必须等于 a 的列维数，生成 $m2 \times n2$ 维的矩阵 X ；利用小矩阵组合生成大矩阵，要严格注意矩阵大小的匹配。（示例： $X(1:10,1:10)=zeros(10,10)$ ， $LX=[X,X;X,X]$ ）

Matlab 中冒号 (:) 的使用方法小结：

- (1) 用于生成向量， $a:b$ ，一般要求 $a < b$ ，否则生成空矩阵。
- (2) 如果 $b-a$ 不是整数时，则向量的最后一位数为 $n+a$ ，其中 $n=\text{fix}(b-a)$ (向零取整)
- (3) $a:c:b$ 表示 $[a, a+c, \dots, a+n*c]$ ，其中 $n=\text{fix}((b-a)/c)$ ，当 $c > 0$ 且 $a > b$ 或者 $c < 0$ 且 $a < b$ 时出现空矩阵。
- (4) $A(:)$ 以一列的方式显示 A 中所有的元素
- (5) $b=A(i,:)$ 表示将 A 中第 i 行存入 b 中
- (6) $b=A(:,j)$ 表示将 A 中第 j 列存入 b 中
- (7) $b=A(j:k)$ 表示将 A 中第 j 到第 k 个元素存入 b 中（Matlab 中矩阵**按列**存储）
- (8) $b=A(:,c:d)$ 表示将 A 中第 c 列到第 d 列存入 b 中，要求 c, d 不能超过 A 的列数。

(9) 当矩阵很大时，不知道矩阵的维数，可以使用 **end** 作为矩阵的最后一行或者一列或者最后一个元素。例如 `b=A(1:2,2:end)` 获取矩阵 A 右上角的元素。

(10) `A=[1 2 3 4;5 6 7 8;9 10 11 12]; A(1:2,[1 4])=[20 21;22 23]`，对矩阵中的某几个元素重新赋值；

(11) `A([1,3],2) A([2 2],[3 3])` 示获取对应的元素然后组成新的矩阵。

2.5 数组及其运算

数组与矩阵在形式上完全一致，只是运算与矩阵不同。同型矩阵之间的运算通常称为**数组运算**。（矩阵的数组运算）

1、基本数组运算

①四则运算。数组的乘除法是指两个同维数组间对应元素之间的乘除法，运算符为`.*`，`./`和`.\`。数组与常数之间的运算可以加`.*`，也可以不加。

②幂运算。`.^`对每个数组元素的幂运算。

③指数运算 `exp`，对数运算 `log` 和开方运算 `sqrt`。

2、数组/矩阵函数运算

只要把运算的数组带入到函数中就可以了，通用形式为 `funname(A)`

3、数组/矩阵的逻辑运算和关系运算

| 指令 | 含义 | 函数名 |
|-----------|------|-----|
| < | 小于 | lt |
| <= | 小于等于 | le |
| > | 大于 | gt |
| >= | 大于等于 | ge |
| == | 等于 | eq |
| ~= | 不等于 | ne |
| & | 逻辑 与 | and |
| | 逻辑 或 | or |
| ~ | 逻辑 非 | not |

| 指令 | 含义 | 指令 | 含义 |
|-----------|-------------------|-----------|----------------------|
| xor | 不相同就取 1，否则取 0 | isequal | 相等取 1，否则取 0 |
| any | 只要有非 0 就取 1，否则取 0 | ismember | 两个矩阵是属于关系取 1，否则取 0 |
| all | 全为 1 取 1，否则为 0 | isempty | 矩阵为空取 1，否则取 0 |
| isnan | 为数 NaN 取 1，否则为 0 | isletter | 是字母取 1，否则取 0（可以是字符串） |
| isinf | 为数 inf 取 1，否则为 0 | isstudent | 学生版取 1 |
| isfinite | 有限大小元素取 1，否则为 0 | isprime | 质数取 1，否则取 0 |
| ischar | 是字符串取 1，否则为 0 | isreal | 实数取 1，否则取 0 |
| find | 寻找非零元素坐标 | isspace | 空格位置取 1，否则取 0 |
| isnumeric | 判断数值矩阵 | islogical | 判断逻辑数组 |

函数示例：

(1) 统计字符串中 str='ab13c34df89hk'字母个数和位置

result=**isletter**(str); count=**sum**(result,2); pos=**find**(result);

(2) 求 1 到 100 中的质数。

zs=find(**isprime**(1:100));

(3) 找出矩阵中元素大于等于 5 的下标。

A=[1 2 3; 4 5 6; 7 8 0];

[i,j]=find(A>=5);

(5)判断矩阵的某列元素全大于或等于 5 时，可以使用 a1=all(A>=5),判断某列中含有大于等于 5 的元素，使用 a=any(A>=5);判断矩阵中是否运算均大于等于 5，可以使用 all(A(:)>5)。

4、多维数组的表示与存储

(1) 多维数组的表示方法：c(:,,1)=[1 2 3;4 5 6]; c(:,,2)=[7 8 9;10 11 12];

可以通过 whos 查看数组结构，2*3*2；

(2) 多维数组的存储：Matlab 中的元素是按列存储的，对于上面的多维数组，其存储的顺序是(1,1,1)，(2,1,1)，(1,2,1)，(2,2,1)，(1,3,1)，(2,3,1)，(1,1,2)，(2,1,2)，(1,2,2)，(2,2,2)，(1,3,2)，(2,3,2)

2.6 元胞（cell）数组（单元型变量）

元胞数组是一种可以容纳多种变量类型的数据结构，功能非常强大。

（1）创建方法：

- 直接创建：C={A,B,D}，这里使用大括号{}；
- 使用 cell 类型的变量创建 cell，C={C1,C2}，C1,C2 都是 cell 类型；
- 使用 cell 函数，C=cell(2,3)创建一个两行三列的 cell 矩阵；

（2）访问方法

- X=C(index)返回 cell 类型；
- X=C{index}返回 cell 中具体的内容
- A{1,1}(2,3:end)访问的是 A 中第一个 cell 里面内容的第 2 行的第三个以后的元素。
- A{1,1}.name 访问的是 A 中的第一行第三列的 name 值

cellplot(C)可以显示元胞数组的图形结构。

（3）删除方法

- cell(1)=[]删除第一个 cell，cell{1}=[]用来将第一个 cell 内容置空；
- cell(:)=[]删除全部的 cell

（4）元胞数组的意义

元胞数组是很多 Matlab 内置函数中参数的数据结构，尤其是在图形用户接口的操作中。例如 Matlab 的输入参数 varargin 就是一个 cell 结构，用来判断输入参数的个数，实现函数的多种重载形式。

示例：

（1）使用 cellstr 函数可以从一个 2D 字符串数组构造一个字符串 cell 数组

```
data=['Line1          '; 'Additional Line']
```

```
c=cellstr(data)
```

```
newdata=char(c)转化为标准的字符数组
```

（2）**varargin** 的用法示例

```
function test1(varargin)
```

```
disp(['There are ' int2str(nargin) 'arguments.']);
```

```
disp('The input arguments are: ');
```

```
disp(varargin);
```

end

2.7 结构体变量

结构型变量类似 C 语言中的结构体，它和单元型变量的区别在于结构型变量是以指针方式来传递数据，并且每一个元素都有一个独立的名字。两种定义方式，直接赋值定义和由函数 struct 定义。

```
student.name='John Doe';  
student.addr1='123 Main Street';  
student.city='Anytown';  
student.zip='255049';
```

如果有多个元素，需要使用下标，如 student(2).name='Kathy'

使用 struct 函数创建结构变量，str_array=struct('field1',val1,'field2',val2,...)

添加新的字段，student(1).exams=[90 80 100]

移除字段：strut2=**rmfield**(structname,'exams')

getfield 函数与 setfield 函数的用法：

f=**getfield**(array,{array_index},'field',{field_index});

等价于 f=array(array_index).field(field_index);

f=**setfield**(array,{array_index},'field',{field_index},value);

等价于：f=array(array_index).field(field_index)=value;

3、Matlab 数据处理

3.1 Matlab 中的默认数据文件 mat 文件

1、将工作区变量保存到文本文件中

save filename var1 var2 var3... -mat 默认保存的文件扩展名为.mat，如果保存的数据需要跨平台处理，需要采用如下格式：

save filename.dat var1 var2 ... -ascii

另外：

save filename var1 var2 ... -append 附加到已经存在的 Mat 文件中

示例 1：把 Matlab 工作空间中的数据矩阵 a，b，c 保存到数据文件 data1.mat 中。

save data1 a b c

注：Matlab 中的默认数据文件 mat 文件可以省略后缀名。

2、将文本文件中的数据导入工作区

load filename

示例 2：把示例 1 生成的 data1.mat 中的所有数据加载到 Matlab 工作空间中。

load data1

3.2 纯文本文件

可以把 word 文档中整行整列的数据粘贴到纯文本文件，然后调入到 Matlab 工作空间中。

例 38 把纯文本文件 data2.txt 加载到工作空间。

a=load('data2.txt');

或者是

a=textread('data2.txt');

如果文本文件中有多种数据，则需要使用如下格式：

[a,b,c,...]=textread(filename,format,n) filename 是要打开的文件名，format 指定每一列的数据类型，n 是要读取的行数，若省略 n，则读到文件末尾。

示例 1：[name,type,x,y,answer]=textread('t.txt','%s %s %f %n %s',2)，注意格式读取格

式与文件中的数据格式要保持一致。

示例 2： 读文件 test_input.dat

```
[first,last,blood,gpa,age,answer]=textread('test_input.dat','%s %s %s %f %d %s')
```

通过使用%*s 可以忽略对应的列数据。

示例 3： 假设文件 test.dat 中的数据如下：

James Jones O+ 3.51 22 Yes

Sally Smith A+ 3.28 23 No

读取时：

```
[first,last,blood,gpa,age,answer]=textread('test.dat','%s %s %s %f %d %s')
```

如果文本文件中有分隔符，则可以使用带有分隔符的文件读取函数 **dlmread**，几种调用方法如下所示：

(1)**M=dlmread(filename)**，使用默认的分隔符**单引号**将矩阵读入到 **M** 中

(2)**M=dlmread(filename,delimiter)** 指定分隔符

(3)**M=dlmread(filename,delimiter,R,C)**，读取指定的行与列

对应的存储方法是 **dlmwrite** 函数，调用方法

(1)**dlmwrite(filename,M)** 使用默认的分隔符逗号将 **M** 写入文件中，从第一行第一列开始写入；

(2)**dlmwrite(filename,M,delimiter)**；指定分隔符写入数据；

(3)**dlmwrite(filename,M, delimiter,R,C)**在指定的行与列开始向文件中写入数据；

示例 4： 使用 **dlmwrite** 命令把矩阵 **b** 保存到纯文本文件 **data3.txt**。

```
dlmwrite('data3.txt',b)
```

使用 **fprintf** 函数写文本文件，需要使用 **fopen** 函数将文件打开。格式为：

```
fopen(filename,format);
```

```
fprintf(id,'format',arg1,agr2,...);
```

示例 5： 生成服从标准正态分布随机数的矩阵，然后用 **fprintf** 命令保存到纯文本文件 **data4.txt**。

解 **clc, clear**

```
fid=fopen('data4.txt','w');
```

```
a=normrnd(0,1,100,200);
```

```
fprintf(fid,'%f\n',a);
```

```
fclose(fid);
```

注：对于高维矩阵，用 **dlmwrite** 构造的纯文本文件，Lingo 软件不识别；为了 Lingo 软件识别，纯文本文件必须用 **fprintf** 构造，而且数据之间的分割符为“\n”。

示例 6：

```
M = gallery('integerdata', 100, [5 8], 0);
dlmwrite('myfile.txt', M, 'delimiter', '\t')
dlmread('myfile.txt')
dlmread('myfile.txt', '\t', 2, 3)
dlmread('myfile.txt', '\t', 'C1..G4')
```

3.3 Excel 文件

xlswrite(filename,A,sheet,range) 将阵列 A 写入 Excel 文件 filename 中 sheet 表格的 range 指示的单元格内。

示例 1： 把一个矩阵写到 Excel 文件 data5.xls 表单 Sheet2 中 B2 开始的域中。

```
a=rand(5,10);
```

```
xlswrite('data5.xls',a,'Sheet2','B2') %默认的表格为 Sheet1
```

[num,txt,row] = xlsread(filename,sheet,range)从 Excel 文件 filename 中的 sheet 表格中读取 range 单元格内的数据返回。

示例 2： 把示例 1 中生成的 Excel 文件 data5.xls 中表单 Sheet2 的域“C3: F6”中的数据赋给 b。

```
b=xlsread('data5.xls','Sheet2','C3:F6')
```

示例 3：

```
values = {1, 2, 3 ; 4, 5, 'x' ; 7, 8, 9};
headers = {'First', 'Second', 'Third'};
xlswrite('myExample.xls', [headers; values]);
A = xlsread('myExample.xls')
subsetA = xlsread('myExample.xls', 1, 'B2:C3')
columnB = xlsread('myExample.xls', 'B:B')
[ndata, text, alldata] = xlsread('myExample.xls')
```

3.4 字符串数据

例 43 统计下列五行字符串中字符 a、c、g、t 出现的频数。

- 1.aggcacggaaaaacgggaataacggaggaggacttggcacggcattacacggagg
- 2.cggaggacaaacgggatggcggtattggaggtggcgggactgttcgggga
- 3.gggacgggatacggattctggccacggacgggaaaggaggacacggcggacataca

4.atggataacggaaacaaccagacaaacttcggtagaaatacagaagctta

5.cggctggcggacaacggactggcggattccaaaaacggaggaggcggacggaggc

解 把上述五行复制到一个纯文本数据文件 shuju.txt 中，编写如下程序

```
clc
fid=fopen('shuju.txt','r'); %以只读的形式打开文件
i=1;
while (~feof(fid)) %判断是否到文件末尾
data=fgetl(fid);
a=length(find(data==97));
b=length(find(data==99));
c=length(find(data==103));
d=length(find(data==116));
e=length(find(data>=97&data<=122));
f(i,:)=a b c d e a+b+c+d;
i=i+1;
end
f, he=sum(f)
dlmwrite('pinshu.txt',f); dlmwrite('pinshu.txt',he,'-append');
fclose(fid); %打开的文件使用完之后一定要关闭。
```

3.5 Matlab 与文件相关的操作函数：

1、fopen 函数

fopen 打开一个文件并返回其文件 **id** 号，基本调用形式为：

- fid=fopen(filename,permission)
- [fid,message]=fopen(filename,permission)
- [fid,message]=fopen(filename,permission,format)

若打开成功，fid 返回一个非负整数，message 为空，否则 fid 返回-1，message 返回对错误的解释。 permission 取值如下：

| 权限 | 说明 | 权限 | 说明 |
|-----|-------------------------------|------|-----------------------------|
| 'r' | 以只读方式打开 | 'r+' | 以读写方式打开 |
| 'w' | 删除存在文件的内容（或者创建一个新的文件）并以只写方式打开 | 'w+' | 删除存在文件的内容（或者创建一个新的文件）并以读写打开 |

| | | | |
|-----|-----------------------------------|------|-----------------------------------|
| 'a' | 打开现有文件（或创建新的文件）并以只读方式打开，在文件末尾追加内容 | 'a+' | 打开现有文件（或创建新的文件）并以读写方式打开，在文件末尾追加内容 |
| 'w' | 无自动溢出的写 | 'A' | 无自动溢出的追加 |

format 字符串有'n','l','b','a','s'等，分别代表不同的数值格式

2、fclose 函数

fclose 函数用来关闭一个文件，形式为：status=fclose(fid)或者 status=fclose('all')对于第一种形式如果关闭成功返回 0，否则返回-1。第二种形式关闭除了 stdout(fid=1)和 stderr(fid=2)之外的所有打开的文件。

3、二进制输入/输出函数

● **fwrite** 以用户指定的格式向文件中写入二进制数据，调用形式为：

count=fwrite(fid,array,precision)

count=fwrite(fid,array,precision,skip) fid 是用 fopen 打开的文件号，array 是写入的数据，以列序写入，precision 指定数据输出的格式，有'char','schar','int8','int16','uint8','float32','bitN'等 15 种格式，也可以接受 C/Fortran 类型的数据格式，如'char*1','unsigned char','integer*1'~'integer*8','real*4'等格式。skip 指定是按位(bit)写入。

● **fread** 从文件中按用户指定的格式读取二进制数据

[array,count]=fread(fid,size,precision)

[array,count]=fread(fid,size,precision,skip)

参数 size 有三个选项，n：读 n 个值，array 返回含有 n 个值的列向量；Inf：读到文件末尾，array 返回包含文件所有数据的列向量。[n,m]：读取 n×m 个值

4、格式化输入输出函数 fscanf/fprintf

fscanf 从文件中读取格式化的数据，调用形式为：

array=fscanf(fid,format)

[array,count]=fscanf(fid,format,size)

count=fprintf(fid,format,val1,val2,...) format 是形如 %-12.5e 的格式化字符串，其中-表示左对齐，12 表示数据宽度，.5 表示精度（小数点后的位数），e 表示科学计数法。

其中： array,count,fid,size 含义同上，format 用来指定数据的格式，有三种类型，

一种是诸如%c,%d,%e,%E,%f,%g,%G等制定数据类型；另一种是格式标志如减号(-),加号(+),0, 减号表示左对齐, 加号表示显示”+”字符, 0表示用0做引导符而非空格, 如%05.2d; 第三种是转义字符, 如\n,\t,\b,\r,\f等

5、fgetl 函数

fgetl 函数返回下一行的数据, 不包括行结束符, 调用形式为: line=fgetl(fid),如果下一行是文件结尾, 则返回-1。

6、fgets 函数

fgets 函数返回下一行的数据, 包括行结束符, 调用形式为: line=fgets(fid),如果下一行是文件结尾, 则返回-1。

7、exist 函数

exist 函数用来检查变量, 内置函数或文件的存在性, 调用形式为: ident=exist('item')
ident=exist('item','kind')

例如: exist('A') 返回 0-8, 分别代表不同的文件, 如 0 表示文件不存在, 1 表示 A 是一个变量, 2 表示 A 是一个 M 文件或者一个普通文件, 7 表示目录, 8 代表 java 类等, 'kind' 指示 item 为 var,file,builtin,dir 四种类型之一。

8、其它相关函数, 如 ferror,feof,ftell,frewind,fseek 等, 具体含义可查阅 MATLAB 帮助信息。

示例 1:

```
filename=input('Enter file name:','s');
out_array=randn(1,10000);
[fid,msg]=fopen(filename,'w');
if fid>0
    count=fwrite(fid,out_array,'float64');
    disp([int2str(count) 'values written...']);
    status=fclose(fid);
else
    disp(msg);
end
[fid,msg]=fopen(filename,'r');
if fid>0
    [in_array,count]=fread(fid,[100,100],'float64');
    disp([int2str(count) 'values read...']);
    status=fclose(fid);
else
    disp(msg);
```

end

示例 2: 使用 fscanf 从文件中读取数据

假设文件 test.dat 中有如下两行数据:

10.00 20.00

30.00 40.00

[z,count]=fscanf(fid,'%f'), 得到 z 是一个列向量, count=4;

[z,count]=fscanf(fid,'%f',[2 2]), 得到 z 是[10 30;20 40], count=4

[z,count]=fscanf(fid,'%d',Inf), 得到 z=10,count=1;因为类型不匹配而不继续读取数据。

[z,count]=fscanf(fid,'%d.%d',[1,inf]) 以小数点作为分割

[z,count]=fscanf(fid,'%c')

[z,count]=fscanf(fid,'%s')

示例 3: 使用 fprintf 将数据输出到屏幕上,

a=[10 20 30 40];fprintf('output=%4d %4d\n',a);

fprintf('%-8s\n','string');

fprintf('%d\n',123); fprintf('%6d\n',123); fprintf('%6.4d\n',123);

fprintf('%-6.4d\n',123); fprintf('%+6.4d\n',123);

fprintf('%f\n',123.4); fprintf('%8.2f\n',123.4); fprintf('%4.2d\n',123.4);

fprintf('%10.2e\n',123); fprintf('%10.2E\n',123);

注意: 必须保留\n 换行符。

3.6 图像文件

示例 1: 把一个比较大的 bmp 图像文件 data6.bmp, 转化成比较小的 jpg 文件, 命名成 data7.jpg, 并显示。

解:

a=imread('data6.bmp');

imshow(a)

imwrite(a,'data7.jpg');

figure, imshow('data7.jpg')

示例 2: 生成 10 幅彩色 jpg 文件, 依次命名成 jpq1.jpg, ..., jpg10.jpg。

解:

clc, clear

for i=1:10

str=['jpg',int2str(i),'.jpg']; %int2str()函数将数字转换成字符串

a(:,1)=rand(500); a(:,2)=rand(500)+100; a(:,3)=rand(500)+200;

imwrite(a,str);

end

示例 3: 文件的批量读取

方法一:

cd('E:\2015 暑假数学建模培训\MatlabData\附件 1');

files=dir('*.bmp');

m=size(files,1);

```

G=[];
for i=1:m
    G=[G imread(files(i).name)]; //读取的每个矩阵扩展成一个大矩阵
end
方法二：
A=cell(1,19);%存放原始图片
for j=1:19
    if j-1<10
        imageName=strcat('00',num2str(j-1),'.bmp');
    else
        imageName=strcat('01',num2str(j-11),'.bmp');
    end
    I{j}=imread(imageName);
end
A=I;

```

3.7 数据的标准化

在数据分析之前，我们通常需要先进行数据标准化（normalization），利用标准化后的数据进行数据分析。数据标准化也就是统计数据的指数化。数据标准化处理主要包括数据同趋化处理和无量纲化处理两个方面。数据同趋化处理主要解决不同性质数据问题，对不同性质指标直接加总不能正确反映不同作用力的综合结果，须先考虑改变逆指标数据性质，使所有指标对测评方案的作用力同趋化，再加总才能得出正确结果。数据无量纲化处理主要解决数据的可比性。数据标准化的方法有很多种，常用的有“极值差法”、“标准差法”和“功效系数法”等。经过上述标准化处理，原始数据均转换为无量纲化指标测评值，即各指标值都处于同一个数量级别上，可以进行综合测评分析。

1、数据类型的标准化（归一化）

一般问题的数据指标 x_1, x_2, \dots, x_m 可能有极大型、极小型、中间型和区间型指标。其中极大型是指期望取值越大越好，极小型则取值越小越好，中间型期望取值为适当的中间值最好；区间型是期望取值落在某一个确定的区间内最好。

(1) 极小型：对某个极小型数据指标 x ，

则 $x' = \frac{1}{x} (x > 0)$ ，或 $x' = M - x$ 。

(2)中间型。对某个中间型指标 x ，则有：

$$x' = \begin{cases} \frac{2(x-m)}{M-m}, & m \leq x \leq \frac{1}{2}(M+m) \\ \frac{2(M-x)}{M-m}, & \frac{1}{2}(M+m) \leq x \leq M \end{cases}$$

其中 M 和 m 分别是指标 x 可能取值的最大和最小值。

(3)区间型。对某个区间型指标 x ,则有:

$$x' = \begin{cases} 1 - \frac{a-x}{c}, & x < a \\ 1, & a \leq x \leq b \\ 1 - \frac{x-b}{c}, & x > b \end{cases}$$

其中 $[a, b]$ 为 x 的最佳稳定区间, $c = \max\{a-m, M-b\}$,

M 和 m 分别为 x 可能取值的最大值和最小值。

2、数据指标的无量纲化处理

实际的数据指标中,往往存在着量纲不同的情况,会出现大数吃小数的错误,导致结果不合理。常用的无量纲处理化方法有标准差法,极值差法和功效系数法等。

(1)标准差法: $x'_{ij} = \frac{x_{ij} - \bar{x}_j}{s_j}$

(2)极值差法: $x'_{ij} = \frac{x_{ij} - m_j}{M_j - m_j}$

(3)功效系数法: $x'_{ij} = c + \frac{x_{ij} - m_j}{M_j - m_j} \cdot d$

$x'_{ij} \in [0, 1] (i=1, 2, \dots, n; j=1, 2, \dots, m)$

$$\bar{x}_j = \frac{1}{n} \sum_{i=1}^n x_{ij}$$

$$s_j = \left[\frac{1}{n} \sum_{i=1}^n (x_{ij} - \bar{x}_j)^2 \right]^{1/2}$$

$$M_j = \max_{1 \leq i \leq n} \{x_{ij}\}$$

$$m_j = \min_{1 \leq i \leq n} \{x_{ij}\}$$

其中, c, d 为确定的常数, c 表示平移量, d 表示旋转量, 表示放大或者缩小的倍数。

3、模糊指标的量化处理方法

在实际中,许多问题都涉及到定性或者模糊指标的定量处理问题,如教学质量、科研水平、人员素质、各种满意度、意识、能力等因素有关的政治、社会、人文等领域的问题。按照国家的评价标准,评价因素一般分为 5 个等级,如 A,B,C,D,E 等。对于这类问题,一般采用构造模糊隶属度函数的量化方法进行。

假设有多个评价人对某项因素评价为A, B, C, D, E共5个等级: $\{v_1, v_2, v_3, v_4, v_5\}$ 。

譬如: 评价人对某事件“满意度”的评价可分为{很满意, 满意, 较满意, 不太满意, 很不满意}将其5个等级依次对应为5, 4, 3, 2, 1。

这里为连续量化, 取偏大型柯西分布和对数函数作为隶属函数:

$$f(x) = \begin{cases} [1 + \alpha(x - \beta)^{-2}]^{-1}, & 1 \leq x \leq 3 \\ a \ln x + b, & 3 \leq x \leq 5 \end{cases}$$

其中 α, β, a, b 为待定常数。

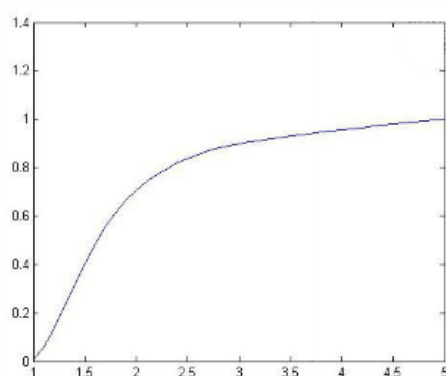
当“很满意”时, 则隶属度为 1, 即 $f(5) = 1$;

当“较满意”时, 则隶属度为 0.8, 即 $f(3) = 0.8$;

当“很不满意”时, 则隶属度为 0.01, 即 $f(1) = 0.01$ 。

计算得 $\alpha = 1.1086, \beta = 0.8942, a = 0.3915, b = 0.3699$ 。

$$f(x) = \begin{cases} [1 + 1.1086(x - 0.8942)^{-2}]^{-1}, & 1 \leq x \leq 3 \\ 0.3915 \ln x + 0.3699, & 3 < x \leq 5 \end{cases}$$



根据这个规律, 对于任何一个评价值, 都可给出一个合适的量化值。

据实际情况可构造其他的隶属函数。如取偏大型正态分布。

$$w_i(x) = \begin{cases} 0, & \text{当 } x \leq \alpha_i \text{ 时,} \\ 1 - e^{-\left(\frac{x - \alpha_i}{\sigma_i}\right)^2}, & \text{当 } x > \alpha_i \text{ 时,} \end{cases}$$

示例 1: 在层次分析法中, 因为不同数据的属性物理意义不同, 通常需要对数据进行归一化处理。下面是一个消费者买车过程中利用层次分析决定购买哪一款车的决策矩

阵构造，其中 X1 表示费用，X2 表示性能，X3 表示款式，很明显，X1 属于费用型指标，

越低越好，而 X2 和 X3 则是效益型指标，越大越好，因此，需要将原始矩阵 $D = \begin{bmatrix} 25 & 9 & 7 \\ 18 & 7 & 7 \\ 12 & 5 & 5 \end{bmatrix}$

都统一为效益型矩阵 $D = \begin{bmatrix} 1/25 & 9 & 7 \\ 1/18 & 7 & 7 \\ 1/12 & 5 & 5 \end{bmatrix}$ ，又因为该矩阵中的数据大小差别较大，因此需

要将矩阵进行列归一化处理， $r_{ij} = \frac{d_{ij}}{\sum_{i=1}^m d_{ij}}$ ，进行计算的 Matlab 代码是：

```
D=[1/25 9 7;1/18,7,7;1/12 5 5];
colSum=sum(D);
GD=[];
for i=1:3
    GD=[GD D(:,i)/colSum(i)];
end
```

4. Matlab 中的标准化和归一化函数

Matlab 中的标准化函数的目标：对原始数据进行缩放处理，限制在一定的范围内。一般指正态化，即均值为 0，方差为 1。

归一化目标：要把你需要处理的数据经过处理后（通过某种算法）限制在你需要的一定范围内。首先归一化是为了后面数据处理的方便，其次是保证程序运行时收敛加快。一般指将数据限制在[0 1]之间。

(1) premnmx 指的是归一到[-1 1],trannmx 是变化测试集输入结果，postmnmx 是转化测试集输出结果。

(2) poststd、prestd 归一到单位方差和零均值。

(3)mapminmax 函数可以把矩阵的每一行归一到[a b].默认为[-1 1]，其计算公式是： $y = (ymax-ymin)*(x-xmin)/(xmax-xmin) + ymin$ 。mapminmax 函数能将矩阵 X 中的每行数据均映射到区间[YMIN,YMAX]内，每行最小值映射为 YMIN，最大值映射为 YMAX。mapminmax 函数还能对变换后矩阵 Y 作逆映射变换，恢复为原始数据矩阵 X。

(4) mapstd 按行逐行地对数据进行标准化处理，将每一行数据分别标准化为均值为 ymean(默认为 0)、标准差为 ystd(默认为 1)的标准化数据，其计算公式是： $y = (x-xmean)*(ystd/xstd) + ymean$ 。

(5) z-score 标准化, 新数据=(原数据-均值)/标准差

示例 1: 用 rand 函数产生一个随机矩阵, 调用 zscore 函数对其按列进行标准化变换。

```
rand('seed',1); %设置随机数生成器的初始种子为 1
```

```
x=[rand(5,1),5*rand(5,1),10*rand(5,1),500*rand(5,1)]
```

```
[xz,mu,sigma]=zscore(x)
```

```
xzmean=mean(xz) %求标准化后矩阵 xz 的各列的均值
```

```
xzstd=std(xz) %求标准化后矩阵 xz 的各列的标准差
```

```
% 调用 zscore 函数对 x 进行标准化变换 (按行标准化)
```

```
% 返回变换后矩阵 xz, 以及矩阵 x 各行的均值构成的向量 mu, 和各行的标准差构成的向量 sigma
```

```
%[xz,mu,sigma]=zscore(x,0,2)
```

```
%xzmean=mean(xz,2) %求标准化后矩阵 xz 的各行的均值
```

```
%xzstd=std(xz,0,2) %求标准化后矩阵 xz 的各列的标准差
```

示例 2.Mapminmax 使用

```
rand('seed',1); %设置随机数生成器的初始种子为 1
```

```
%调用 rand 函数产生一个 5 行 4 列的随机矩阵, 每列服从不同的均匀分布
```

```
x=[rand(5,1),5*rand(5,1),10*rand(5,1),500*rand(5,1)]
```

```
%调用 mapminmax 函数对装置后的 x 按行进行极差归一化变换
```

```
[y,ps]=mapminmax(x',0,1)
```

```
% ps.xmax %查看 ps
```

```
y' %变换后的矩阵
```

```
x0=mapminmax('reverse',y,ps)
```

4、Matlab 编程

M 文件分为两种，一种是脚本文件，由一系列 Matlab 的命令组成，可以直接运行；一种是函数文件，必须由其他 M 文件或者在命令行窗口中调用执行。函数文件具有一定的通用性，并且可以进行递归调用。

4.1 Matlab 中流程控制语句

(1) **if 语句**

有三种形式：

- if(表达式) 语句组 A; end
- if(表达式) 语句组 A, else 语句组 B, end
- if(表达式 1) 语句组 A, elseif(表达式 2) 语句组 B, else 语句组 C, end

(2) 循环语句

- while (表达式) 语句组 A, end
- for k=初值:增量:终值 语句组 A, end

循环语句可以结合 break 命令来控制程序的执行顺序。

(3) 开关结构 **switch 语句**

switch 表达式(标量或者字符串)

```
case 值 1
    语句组 A
case 值 2
    语句组 B
...
otherwise
    语句组 N
end
```

用法示例：

示例 1：输入数 n，判断其奇偶性

解 Matlab 程序如下：

```
n=input('n=');
if rem(n,2)==0 %求余数，与 mod 函数的区别，当同符号时两者相同，否则不同
    A='even';
```

```
else A='odd',
```

```
end
```

示例 2: 求 Matlab 中的最大实数

解: Matlab 程序如下:

```
x=1;
while x~=inf,
x1=x;
x=2*x; %为了获取更接近 realmax 的值, 可以改为 x=1.01*x
end
x1
```

示例 3: 求 Matlab 的相对精度

解: matlab 程序如下:

```
y=1;
while 1+y>1
    y1=y;
    y=y/2;
end
y1
```

示例 4: 利用 Taylor 级数估计 e 的近似值 $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!} + \cdots$

解: Matlab 程序如下:

Method 1:

```
n=10;
temp=1;
S=1;
for i=1:n
    temp=temp*i;
    S=S+1/temp;
end
S
```

Method 2:

```
n=10;
temp=1;
S=1;
i=1;
while i<=n
    temp=temp*i;
    S=S+1/temp;
    i=i+1;
end
```

S

示例 5: Switch 结构演示，学生成绩管理。

```
clear;
%划分区域：满分(100)，优秀(90-99)，良好(80-89)，及格(60-79)，不及格(<60)。
for i=1:10;
    a{i}=89+i;
    b{i}=79+i;
    c{i}=69+i;
    d{i}=59+i;
end;
c=[d,c];
Name={'Jack','Marry','Peter','Rose','Tom'}; %元胞数组
Mark={72,83,56,94,100};Rank=cell(1,5);
%创建一个含 5 个元素的构架数组 S，它有三个域。
S=struct('Name',Name,'Marks',Mark,'Rank',Rank);
%根据学生的分数，求出相应的等级。
for i=1:5
    switch S(i).Marks
    case 100 %得分为 100 时
        S(i).Rank='满分'; %列为'满分'等级
    case a %得分在 90 和 99 之间
        S(i).Rank=' 优秀'; %列为'优秀'等级
    case b %得分在 80 和 89 之间
        S(i).Rank=' 良好'; %列为'良好'等级
    case c %得分在 60 和 79 之间
        S(i).Rank=' 及格'; %列为'及格'等级
    otherwise %得分低于 60
        S(i).Rank='不及格'; %列为'不及格'等级
    end
end
%将学生姓名，得分，登记等信息打印出来。
disp(['学生姓名',' 得分',' 等级']);disp(' ')
for i=1:5;
    disp([S(i).Name,blanks(6),num2str(S(i).Marks),blanks(6),S(i).Rank]);
end;
```

4.2 脚本文件

主要特征如下：

(1)一般由 `clc,clear` 命令开始，清除掉屏幕和工作空间中原有的变量和图形，以避免其他已执行的程序残留数据对本程序的影响。程序中应该添加有注释。

(2)接下来是程序的主体，如果文件中有全局变量，则需要使用 `global` 在程序的起始

部分注明。为了提高程序的可读性，注意语句的缩进。

(3)整个程序应按照 Matlab 标识符的要求起文件名，扩展名为 m。

示例 1：列出求素数的程序

解 Matlab 程序如下：

```
clear,clc
N=input('N='),x=2:N; %列出从 2 到 N 的全部自然数
for u=2:sqrt(N) %依次列出除数
    n=find(rem(x,u)==0&x~=u);%找到能被 u 整除而不等于 u 的数序号
    x(n)=[]; %剔除该数
end
x
```

4.3 函数文件

函数文件与脚本文件的区别：

- (1)由 **function** 开头，后跟的函数名必须与文件名相同；
- (2)有输入变量和输出变量，可进行变量传递；
- (3)除非使用 **global** 声明，程序中的变量均为局部变量，运行后不保存在工作空间中。

函数的基本形式为：

```
function [outarg1,outarg2,...]=fname(inarg1,inarg2,...)
```

```
% H1comment line    用来描述函数的作用
```

```
空白行
```

```
%Other comment lines
```

```
...
```

```
Executalbel code
```

```
...
```

```
(return )
```

示例 1：使用函数文件写出求素数的程序

解：Matlab 程序如下：

```
function y=qiuprime(N)
%求出 1 到 N 之间的素数，返回素数组成的向量
x=2:N; %列出从 2 到 N 的全部自然数
for u=2:sqrt(N) %依次列出除数
```

```

n=find(rem(x,u)==0&x~=u);%找到能被 u 整除而不等于 u 的数序号
x(n)=[]; %剔除该数
end
y=x;

```

示例 2： 写一个递归函数求 $n!$

解：

```

function f=factor(n)
    if n<=1
        f=1;
    else
        f=factor(n-1)*n;
    end
end

```

示例 3： 数据排序—使用选择排序（效率较低，适合于数据量少的情况）

编程的基本步骤：

- （1）描述问题：编程实现给定数据的排序算法；
- （2）定义输入和输出；
- （3）描述算法： 从数组中读取数据；按照升序排列数据（调用函数）；输出排序后的数据；
- （4）返回主程序，输出结果。

完整的程序如下：

```

function out=ssort(a)
%SSORT 按照降序对数据进行选择排序
%选择排序效率较低，不能使用这个函数排序数据量大的情况，如果数据量很大的
话，
%可以使用 sort 函数

%定义变量
% a—要排序的数据
% ii,jj—索引变量
% iptr—指向最小值的指针
% nvals—a 中数据个数
% out—排序结果
% temp—用于交换的临时变量

% 修订记录
%      时间          编程者      修改描述

```



```

% 27/07/2016          zmw          初始代码

% 获取 a 的长度
nvals=size(a,2);

%排序过程
for ii=1:nvals-1
    iptr=ii;
    for jj=ii+1:nvals
        if a(jj)<a(iptr)
            iptr=jj;
        end
    end
    %iptr 现在指向最小的数据，如果 ii 和 iptr 不等的话进行排序
    if ii~=iptr
        temp=a(ii);
        a(ii)=a(iptr);
        a(iptr)=temp;
    end
end
%返回值
out=a;
调用脚本文件：
% 脚本文件： test_ssort.m

% 目的： 检验排序算法 ssort

% 修订记录
%      时间          编程者          修改描述
% 28/07/2014        zmw          初始代码

% 定义的变量
% array—输入数据
% ii—索引
% nvals—输入数据的个数
% sorted—排序号的数据

%输入数据个数
nvals=input('Enter number of values to sort:');
%预分配空间
array=zeros(1,nvals);
%获取数据
for ii=1:nvals
    %输入每一个数据

```

```

        string=['Enter value' int2str(ii) ': '];
        array(ii)=input(string);
    end
    %调用排序函数
    sorted=ssort(array);
    %显示结果
    fprintf('\nSorted data:\n');
    for ii=1:nvals
        fprintf(' %8.4f\n',sorted(ii));
    end

```

示例 4： 使用函数文件计算前面示例中的 e 值

Method 3:

```

function y=fexp(n)
    temp=1;S=1;
    for i=1:n
        temp=temp*i;
        S=S+1/temp;
    end
end
调用方法：
y=fexp(10);

```

4.4 函数参数的可调性

Matlab 中有两个永久变量 `nargin` 和 `nargout` 分别记录调用该函数时的输入实参和输出实参的个数。只要在函数中包含这两个变量，就可以准确的知道该函数文件被调用时的输入输出参数个数，从而决定函数如何处理。其他类似的变量还有 `varargout`, `varargin`。

示例 1:

```

function fout=examp(a,b,c)
if nargin==1
    fout=a;
elseif nargin==2
    fout=a+b;
elseif nargin==3
    fout=a+b+c;
end

```

示例 2:

```

function [x,y,z]=examp(a,b,c)
    if nargout==1
        x=a;

```

```

elseif nargout==2
    x=a;
    y=b;
elseif nargout==3
    x=a;
    y=b;
    z=c;
end
end

```

示例 3:

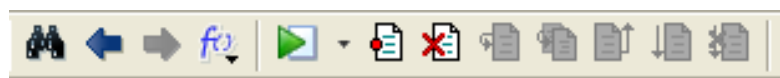
```

function vartest(argA, argB, varargin)
    optargin = size(varargin,2); %可选输入
    stdargin = nargin - optargin; %标准输入
    fprintf('Number of inputs = %d\n', nargin)
    fprintf('  Inputs from individual arguments(%d):\n', ...
        stdargin)
    if stdargin >= 1
        fprintf('      %d\n', argA)
    end
    if stdargin == 2
        fprintf('      %d\n', argB)
    end
    fprintf('  Inputs packaged in varargin(%d):\n', optargin)
    for k= 1 : size(varargin,2)
        fprintf('      %d\n', varargin{k})
    end
end
end

```

4.5 程序调试

M 文件编辑器中提供了强大的程序调试功能，使用方式如同 VC 程序中的调试器。在 M 文件编辑器中的工具栏中有如下程序调试图标：



这些图标的使用和菜单 Debug 中的一些命令相同。

4.6 匿名函数与 inline 函数

有时为了描述某个数学函数的方便，可以用匿名函数来直接编写该数学函数，形式如下：

f=@(变量列表)函数内容

示例 1: `f=@(x,y)sin(x.^2+y.^2)`

调用时，可以直接使用 `f` 句柄，如求函数值 `result=feval(f,1,2);`

该函数还允许直接使用 Matlab 工作空间中的变量，如在工作空间中已经定义了变量 `a` 和 `b`，则匿名函数可以定义为 `f=@(x,y)a*x.^2+b*y.^2;`

早期的 Matlab 版本中使用 `inline` 函数定义匿名函数，但是不如上述过程方便，定义形式为：`fun=inline('函数内容',自变量列表);`

示例 2: `f=inline('sin(x.^2+y.^2)','x','y')`

4.7 伪代码与代码保密处理

主要目的：提高程序的执行速度；对源代码起保密作用

使用命令：`pcode`

`pcode mytest;` %伪代码文件的后缀名是.p，并且该文件不可逆。

4.8 其它编程说明

1、实际编程中，如果能对整个矩阵进行运算时，尽量不要采用循环结构，这样可以提高代码的效率。

例 1：求解级数求和问题

$$S = \sum_{i=1}^{100000} \left(\frac{1}{2^i} + \frac{1}{3^i} \right)$$

解：

循环：

```
tic,
```

```
s=0;
```

```
for i=1:100000
```

```
    s=s+1/2^i+1/3^i;
```

```
end
```

```
toc
```

向量化：

```
tic, i=1:100000;
```

```
S=sum(1./2.^i+1./3.^i);
```

toc

2、在实际编程中，避免滥用递归过程，一般循环方法用极短的事件就能算出来递归调用需要很长时间才能解决的问题。

例 2：求 Fibonacci 数列

递归程序：

```
function a=Fibo(k)
if k==1 | k==2
a=1;
else
a=Fibo(k-1)+Fibo(k-2);
end
tic,Fibo(25),toc
```

循环程序：

```
tic, a=[1 1];
for k=3:100
a(k)=a(k-1)+a(k-2);
end
toc
```

3、使用 parfor 并行计算可以提高大数据量计算的运行效率

matlabpool n; %打开 n 个 worker

matlabpool close %关闭并行

示例 1：

```
c = 1:100000;
a = ones(100000,1);
tic
parfor i = 1:length(c)
    a(i)= a(i)+ c(i);
end
toc
```

```
fprintf('%d\n',a);
```

示例 2:

```
tmp = 5; %临时变量
```

`broadcast = 1;` %广播变量: 在 `parfor` 之前赋值, 在 `parfor` 内只进行读取操作, 不能再 `parfor` 之内对其赋值。

`reduced = 0;` %简约变量: 一般 `parfor` 中各次循环对应的运算应该相互独立, 但简约操作可以在多次循环内同时对一个变量操作。

`sliced = ones(1, 10);` %切片变量: `parfor` 中可能需要读取或写入 `parfor` 之外的矩阵, 读取写入位置与循环变量相关。

```
parfor i = 1:10    %循环变量
```

```
    tmp = i;
```

```
    reduced = reduced + i + broadcast;
```

```
    sliced(i) = sliced(i) * i;
```

```
end
```

4、使用 `logical` 函数进行操作, 此时矩阵操作速度最快。

```
A=[1 2 3;4 5 6;7 8 9];
```

```
B=logical([0 1 0;1 0 1;1 1 0]);
```

```
A(B)
```

5、Matlab 绘图命令

5.1 二维绘图命令

二维绘图的基本命令有 **plot**, **loglog**, **semilogx**, **semilogy** 和 **polar**。它们的使用方法基本相同,其不同特点是在不同的坐标中绘制图形。**plot** 命令使用线性坐标空间绘制图形;**loglog** 命令在两个对数坐标空间中绘制图形;而 **semilogx**(或 **semilogy**)命令使用 x 轴(或 y 轴)为对数刻度,另外一个轴为线性刻度的坐标空间绘制图形;**polar** 使用极坐标空间绘制图形。

二维绘图命令 **plot** 为了适应各种绘图需要,提供了用于控制线色、数据点和线型的 3 组基本参数。它的使用格式如下:

plot(x,y, 'color_point_linestyle')

该命令是绘制 y 对应 x 的轨迹的命令。y 与 x 均为向量,且具有相同的元素个数。用字符串, "color_point_linestyle"完成对上面 3 个参数的设置。

线色: **r-red**, **g-green**, **b-blue**, **w-white**, **k-black**, **i-invisible**, **y-yellow**, **c-cyan** (青色), **m-紫色**。

数据点: **.** (圆点), **o** (小圆圈), **x** (叉号), **+** (加号), ***** (星号), **s** (square 方形), **h** (hexagram 六角星), **d** (diamond 菱型), **p** (pentagram 五角星), **v** (下三角), **^** (上三角), **<** (左三角), **>** (右三角)。

线型: **-** (实线), **--** (虚线), **-.** (点画线), **:** (点线)。

当 **plot(x,y)** 中的 x 和 y 均为 **m×n** 矩阵时, **plot** 命令将绘制 **n** 条曲线。

plot(t,[x1,x2,x3]) 在同一坐标轴内同时绘制三条曲线。

如果多重曲线对应不同的 x 轴向量绘制,可使用命令

plot(t1,x1,t2,x2,t3,x3)

式中 x1 对应 t1, x2 对应 t2 等等。在这种情况下, t1, t2 和 t3 可以具有不同的元素个数,但要求 x1, x2 和 x3 必须分别与 t1, t2 和 t3 具有相同的元素个数。

subplot 命令使得在一个屏幕上可以分开显示 n 个不同坐标系,且可分别在每一个坐标系中绘制曲线。其命令格式如下

subplot(r,c,p)

该命令将屏幕分成 $r \times c$ 个子窗口，而 p 表示激活第 p 个子窗口。窗口的排号是从左到右，自上而下。

示例 1： CUMCA2002A 题 车灯光源投影区域的绘制程序。

```
p=0.03;x=25.0216;
for y1=-0.002:0.0004:0.002
    y0=(-0.036:0.001:0.036)*ones(1,73);
    z0=ones(73,1)*(-0.036:0.001:0.036);
    x0=(y0.^2+z0.^2)/(2*p);
    xn=(p^3+4*x0*2*p.*x0+p*(-4*y1*y0+3*2*p*x0))./(2*(p^2+2*p*x0));
    yn=(2*p*x0.*y0+p^2*(-y1+y0)+y1*(y0.^2-z0.^2))./(p^2+2*p*x0);
    zn=(p^2+2*p*x0+2*y1*y0).*z0./(p^2+2*p*x0);
    y=y0+(yn-y0).*(x-x0)./(xn-x0);
    z=z0+(zn-z0).*(x-x0)./(xn-x0);
    plot(y,z,'b. ');
    xlabel('y');
    ylabel('z')
    hold on
end
```

下面对几个特殊的坐标系统进行简要介绍：

①对数坐标曲线，主要有 **semilogx, semilogy** 和 **loglog**，前两个分别以 x 坐标和 y 坐标为对数坐标，后一个是双对数坐标。

示例 2：

```
x=1:0.1*pi:2*pi;
y=sin(x);
semilogx(x,y,'-s')
x = 0:1:10;
semilogy(x,10.^x)
x = logspace(-1,2);
loglog(x,exp(x),'-s')
grid on
```

②双纵坐标（双 y 坐标系）函数 **plotyy**，调用形式为：

plotyy(X1,Y1,X2,Y2)

plotyy(X1,Y1,X2,Y2,fun) fun 可以是 plot、semilogx、semilogy 或 loglog

plotyy(X1,Y1,X2,Y2,fun1,fun2) fun1 绘制(X1,Y1)，fun2 绘制(X2,Y2)

③极坐标绘图函数 **polar**

调用形式为：**polar(theta,rho)**

示例 3： 绘制极坐标下的平面曲线 $\rho = a \cos(b + n\theta)$ ，并讨论参数 a, b, n 对曲线的影响。

解 Matlab 程序如下：

```
N=100;
theta=linspace(0,2*pi,N);
for i=1:2
    a(i)=input('a=');
    b(i)=input('b=');
    n(i)=input('n=');
    rho(i,:)=a(i)*cos(b(i)+n(i)*theta);
    subplot(1,2,i),polar(theta,rho(i,:));
end
```

几个比较特殊的极坐标系下的图形：

(1)蝴蝶图案

```
t=0:0.01:36;
f=exp(cos(t-pi/2))-2*cos(4*(t-pi/2))+sin((t-pi/2)/12).^5;
polar(t,f,'r')
```

(2)枫叶图案

```
t=-pi/2:0.05:1.5*pi;
f=100./(100+(t-pi/2).^8)*2-sin(7*t)-cos(30*t)/2;
polar(t,f,'r')
```

在图形绘制完毕后，执行如下命令可以再在图中加入标题、标号、说明和分格线等。

这些命令有 **title**, **xlabel**, **ylabel**, **text**, **gtext**, **legend** 等。它们的命令格式如下

```
title("My Title"), xlabel("My X-axis Label"), ylabel("My Y-axis Label"), xlabel({'first
line';'second line'}), text(x, y,'Text for annotation'), gtext('Text for annotation'), grid on
xlabel 和 ylabel 中可以包含上标和小标，分别使用^{上标内容}和_{下标内容}来表示。
```

gtext 命令是使用鼠标器定位的文字注释命令。当输入命令后，可以在屏幕上得到一个光标，然后使用鼠标器控制它的位置。按鼠标器的左键，即可确定文字设定的位置。

hold on 是图形保持命令，可以把当前图形保持在屏幕上不变，同时在这个坐标系内绘制另外一个图形。**hold on** 命令是一个交替转换命令，即执行一次，转变一个状态（相当于 **hold on**、**hold off**）。

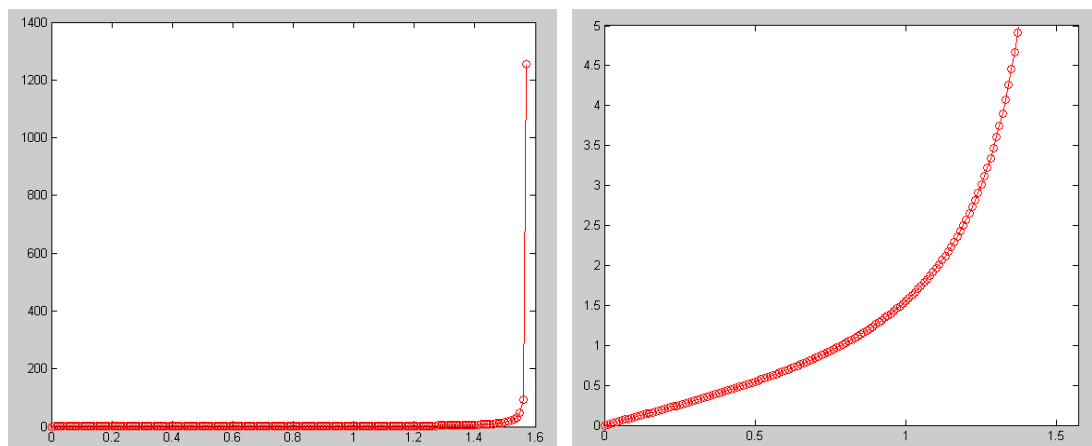
还可以设置坐标轴的范围，使用命令 **axis**，其格式为：

```
axis([xmin xmax ymin ymax])
axis([xmin xmax ymin ymax zmin zmax cmin cmax])
```

示例 4:

$x = 0:0.01:\pi/2;$

`plot(x,tan(x),'-ro')` 图形效果如下所示:



默认情况下,横纵坐标轴的范围是根据函数自变量和因变量的值自动变化的,有时效果不好,此时需要设定横纵坐标轴的范围: `axis([0 pi/2 0 5])`,效果如上右图所示。

例 1: 画出 $y = \sin(x)$, $y = \sin(x + \frac{\pi}{3}) + 2$, $y = \cos(x)$ 的对比图。

解 画图的 Matlab 程序如下,结果见下图 1。

```
clc, clear
x=-2*pi:0.1:2*pi;
y1=sin(x); y2=sin(x+pi/3)+2; y3=cos(x);
plot(x,y1,'-');
hold on %图形保持命令
plot(x,y2,'*-'); plot(x,y3,'-o');
h=legend('sin($x$)','sin($x+\frac{\pi}{3}$)','cos($x$)') %latex 格式显示
set(h,'Interpreter','latex') %设置 Interpreter 的属性值为 latex, 可以使用数学公式
xlabel('$x$', 'Interpreter','latex') %latex 格式显示
ylabel('$y$', 'Interpreter','latex') %latex 格式显示
```

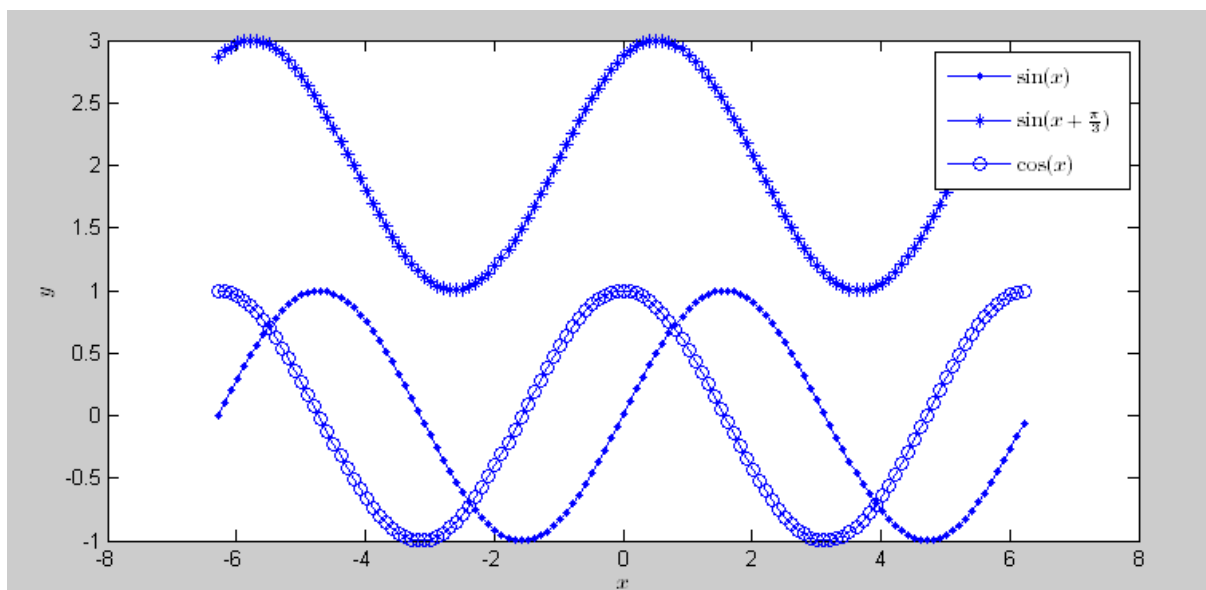


图 1

例3（续例2）在三个子窗口中分别画出 $y = \sin(x)$ ， $y = \sin(x + \frac{\pi}{3}) + 2$ ， $y = \cos(x)$ 。

解 画图的 Matlab 程序如下，结果见图 2。

```
clc, clear
x=-2*pi:0.1:2*pi;
y1=sin(x); y2=sin(x+pi/3)+2; y3=cos(x);
subplot(3,1,1), plot(x,y1,'-'), title('sin($x$)', 'Interpreter', 'latex')
subplot(3,1,2), plot(x,y2,'*-')
title('sin($x+\frac{\pi}{3}$)', 'Interpreter', 'latex')
ylabel('$y$', 'Interpreter', 'latex') %latex 格式显示
subplot(3,1,3), plot(x,y3,'-o')
title('cos($x$)', 'Interpreter', 'latex') %latex 格式显示
xlabel('$x$', 'Interpreter', 'latex') %latex 格式显示
```

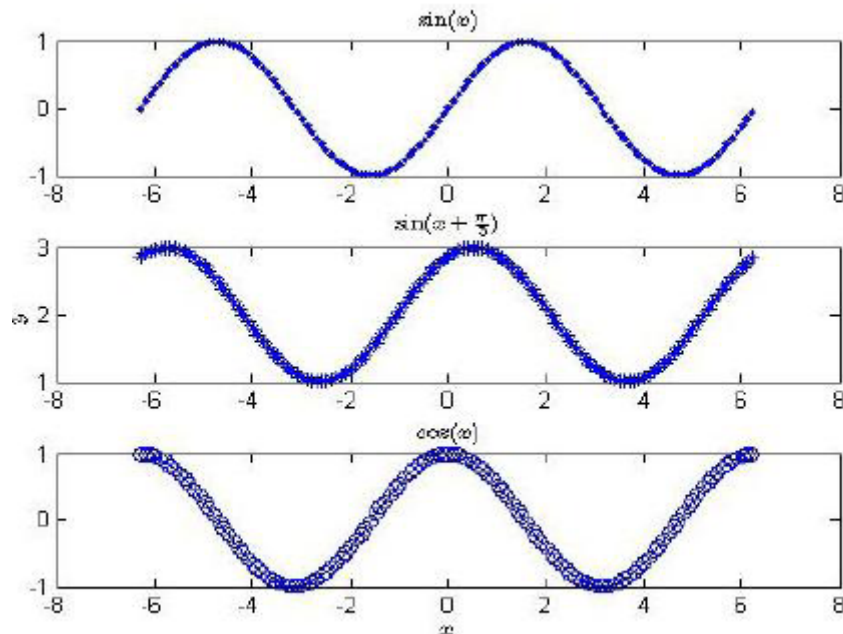


图 2

5.2 复数绘图

`plot(z)` : z 为复数时相当于 `plot(real(z),imag(z))`; 如果是双变量如 `plot(t,z)`, 则 z 中的虚数部分将丢弃。在复平面中绘出多条曲线, 必须使用 **hold on** 命令, 或者把多条曲线的实部和虚部明确的写出。即 **`plot(real(z1),imag(z1),real(z2),imag(z2))`**

Matlab 中专门用来绘制复变量函数图形的相关命令是 **`cplxmap`**, **`cplxgrid`**, **`cplxroot`**, 格式如下:

`z = cplxgrid(m)` %产生 $(m+1)*(2m+1)$ 的极坐标下的复数数据网格

`cplxmap(z,f(z),(optional bound))` %绘制复变函数的图形, 以 xy 平面为自变量所在的复平面, 以 z 轴表示复变函数的实部, 颜色表示复变函数的虚部。

`cplxroot`: 画复数的 n 次函数曲面

`cplxroot(n)` %画复数 n 次根的函数曲面, 复数为最大半径为 1 的圆面

`cplxroot(n,m)` %画复数 n 次根的函数曲面, 复数为最大半径为 1 的圆面, 为 $(m+1) * (2m+1)$ 的方阵

例 1: 绘制 $z=\exp((-0.1+i)*t)$ 的复数图形

解 Matlab 程序如下:

```
t=0:0.1:15;
z=exp((-0.1+i)*t);
subplot(2,2,1)
```

plot(z),pause; % **pause** 可以暂停程序的执行，通过按任意键可使程序继续进行，也可以在 pause(n)中设置时间使执行结果出现动态效果。

```
title('复数绘图 plot(z)');
subplot(2,2,2)
plot(t,z),pause
title('plot(t,z)')
subplot(2,2,3)
polar(angle(z),abs(z));pause;
title('polar(angle(z),abs(z))')
subplot(2,2,4)
semilogx(t,z);
title('semilogx(t,z)')
```

注：Matlab 中使用函数 **C=complex(A,B)**构造复数。

例 2：绘图 $y=\cos(x+i)$ 的图形

解法一： $x=-\pi:0.1:\pi$;

```
fun=@(x)cos(x+i); %如果直接使用 plot(x,y)，会忽略虚部。
plot(fun(x))
```

解法二： plot(cos(x+i));

例 3：绘制 $f=z^4$ 的图形，其中 z 为复数

解： $z=\text{cplxgrid}(30)$;

```
cplxmap(z,z.^4)
```

5.3 显函数，符号函数或隐函数的绘图

fplot(fun,lims)绘制由字符串 fun 指定函数名的函数在 x 轴区间为 lims=[xmin, xmax] 的函数图。若 lims=[xmin,xmax,ymin,ymax]，则 y 轴也被限制。

例 4 画 $f(x) = \begin{cases} x+1, & x < 1 \\ 1+\frac{1}{x}, & x \geq 1 \end{cases}$ 的图形。

解 （1）首先用 M 文件 fun1.m 定义函数 f(x)如下

```
function y=Afun1(x);
```

```
if x<1
```

```
    y=x+1;
```

```
else
```

```
y=1+1./x;
```

```
end
```

在 matlab 命令窗口输入

```
fplot('Afun1',[-3,3])
```

就可画出函数 $f(x)$ 的图形。

(2) 可以使用**匿名函数**，编写程序如下

```
fun2=@(x) (x+1)*(x<1)+(1+1/x)*(x>=1);
```

```
fplot(fun2,[-3,3])
```

ezplot(f)绘制符号函数或者隐函数 $f(x)$ 的图形， x 轴的近似范围为 $[-2\pi, 2\pi]$ ，可以根据需要修改定义域的范围。

ezplot(f,[xmin,xmax])使用输入参数来代替默认横坐标范围 $[-2\pi, 2\pi]$ 。ezplot 函数的其他格式有：

ezplot(fun2)绘制 $\text{fun2}(x,y)=0$ 的隐函数曲线，默认 x,y 的范围是 $[-2\pi, 2\pi]$

```
ezplot(fun2,[xmin,xmax,ymin,ymax])
```

示例：

```
function z = myfun(x,y,k) %建立 M 文件
```

```
z = x.^k - y.^k - 1;
```

```
%在命令窗口中输入： ezplot(@(x,y)myfun(x,y,2));
```

例 5 画出函数 $y = \text{ctg}x$ 的图形

解 `ezplot('cot(x)')`

例 6 画出椭圆 $x^2 + \frac{y^2}{4} = 1$ 的图形。

解 `ezplot('x^2+y^2/4=1')`

例 6-1：同一坐标系下绘制曲线 $x^2+y^2=1$ 和 $x^2-y^2=1$ 的曲线

解：`ezplot('x^2-y^2=1');`

```
hold on;
```

```
ezplot('x^2+y^2=1');
```

```
colormap([0,0,1]);
```

5.4 三维图形

在实际工程计算中，最常用的三维绘图是三维曲线图、三维网格图和三维曲面图 3 种基本类型。与此对应，Matlab 也提供了一些三维基本绘图命令，如三维曲线命令 **plot3**，三维网格图命令 **mesh** 和三维表面图命令 **surf**。

1. 三维曲线

plot3(x,y,z)通过描点连线画出曲线，这里 x,y,z 都是 n 维向量，分别表示该曲线上点集的横坐标、纵坐标、竖坐标。

例 7

```
t=0:0.05:20*pi;
x=sin(t);
y=cos(t);
z=t.*sin(t).*cos(t);
plot3(x,y,z), title('Line in 3-D Space')
xlabel('X'), ylabel('Y'), zlabel('Z'), grid on
```

2. 三维网格图

命令 **mesh(x,y,z)**画网格曲面。这里 x,y,z 是三个同维数的数据矩阵，分别表示数据点的横坐标、纵坐标、竖坐标，命令 **mesh(x,y,z)**将该数据点在空间中描出，并连成网格。

例 8 绘制二元函数

$$z = \frac{\sin(xy)}{xy}$$

的三维网格图。

解： `x=-3:0.1:3;y=-5:0.1:5;`

```
x1=ones(size(y))*x;y1=y'*ones(size(x));
```

`[x2,y2]=meshgrid(x,y); %meshgrid 函数生成 2D/3D 网格矩阵，用于分割空间，在绘制 3D 网格图或者表面图时都需要调用该函数对生成绘图时所用的数据。`

```
z1=(sin(x1.*y1)+eps)./(x1.*y1+eps);
```

```
z2=(sin(x2.*y2)+eps)./(x2.*y2+eps);
```

```
subplot(1,2,1),mesh(x1,y1,z1)
```

```
subplot(1,2,2),mesh(x2,y2,z2)
```

例 8-1：绘制两个空间相交的平面 $z=x+2y-1$ 和 $z=x-y+2$

解：

```
x=-3:0.1:3;y=-3:0.1:3;
```

```
[X,Y]=meshgrid(x,y);
```

```
mesh(X,Y,X+2*Y-1)
```

```
hold on
```

```
mesh(X,Y,X-Y+2)
```

3. 表面图

命令 **surf(x,y,z)** 画三维表面图，这里 x,y,z 是三个同维数的数据矩阵，分别表示数据点的横坐标、纵坐标、竖坐标。

例 9 绘制二元函数

$$z = \frac{\sin(xy)}{xy}$$

的三维表面图。

解： `[x,y]=meshgrid([-3:0.2:3]);`
`z=(sin(x.*y)+eps)./(x.*y+eps);`
`surf(x,y,z)`

注意 1： surf 函数有一些变体形式，如：**surfc()**：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；**surfl()**：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；**waterfall()**：绘制瀑布图。
surfnorm()：计算与显示三维曲面的法线。该命令计算用户命令 surf 中的曲面法线。

(1) `[X,Y,Z] = peaks(30);`
`surfc(X,Y,Z)`
`colormap hsv`
(2) `[X,Y] = meshgrid(-3:1/8:3);`
`Z = peaks(X,Y);`
`surfl(X,Y,Z);`
`shading interp`
`colormap(gray);`
(3) `[X,Y,Z] = peaks(30);`
`waterfall(X,Y,Z)`
(4) `[x,y,z] = cylinder(1:10);`
`surfnorm(y,x,z)`
`axis([-12 12 -12 12 -0.1 1])`

注意 2： shading()：设置颜色色调属性。该命令控制曲面与补片等的图形对象的颜色色调。同时设置当前坐标轴中的所有曲面与补片图形对象的属性 EdgeColor 与 FaceColor。主要用法：(1) shading flat 使网格图上的每一线段与每一小面有一相同颜色，该颜色由线段的末端的端点颜色确定；或由小面的、有小型的下标或索引的四个角

的颜色确定。(2) shading faceted 带重叠的黑色网格线的平面色调模式。这是缺省的色调模式。(3) shading interp 在每一线段与曲面上显示不同的颜色,该颜色为通过在每一线段两边的、或者为不同小曲面之间的色图的索引或真颜色进行内插值得到的颜色。

4. 旋转曲面

方法一: 使用命令函数 **cylinder**

$[X,Y,Z] = \text{cylinder}(r)$ 这里的 r 表示构成旋转曲面的曲线。

例10 画出

$$y = 30e^{-\frac{x}{400}} \sin\left(\frac{1}{100}(x + 25\pi)\right) + 130, \quad x \in [0, 600]$$

绕 x 轴旋转一周形成的旋转曲面。

解 Matlab 程序如下。

```
x=0:10:600;
[X,Y,Z]=cylinder(30*exp(-x/400).*sin((x+25*pi)/100)+130);
surf(X,Y,Z)
```

方法二: 将旋转曲面用参数方程表示。

例 11 画出 $x^2 + (y-5)^2 = 16$ 绕 x 轴旋转一周所形成的旋转曲面。

解 因为这里的函数是隐函数,化成显函数后有两支,必须使用参数方程,旋转面的参数方程为

$$\begin{aligned} x &= 4\cos\alpha, \\ y &= (5 + 4\sin\alpha)\cos\beta \\ z &= (5 + 4\sin\alpha)\sin\beta \quad \text{其中 } \alpha, \beta \in [0, 2\pi]. \end{aligned}$$

画图的 Matlab 程序如下:

```
alpha=[0:0.1:2*pi]'; beta=0:0.1:2*pi;
x=4*cos(alpha)*ones(size(beta));
y=(5+4*sin(alpha))*cos(beta);
z=(5+4*sin(alpha))*sin(beta);
surf(x,y,z)
```

或者利用绘制三维隐函数曲面图形的命令 **ezsurf** 或者 **ezmesh**, 其命令格式如下:

```
ezmesh(fun)
```

ezmesh(fun,domain) domain 要求是一个向量,指定 x,y 轴的范围, fun 是函数句柄

ezmesh(funx,funy,funz) % 参数方程绘图 funx(s,t), funy(s,t), and funz(s,t) over the

square: $-2\pi < s < 2\pi, -2\pi < t < 2\pi$.

```
ezsurf(fun)
```

```
ezsurf(fun,domain)
```

```
ezsurf(funx,funy,funz)
```

画图的 Matlab 程序也可以写成

```
x=@(alpha,beta) 4*cos(alpha);
```

```
y=@(alpha,beta) (5+4*sin(alpha))*cos(beta);
```

```
z=@(alpha,beta) (5+4*sin(alpha))*sin(beta);
```

```
ezsurf(x,y,z)
```

对于形如 $f(x,y,z)=0$ 的三维隐函数的图像没有现成的函数可以画，但可以利用

isosurface 函数绘制三角网格图。例如

```
f=@(x,y,z)x.^2+y.^2+z.^2-10;%定义函数 f=x^2+y^2+z^2-10
```

```
[x,y,z]=meshgrid(linspace(-4,4,25));%设定网格大小和范围
```

```
val=f(x,y,z);
```

```
[p,v]=isosurface(x,y,z,val,0);%用 isosurface 得到函数 f=0 图形的点和面
```

```
patch('faces',p,'vertices',v,'facevertexcdata',jet(size(v,1)),'facecolor','w','edgecolor','flat');%
```

用 patch 绘制三角网格图并设定色彩

```
view(3);
```

```
grid on;
```

```
axis equal
```

注： patch 函数用来创建一个或多个填充多边形。

5、绘制柱面

柱面平行于某个坐标轴，方程中不出现某个坐标轴的变量，方程表示为 $F(x,y)=0$ 或者 $F(x,z)=0$ 或者 $F(y,z)=0$

示例 1： 画出方程 $z = 2 - x^2$ 表示的柱面

解：方程中的 x 是自变量矩阵， z 是因变量，则另一个自变量矩阵为 y ，自变量平面是 xoy 面， x 轴是真正的自变量， y 轴是柱面方向。Matlab 程序如下：

```
u=linspace(-5,5,10)';%设定参数列向量 u
```

```
v=linspace(-5,10,10); %设定参数行向量 v
```

```
X=u*ones(size(v)); %构成自变量矩阵 X
```

```
Y=ones(size(u))*v; %构成自变量矩阵 Y，X,Y 的数据可以直接使用 meshgrid 生成
```

```
Z=2-X.^2; %求因变量 Z
```

```
mesh(X,Y,Z)
```

示例 2： 画出方程 $-x^2 + \frac{y^2}{4} = 1$ 表示的柱面

解：方程整理为显示函数： $y = \pm 2\sqrt{1+x^2}$ ，分别绘制正负两个分量。

```

clear,clc
u=linspace(-5,5,10)';%设定参数列向量 u
v=linspace(-5,10,10); %设定参数行向量 v
X1=u*ones(size(v)); %构成自变量矩阵 X1
Y1=2*sqrt(1+X1.^2); %求因变量 Y1
Y2=-2*sqrt(1+X1.^2);
Z1=ones(size(u))*v; %求因变量 Z1
mesh(X1,Y1,Z1),hold on
mesh(X1,Y2,Z1)

```

6. 其它二次曲面

对于旋转面，如果母线的方程可以表示成关于旋转轴变量的显式函数，则可以直接使用 Matlab 工具箱中的命令 **cylinder**，否则必须把旋转面化成参数方程，然后使用 **ezmesh** 或 **ezsurf** 命令绘图。对于其它的二次曲面，如果可以写成显函数直接使用命令 **ezmesh** 或 **ezsurf**，否则必须先化成参数方程。还有一些特殊的绘制函数如 **ellipsoid** 等命令。

例 12 画出下列曲面的图形

- (1) 旋转单叶双曲面 $\frac{x^2 + y^2}{9} - \frac{z^2}{4} = 1$;
- (2) 旋转双叶双曲面 $\frac{x^2}{9} - \frac{y^2 + z^2}{4} = 1$;
- (3) 抛物柱面 $y^2 = x$;
- (4) 椭圆锥面 $\frac{x^2}{9} + \frac{y^2}{4} = z^2$;
- (5) 椭球面 $\frac{x^2}{9} + \frac{y^2}{4} + \frac{z^2}{6} = 1$;
- (6) 马鞍面 $z = xy$;
- (7) 椭圆柱面 $\frac{x^2}{9} + \frac{y^2}{4} = 1$.

解：

```

(1) x=@(s,t) 3*sec(s)*cos(t); %化为参数方程
y=@(s,t) 3*sec(s)*sin(t);
z=@(s,t) 2*tan(s);
ezmesh(x,y,z)
(2) x=@(s,t) 3*sec(s);
y=@(s,t) 2*tan(s)*cos(t);
z=@(s,t) 2*tan(s)*sin(t);
ezmesh(x,y,z)

```

```

(3) ezsurf(@(y,z) y.^2,50) %直接调用 ezsurf
(4) x=@(s,t) 3*tan(s)*cos(t);
y=@(s,t) 2*tan(s)*sin(t);
z=@(s,t) tan(s);
ezsurf(x,y,z)
(5) ellipsoid(0,0,0,3,2,sqrt(6)) %专门绘制椭球面
(6) ezsurf(@(x,y)x*y)
(7) x=@(s,t) 3*cos(s);
y=@(s,t) 2*sin(s);
z=@(s,t) t;
ezmesh(x,y,z)

```

7、绘制空间两曲面的交线

示例 1： 绘制由水平截面与方程 $z = x^2 - 2y^2$ 构成的马鞍面形成的交线，并讨论等高线和方向导数（梯度）的意义

解：水平平面与曲面的交线就是**等高线**，在 Matlab 中绘制等高线有两个命令，**contour** 和 **contour3**，前者把等高线画在 xoy 平面上，后者把等高线画在一定高度的平面上，使之成为立体的，与所在曲面对应。

Matlab 程序如下：

```

clc,clear
[x,y]=meshgrid(-10:2:10);%确定计算和绘图的定义域网格
z1=(x.^2-2*y.^2)+eps;%第一个曲面方程
a=input('a=(-50<a<50)');
z2=a*ones(size(x));%水平面方程 z2=a, z2 必须与 x, y 具有相同的维数,
subplot(1,3,1),mesh(x,y,z1);hold on;mesh(x,y,z2);%分别画出两个曲面
v=[-10 10 -10 10 -100 100];axis(v),grid;%确定第一个分图的坐标系
colormap(gray);hold off;
r0=abs(z1-z2)<=1;%求两曲面 z 坐标只差小于 0.5 的网格
zz=r0.*z2;yy=r0.*y;xx=r0.*x;%求这些网格上的坐标值，即交线坐标值
subplot(1,3,2),plot3(xx,yy,zz,'x');%画出这些点
axis(v),grid;%使第二个分图取第一个分图的坐标系
pause,subplot(1,3,3);
contour3(x,y,z1,20);%用等高线命令求出 20 条不同高度的交线。

```

等高线与方向导数和梯度的概念密切相关，函数 $z1$ 在每一点的梯度与该处的等高线垂直，也就是指向**最陡**的方向。Matlab 中求梯度用 **gradient** 函数，**quiver** 函数画出梯度向量，这两个函数要求在给定的点阵上求梯度和画梯度向量。Matlab 程序如下：

```

[x,y]=meshgrid(-10:2:10);
z1=(x.^2-2*y.^2)+eps;
contour(x,y,z1,20);hold on;
[px,py]=gradient(z1,2,2);%以步长为 2 求 z1 的梯度的 x, y 分量

```

`quiver(x,y,px,py); %绘制梯度向量`

注 1: 三维图形中有时使用 `shading` 命令修饰其显示形式, 具有三个不同的选型, `flat` (不带网格线), `interp` 和 `faceted`(默认选项, 带网格线)

注 2: 绘制等高线并添加等高线数值时使用 `clabel` 函数与 `contour` 搭配使用。`Contourf` 可以绘制填充的等高线图, `contour3` 绘制三维等高线图。

示例 2:

```
[x,y]=meshgrid(-1:1:1,-2:1:2);
z=0.5457*exp(-0.75*y.^2-3.75*x.^2-1.5*x).*(x+y>1)+...
    0.7575*exp(-y.^2-6*x.^2).*((x+y>-1) & (x+y<=1))+...
    0.5457*exp(-0.75*y.^2-3.75*x.^2+1.5*x).*(x+y<=-1);
[C,h]=contour(x,y,z);
clabel(C,h);
%contourf(x,y,z);
figure;
contour3(x,y,z,30)
```

注 3: 三维曲面的视角和旋转变换分别使用 `view` 函数和 `rotate` 函数进行设置。

示例 3:

```
h=surf(x,y,z);
rot_ax=[1,0,0];
axis tight;%保证坐标尺度不变
for i=0:360
    rotate(h,rot_ax,1);
    pause(0.02);
end
```

8、离散点的图形绘制相关函数 **griddata, TripScatterInterp, scatter (scatter3)**

griddata 用来对离散数据进行曲面拟合

(1)`ZI = griddata(x,y,z,XI,YI)` 用二元函数 $z=f(x,y)$ 的曲面拟合有不规则的数据向量 x,y,z 。`griddata` 将返回曲面 z 在点 (XI,YI) 处的插值。曲面总是经过这些数据点 (x,y,z) 的。输入参量 (XI,YI) 通常是规则的格点 (像用命令 `meshgrid` 生成的一样)。 XI 可以是一行向量, 这时 XI 指定一有常数列向量的矩阵。类似地, YI 可以是一列向量, 它指定一有常数行向量的矩阵。

(2)`[XI,YI,ZI] = griddata(x,y,z,xi,yi)` 返回的矩阵 ZI 含义同上, 同时, 返回的矩阵 XI,YI 是由行向量 xi 与列向量 yi 用命令 `meshgrid` 生成的。

(3)`[XI,YI,ZI] = griddata(.....,method)`

用指定的算法 `method` 计算:

‘linear’: 基于三角形的线性插值（缺省算法）;

‘cubic’: 基于三角形的三次插值;

‘nearest’: 最邻近插值法;

示例 1:

```
x = rand(100,1)*4-2; y = rand(100,1)*4-2; z = x.*exp(-x.^2-y.^2);
ti = -2:.25:2;
[xi,yi] = meshgrid(ti,ti);
zi = griddata(x,y,z,xi,yi);
mesh(xi,yi,zi), hold on, plot3(x,y,z,'o'), hold off
```

在新版本的 Matlab 中，该函数逐渐被 TriScatteredInterp 函数所代替，TriScatteredInterp 的示例如下：

```
x = rand(100,1)*4-2;
y = rand(100,1)*4-2;
z = x.*exp(-x.^2-y.^2);
F = TriScatteredInterp(x,y,z);
ti = -2:.25:2;
[qx,qy] = meshgrid(ti,ti);
qz = F(qx,qy);% 计算指定位置的插值
mesh(qx,qy,qz);
hold on;
plot3(x,y,z,'o');
```

Matlab 中绘制散点图的命令是 **scatter (scatter3)**，命令格式如下：

scatter(X,Y,S,C) %X, Y 对应散点值，S 代表标记大小，C 代表颜色值，它都可以是向量。

```
scatter3(X,Y,Z,S,C)
```

示例 2:

```
A=[1.486,3.059,0.1;2.121,4.041,0.1;2.570,3.959,0.1;3.439,4.396,0.1;
4.505,3.012,0.1;3.402,1.604,0.1;2.570,2.065,0.1;2.150,1.970,0.1;
1.794,3.059,0.2;2.121,3.615,0.2;2.570,3.473,0.2;3.421,4.160,0.2;
4.271,3.036,0.2;3.411,1.876,0.2;2.561,2.562,0.2;2.179,2.420,0.2;
2.757,3.024,0.3;3.439,3.970,0.3;4.084,3.036,0.3;3.402,2.077,0.3;
2.879,3.036,0.4;3.421,3.793,0.4;3.953,3.036,0.4;3.402,2.219,0.4;
3.000,3.047,0.5;3.430,3.639,0.5;3.822,3.012,0.5;3.411,2.385,0.5;
3.103,3.012,0.6;3.430,3.462,0.6;3.710,3.036,0.6;3.402,2.562,0.6;
3.224,3.047,0.7;3.411,3.260,0.7;3.542,3.024,0.7;3.393,2.763,0.7];
x=A(:,1);y=A(:,2);z=A(:,3);
scatter(x,y,5,z)% 散点图
```

示例 3: 2011 国赛建模 A 题 重金属污染地形图绘制

```

clc
clear
close all
D=xlsread('E:\2015 暑假数学建模培训\2011A\2011A.xls','附件 1','A4:E322');
M=xlsread('E:\2015 暑假数学建模培训\2011A\2011A.xls','附件 2','A4:I322');
x=D(:,2); y=D(:,3); z=D(:,4); c=D(:,5);
xi=linspace(min(x),max(x),100);
yi=linspace(min(y),max(y),100);
[xi,yi]=meshgrid(xi,yi);
zi=griddata(x,y,z,xi,yi);
ci=griddata(x,y,c,xi,yi);
marker={'*','o','s','^','p'};
color={'k','r','y','c','b'};
mat={'As','Cd','Cr','Cu','Hg','Ni','Pb','Zn'};
str={'等高线','生活区','工业区','山林区','交通区','绿地区'};
for j=1:8      % 等高线图形
    figure
    h=contourf(xi,yi,zi,0:10:500);
    title(['金属 ',mat{j}, ' 二维等高线分布图'])
    xlabel('X')
    ylabel('Y')
    colormap summer
    colorbar
    grid on
    hold on
    for i=1:5
        loc=c==i;
        plot(x(loc),y(loc),marker{i},'markerfacecolor',color{i},'MarkerEdgeColor',color{i});
    end
    legend(str,'location','best')
    for k=1:length(x)
        text(x(k)-200,y(k)+200,num2str(M(k,j+1)),'fontsize',8);
    end
end
end
% 三维地形图
figure
h=surf(xi,yi,zi);
set(h,'cdata',ci);
colormap hsv
title('三维图立体（颜色条表示分类）')
xlabel('X')
ylabel('Y')
colorbar
hidden off

```

```

hold on
for i=1:5
    loc=c==i;
    plot3(x(loc),y(loc),z(loc),marker{i},'markerfacecolor',color{i});
end
str{1}='三维图';
legend(str,'location','best')

```

5.5 特殊的二维/三维图形

下表列出了 Matlab 中的一些特殊二维/三维绘图函数：

| 功能类别 | 命令函数 | 说 明 |
|---------------|-----------------|--------------------|
| 特殊的二维图形 | area | 面积填充图 |
| | bar | 条形图 |
| | barh | 水平条形图 |
| | comet | 彗星图 |
| | compass | 画区域图 |
| | errorbar | 误差条形图 |
| | ezplot | 利用函数绘图器绘图 |
| | ezpolar | 利用三维参数曲线绘图器绘图 |
| | feather | 绘制箭头图 |
| | fill | 2 维填充的多边形 |
| | fplot | 绘制函数图形 |
| | hist | 直方图 |
| | pareto | 绘制排列图 |
| | pie | 绘制饼状图 |
| | plotmatrix | 绘制矩阵的分散图 |
| | rose | 角度直方图 |
| | scatter | 分散图 |
| | stairs | 阶梯图 |
| | stem | 离散序列图或者杆状图 |
| 2 维和 3 维等高线绘图 | clabel | 在等高线图上增加高度标记 |
| | contour | 绘制等高线图 |
| | contourf | 填充的等高线图 |
| | contour3 | 绘制 3 维等高线图 |
| | ezcontour | 利用等高线绘图器绘制等高线图形 |
| | ezcontourf | 利用等高线绘图器绘制填充的等高线图形 |
| | pcolor | 画伪色彩图形 |
| 特殊的 3 维图形 | voronoi | 画 Voronoi 图 |
| | bar3 | 3 维条形图 |
| | bar3h | 水平方向的 3 维条形图 |
| | comet3 | 3 维彗星图 |

| | | |
|----------|-------------------|---------------------|
| | ezgraph3 | 绘制满足一般要求的表面图 |
| | ezmesh | 利用 3 维网格绘图器绘图 |
| | ezmeshc | 利用网格 / 等高线绘图器绘图 |
| | ezplot3 | 利用 3 维参数曲线绘图器绘图 |
| | ezsurf | 利用 3 维表面阴影绘图器绘图 |
| | ezsurfz | 利用 3 维表面 / 等高线绘图器绘图 |
| | meshc | 网格/等高线混合图形 |
| | meshz | 带参考平面的 3 维网格图 |
| | pie3 | 3 维饼状图 |
| | ribbon | 带状图 |
| | scatter3 | 3 维离散图 |
| | stem3 | 3 维杆图 |
| | surfz | 表面/等高线混合图形 |
| | trimesh | 三角网格图 |
| | trisurf | 三角曲面图 |
| | waterfall | 落差图 |
| 体积和矢量可视化 | coneplot | 3 维锥形图 |
| | contourslice | 在截面上绘制等高线 |
| | curl | 计算矢量场的旋度后角速度 |
| | divergence | 计算矢量场的散度 |
| | interpstreamspeed | 计算流场的顶点 |
| | isocaps | 计算等值面的末端 |
| | isocolors | 计算等值面的颜色 |
| | isonormals | 计算等值面顶点的范数 |
| | isosurface | 从体积数据中提取等值面数据 |
| | quiver | 2 维箭头图 |
| | quiver3 | 3 维箭头图 |
| | reducepatch | 减少填补面的数值 |
| | reducevolume | 减少体积数据集的数值 |
| | shrinkfaces | 减少填补面的尺寸 |
| | Slice | 立体截面图 |
| | smooth3 | 光滑 3 维数据 |
| | Streamline | 绘制 2 维或 3 维矢量的流线数据 |
| | stream2 | 计算 2 维流线数据 |
| | stream3 | 计算 3 维流线数据 |
| | streamparticles | 画出流极小量 |
| | streamribbon | 绘制流带图 |
| | streamslice | 在截平面上绘制流图 |
| | streamtube | 绘制流的管状图 |
| | subvolume | 提取体积数据集的子集 |
| | vissuite | 可视序列 |
| | volumebounds | 返回体积数据的坐标和颜色范围 |

| | | |
|--------------|------------|--------------------|
| 图形显示及文件输入/输出 | brighten | 增量和变暗颜色板 |
| | colorbar | 显示颜色条(代表颜色比例) |
| | colormap | 颜色查找表 |
| | contrast | 调节对比度颜色板以增强图像的对比效 |
| | gray | 线性灰度变换表 |
| | image | 显示图像 |
| | imagesc | 缩放数据并显示成图像 |
| | iminfo | 图形文件的有关信息 |
| | imread | 从图形文件读入图像 |
| | imwrite | 将图像写入图形文件 |
| 动画 | capture | 抓取当前屏幕的图形 |
| | frame2im | 将动画帧转换为编入索引的图像 |
| | getframe | 获取动画帧 |
| | im2frame | 将编入索引的图像转换为动画帧格式 |
| | movie | 播放所记录的动画帧 |
| | moviein | 动画帧内存初始化 |
| | rotate | 沿着指定的原点和方向旋转对象 |
| 与颜色有关的函数 | colstyle | 从字符串中分出颜色和样式 |
| | ind2rgb | 将编入索引的图像转换为 RGB 图像 |
| | rgbplot | 绘制颜色板图形 |
| | spinmap | 旋转颜色板 |
| 立体建模 | cylinder | 产生圆柱体 |
| | ellipsoid | 产生椭圆柱体 |
| | patch | 创建图形填充块 |
| | sphere | 产生球 |
| | surf2patch | 将表面数据转换为图形填充块数据 |

例 1：山崩地裂特效

```

data=rand(10,10,10);
datas=smooth3(data,'box',3);
subplot(121)
p=patch(isosurface(data,0.5));
patch(isocaps(data,0.5));
isonormals(data,p);
camlight
lighting phong
axis vis3d off
view(3)
subplot(122)
p=patch(isosurface(datas,0.5));
patch(isocaps(datas,0.5));
isonormals(datas,p);
camlight

```

lighting phong

axis vis3d off

view(3)

例 2：分段函数绘制

$$p(x_1, x_2) = \begin{cases} 0.5457 \exp(-0.75 x_2^2 - 3.75 x_1^2 - 1.5 x_1), & x_1 + x_2 > 1 \\ 0.7575 \exp(-x_2^2 - 6 x_1), & -1 < x_1 + x_2 \leq 1 \\ 0.5457 \exp(-0.75 x_2^2 - 3.75 x_1^2 + 1.5 x_1), & x_1 + x_2 \leq -1 \end{cases}$$

解：

```
[x,y]=meshgrid(-1:0.04:1,-2:0.04:2);
```

```
z=0.5457*exp(-0.75*y.^2-3.75*x.^2-1.5*x).*(x+y>1)+...
```

```
0.7575*exp(-y.^2-6*x.^2)*((x+y>-1)&(x+y<=1))+...
```

```
0.5457*exp(-0.75*y.^2-3.75*x.^2+1.5*x).*(x+y<=-1);
```

```
h=surf(x,y,z),
```

```
shading flat
```

例 3：绘制上图的等高线

```
[C,h]=contour(x,y,z)
```

```
clabel(C,h)
```

例 4：四维切面图函数 slice

```
[x,y,z]=meshgrid(0:0.1:2);
```

```
V=sqrt(x.^x+y.^((x+y)/2)+z.^((x+y+z)/3));
```

```
slice(x,y,z,V,[1 2],[1,2],[0,1])
```

例 5-1：绘制伪彩色图像 pcolor

```
X=[1 2 3 4 5 6];
```

```
Y=[2 4 8 10 12 14];
```

```
C=rand(6);
```

```
pcolor(X,Y,C)
```

```
colorbar %显示颜色条
```

例 5-2：

```
clear,clc
```

```
n=6;
```

```
r=(0:n)/n;
```

```
theta=pi*(-n:n)/n;
```

```
X=r*cos(theta);
```

```
Y=r*sin(theta);
```

```
C=r*cos(2*theta);
```

```
pcolor(X,Y,C);
```

axis equal tight

例 5-3:

```
pcolor(magic(20))
colormap(gray(2))
axis ij;
axis square
```

例 5-4: 填充函数 **fill3** 用法示例

用指定的颜色填充三维多边形。

fill3(X,Y,Z,C)填充由参数 **X**, **Y**, **Z** 确定的多边形, 若 **X**, **Y** 或 **Z** 为矩阵, **fill3** 生成 **n** 个多边形, 其中 **n** 为矩阵的列数。在必要的时候, **fill3** 会自动连接最后一个节点和第一个节点。以便能形成封闭的多边形。参数 **C** 指定颜色, 这儿 **C** 为引用当前色图的下标向量或矩阵。若 **C** 为行向量, 则 **c** 的维数必须等于 **X** 的列数和 **Y** 的列数, 若 **C** 为列向量, 则 **C** 的维数必须等于矩阵 **X** 的行数和 **Y** 的行数。

fill3(X1,Y1,Z1,C1,X2,Y2,Z2,C2,...) 对多边形的不同区域用不同的颜色进行填充。

fill3(X,Y,Z,ColorSpec) 用指定的颜色 **ColorSpec** 填充由 **x**, **y** 和 **z** 确定的多边形。

注意:(1)若用户对填充的颜色指定为 **ColorSpec**, 则 **fill3** 生成阴影类型为 **flat-shaded** 的多边形, 且设置块 (**patch**) 的属性 **FaceColor** 为 **RGB** 颜色形式的矩阵。(2) 若参数 **C** 为一行向量, 命令 **fill3** 生成带平面阴影 (**flat-shaded**) 的多边形, 同时设置补片对象的面颜色 (**FaceColor**) 属性为 **flat**。向量 **c** 中的每一元素成为每一补片对象的颜色数据 (**CData**) 属性的值。

```
X = 10*rand(4); Y=10*rand(4); Z=10*rand(4);
```

```
C = rand(4);
```

```
fill3(X,Y,Z,C)
```

5.6 颜色和光照设置命令

1、颜色控制命令

(1) **colormap()**: 设置或获取当前色图。色图为一个 **m*3** 的、元素在 0 到 1 之间的实数的矩阵, 每一行为定义一个颜色的 **RGB** 向量。色图矩阵的第 **k** 行定义了第 **k** 个颜色, 其中 **map(k,:)= [r(k) g(k) b(k)]** 指定了组成该颜色中红色、绿色、蓝色的强度。

用法: **colormap(map)** 通过矩阵 **map** 设置色图。。若矩阵 **map** 中的元素不在 [0 1] 区间之内, 则返回一个错误。**map** 的取值有一些默认的取值, 如 **Cool, Bone, Flag, Jet,**

Spring,Autumn,Summer,Winter 等;

(2) 其他命令: bone, cool, pink, copper, flag, gray, hot, hsv, jet, prism;

示例 1:

```
figure
surf(peaks)
colormap winter;
```

示例 2:

```
c = jet(10);
surf(peaks);
colormap(c);
shading interp;
```

2、色图控制命令

(1) brighten(): 增亮或变暗色图;

用法: brighten(beta) 增亮或变暗当前的色图。若 $0 < \text{beta} < 1$, 则增亮色图; 若 $-1 < \text{beta} < 0$, 则变暗色图。改变的色图将代替原来的色图, 但本质上是相同的颜色。

(2) colorbar(): 显示能指定颜色刻度的颜色条。且调整当前坐标轴, 以适应当前的颜色条。

用法: colorbar('vert') 增加一垂直的颜色条到当前的坐标轴。

colorbar('horiz') 增加一水平的颜色条到当前的坐标轴。

示例 1:

```
surf(peaks)
c = colorbar;
c.Label.String = 'Elevation (ft in 1000s)';
```

示例 2:

```
contourf(peaks)
colorbar('Ticks',[-5,-2,1,4,7],...
        'TickLabels',{'Cold','Cool','Neutral','Warm','Hot'})
```

(3) contrast(): 提高灰度色图的对比度。该命令可以增强图像的对比度。

用法: cmap = contrast(X) 返回一灰度色图, 该色图与当前色图有相同的维数。参量 cmap 为生成的灰度色图。cmap = contrast(X,m) 返回维数为 $m \times 3$ 的灰度色图 cmap。

示例 1:

```
load clown;
```

```
cmap = contrast(X);
```

```
image(X);
```

```
colormap(cmap);
```

(4) 其他命令: `rgbplot` (色图命令), `diffuse` (漫反射率), `specular` (镜面反射率) 等

5.7 动态可视化图形

Matlab 中的动画命令, **moviein**, **getframe**, **drawnow** 和 **movie**, 用 `getframe` 把 Matlab 产生的图形存储下来, 每个图形成一个很大的列向量; 再用 `N` 行这样的列保存 `N` 幅图, 成为一个大矩阵; `movie` 命令把他们连接起来重放, 产生动画效果; `moviein` 用来预留存储空间以加快运行的速度。有些情况也需要借助 **pause** 命令和循环语句来实现点轨迹的动态演示。绘制彗星图的命令 `comet` 或者 `comet3` 也可以实现点的运动轨迹, 不过这种方式速度较快。

示例 1:

```
axis equal, %把坐标设成相等比例
```

```
M=moviein(16);%为变量 M 预留 16 幅图的存储空间
```

```
for j=1:16
```

```
    plot(fft(eye(j+16)));
```

```
    M(:,j)=getframe; %返回当前坐标轴下的一帧图像
```

```
end
```

```
movie(M,30) %以每秒 30 帧的速度播放 M 中的图形
```

示例 2: 使用 `comet/comet3` 绘制彗星图

(1)

```
t = 0:.01:2*pi;
```

```
x = cos(2*t).*(cos(t).^2);
```

```
y = sin(2*t).*(sin(t).^2);
```

```
comet(x,y);
```

(2)

```
n=10;
```

```
t=n*pi*(0:0.0005:1);
```

```
x=sin(t);
```

```
y=cos(t);
```

```
plot(x,y,'g');
```

```
axis square;
```

```
hold on;
```

```
comet(x,y,0.01);
```

```
hold off;
```

(3)

```
t = -20*pi:pi/50:20*pi;
```

```
comet3((cos(2*t).^2).*sin(t),(sin(2*t).^2).*cos(t),t);
```

示例 3: 极坐标下一个点沿着图形移动的轨迹

```
a=0:.01:2*pi;
```

```
b=3;
```

```
polar(a,b*(1-cos(a)), 'r');
```

```
hold on;
```

```
for a=0:0.05:2*pi
```

cla(findobj(gca,'color','r')); %findobj 函数用来使用特定的属性获取图形句柄, cla 用来清理由函数句柄指定的图形所在的坐标轴。

```
h2=polar(a,b*(1-cos(a)), 'b');
```

```
set(h2,'linestyle','.', 'markersize',30); %设置句柄指定函数图形的属性
```

```
pause(0.001)
```

```
end
```

示例 4: 普通二维坐标下一个点沿着曲线移动的轨迹, 程序与上面的结果基本相同。

```
x=-2*pi:0.1:2*pi;
```

```
y=sin(x);
```

```
plot(x,y, 'b-');
```

```
hold on
```

```
for m=-2*pi:0.05:2*pi
```

```
cla(findobj(gca,'color','b')); %cla 清除当前坐标轴上面的子对象
```

```
h=plot(m,sin(m), 'r');
```

```
set(h,'LineStyle','.', 'Marker','*', 'MarkerSize',20);
```

```
pause(0.001); %此处可以改为 drawnow, 不过速度不能控制
```

```
end
```

示例 5: 普通三维坐标下一个点沿着曲线移动的轨迹。

解:

% 三维螺旋线坐标

```
t1=10*pi*(0:1000)/1000;
```

```
x1=cos(t1);
```

```
y1=sin(t1);
```

```
z1=-t1;
```

%绘制曲线下面的水平直线

```
t2=(0:10)/10;
```

```
x2=x1(end)*(1-t2);
```

```
y2=y1(end)*(1-t2);
```

```
z2=z1(end)*ones(size(x2));
```

%绘制垂直直线

```
t3=t2;
```

```
z3=(1-t3)*z1(end);
```

```
x3=zeros(size(z3));
```

```

y3=x3;
%绘制曲线上面的水平直线
t4=t2;
x4=t4;
y4=zeros(size(x4));
z4=y4;
%绘制曲线
x=[x1,x2,x3,x4];
y=[y1,y2,y3,y4];
z=[z1,z2,z3,z4];
plot3(x,y,z,'b','linewidth',3)
axis off
%绘制移动的点
h=line('Color',[1 0 0],'Marker','.', 'MarkerSize',40,'EraseMode','xor');
n=length(x);
i=1;j=1;
while 1
    set(h,'xdata',x(i),'ydata',y(i),'zdata',z(i));
    drawnow %使 Matlab 暂停目前的任务序列而去刷新屏幕
    pause(0.0005);
    f=getframe(gcf);
    i=i+1;
    j=j+1;
    if i>n & j>450
        break;
    end
end
end
示例 6:
x=0:0.01:2*pi;
y=sin(x);
z=cos(x);
h=plot(y,z,'b-');
axis([-2 2 -2 2]);
hold on;
axis square;
for k=0:0.01:2*pi
    x=sin(k);
    y=cos(k);
    plot(x,y,'r*');
    drawnow;
end

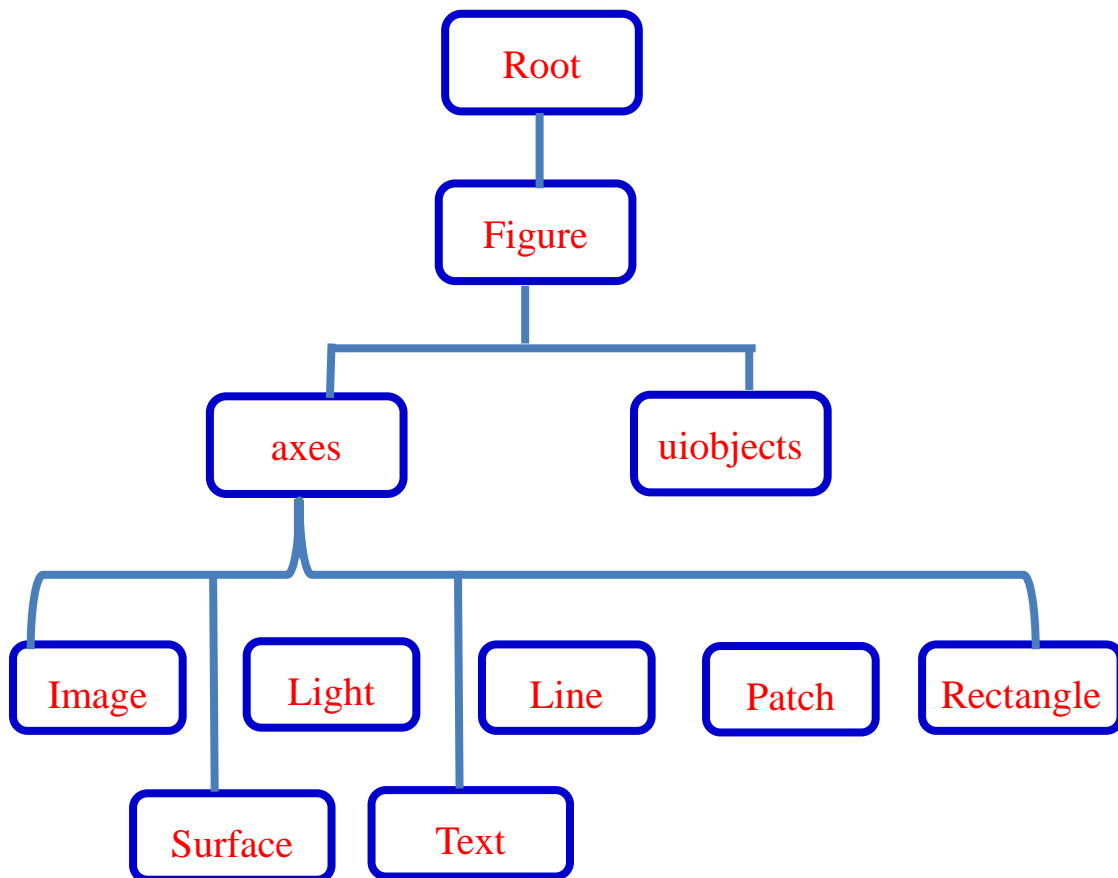
```


5.8 Matlab 的图形句柄

1、Matlab 图形系统

Matlab 的图形系统是一个分层的图形对象系统，每一个对象都对应一个唯一的句柄，每一个图形对象都对应着称为属性的特殊数据，通过这些属性，可以修改对象的行形态。例如 line 的属性有 x-data, y-data, color, line style, line width, marker type 等。所有的图形对象以父对象和子对象的形式分层，子对象会继承父对象的属性。Matlab 中的最高层是 root 层，可以认为是整个计算机屏幕，root 对象是随着 Matlab 的启动而自动创建的，root 下面是一个或者多个 Figure，每个 Figure 都是一个独立的窗口，用来显示图像数据，每一个 Figure 有自己的属性，包括 color, color map, paper size, paper orientation, pointer type 等；

每一个 Figure 包含 4 种类型的对象：**uimenu**, **uicontextmenu**, **uicontrols**, **axes**。前三个是用来创建图形用户界面的特殊图形，统称为 uiobjects, axes 是坐标轴。每个 axes 又包含有七种类型的对象，层次结构如下图所示：



2、图形对象的句柄

每一个图形的句柄都是唯一的，称为 **handle**，获取的方法如下所示：

`hndl=figure`; `root` 对象的句柄总是 0，`figure` 对象的句柄是非负整数，而其他图形对象是任意的实数。

Matlab 中获取图形对象句柄的方法主要是：`gcf` 返回当前选择**图像**的句柄，`gco` 返回当前选择**对象**的句柄；

用来设置和获取对象属性值的方法分别是 `set` 和 `get`，调用方法是：

`value=get(handle,'PropertyName');` %获得指定属性的值

`value=get(handle);` %返回值是一个结构体，包含全部属性信息。

`set(handle,'PropertyName',value1,...);`

对于普通用户而言，属性的修改也可以采用属性编辑框进行。调用命令为 `propedit(Handle)`（注意：需要首先使用 `figure` 命令将图形绘制在 `figure` 中才可以使用该命令）

`set` 函数也可以用来提供所有可能的固定的属性值，调用方法是：

`set(handle,'ProperName')`或者 `set(handle)`

3、用户自定义数据

用户可以自定义和图形对象相关联的附加数据，使用的函数为 `getappdata` 和 `setappdata`，调用方法：

`setappdata(Hndl,'DataName',DataValue)`

`value=getappdata(Hndl,'DataName')`

`struct=getappdata(Hndl)`

示例：`setappdata(Hndl,'ErrorCount',0)`，`setappdata(Hndl,'LastError','No Error');`

4、图形对象定位

如果因为某些原因不能直接得到 `handle` 的时候，Matlab 提供了四种特殊的方法来获取对象的句柄，分别是：

- `gcf`：返回当前 `figure` 的句柄，如果不存在，则创建一个 `Figure`；
- `gca`：返回在当前 `figure` 上的 `axes` 的句柄；
- `gco`：返回当前对象的句柄，如果不存在 `figure` 或者不存在 `axes`，则创建一个 `axes`；
- `findobj`：寻找带有特定属性值的图形对象；

调用方法:

```
H_fig=gcf;
```

```
H_axe=gca;
```

```
H_obj=gco;
```

```
H_obj=gco(H_fig);
```

```
type=get(H_obj,'Type'); %Type 可以是 figure, line, text 等
```

Hndls=findobj('PropertyName',valule1,...); 从 root 开始查找整个层次树中符合这个属性值得所有对象的句柄, 当然也可以指定具体的 Figure 来进行查找对象的句柄:

```
Hndls=findobj(SrcHandle,'PropertyName',value1,...)
```

示例: Hndls=findobj(1,'Type','line','LineStyle','--');

对象的选择也可以使用鼠标, 对应的函数式: k=waitforbuttonpress, 这个函数执行时会中断程序等待用户鼠标点击, 如果用户点击之后, 需要使用 gco 命令获取点击对象的句柄: Hndl=gco;

图形对象的应用:

示例 1: 选择图形对象。选择图形上面的对象并获取属性。

```
x=-2*pi:pi/10:3*pi;
y1=sin(x);
y2=cos(x);
H1=plot(x,y1);
set(H1,'LineWidth',2);
hold on;
H2=plot(x,y2);
set(H2,'LineWidth',2,'LineStyle',':','Color','r');
title('\bfPlot of sin\itx \rm\bf and cos \itx');
xlabel('\bf\itx');
ylabel('\bf\itx \rm\bf and cos \itx');
legend('sine','cosine');
hold off
% Now set up a loop and wait for a mouse click
k=waitforbuttonpress;
while k==0
    Handle=gco;
    type=get(Handle,'Type');
    disp(['Object type=' type '']);
    yn=input('Do you want to display details?[y/n]','s');
    if yn=='y'
        details=get(Handle);
```

```

        disp(details);
    end
    k=waitforbuttonpress;
end

```

示例 2： 图形对象的位置属性设置。

```

x=-2*pi:pi/10:2*pi;
y1=sin(x);
y2=cos(x);
figure;
%Create the first set of axes and plot sin(x)
Ha1=axes('Position',[.05 .05 .5 .5]);
H1=plot(x,y1);
set(H1,'LineWidth',2);
title('\bfPlot of sin \itx');
xlabel('\bf\itx');
ylabel('\bfsin \itx');
axis([-8 8 -1 1]);
%Create the second set of axes and plot cos(x)
Ha2=axes('Position',[.45 .45 .5 .5]);
H2=plot(x,y2);
set(H2,'LineWidth',2,'Color','r','LineStyle','--');
title('\bfPlot of cos \itx');
xlabel('\bf\itx');
ylabel('\bfcos \itx');
axis([-8 8 -1 1]);
%Create a text string attached to the line on the axes
axes(Ha1);
text(-pi,0.0,'sin(x)\rightarrow','HorizontalAlignment','right');
axes(Ha2);
text(-7.5,-0.9,'Test String 2');

```

5.9 图形用户接口（Graphics User Interface）

Matlab 提供了利用 GUI 进行可视化编程的方法。GUI 提供了一个编程环境，在这个环境里面可以使用 `pushbutton`，`togglebuttons`，`lists`，`menus`，`text boxes` 等控件编程。GUI 编程是基于用户点击鼠标产生输入事件，然后程序会对这个事件产生响应。因此 Matlab 的 GUI 编程是事件驱动的。Matlab 使用 GUI 的三个基本元素：

（1）Components 组件（控件），使用 `uicontrol` 函数创建控件，`uimenu` 和 `uicontextmenu` 创建菜单，坐标轴使用 `axes` 函数创建；

（2）Figures 图形界面，用来放置各种组件；

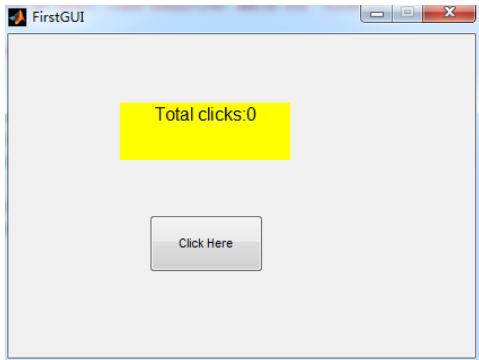
(3) Callbacks 回调函数，用来响应产生的各种事件，需要编程者在此实现编写程序。

Matlab 中使用 **uicontrol** 创建的控件有：pushbutton, Toggle button, Radio button, Check box, Edit box, List box, Popup menus, Slider, Frame, Text Field。

1、创建 GUI 的基本步骤：

- (1) 选择需要哪些元素以及元素实现的函数，并给出组件的布局；
- (2) 调用 `guide` 命令；
- (3) 调用 **Property Inspector** 给每个元素命名 (tag)，设置每个元素的属性特征；
- (4) 将图形界面保存到一个文件中，保存的同时，会自动产生两个文件，.fig 文件和.m 文件，.m 文件中提供了回调函数的框架；
- (5) 在回调函数的框架中编写相应的代码。

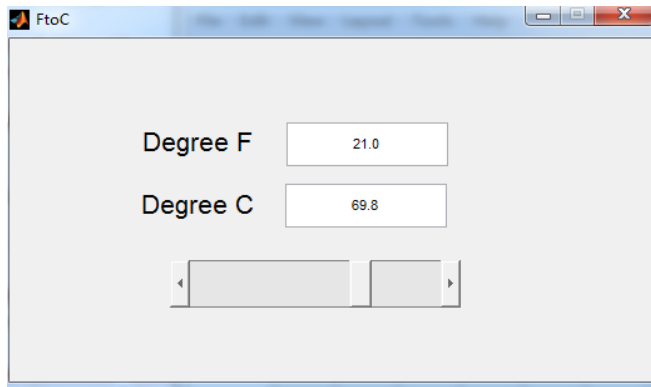
示例 1：简单计数器（按钮和静态文本框的使用）



按钮的 callback 函数的主要代码：

```
persistent count; %永久变量，类似静态变量的作用
if isempty(count)
    count=0;
end
count=count+1;
str=sprintf('Total Clicks:%d',count);
set(handles.txtCount,'String',str);
```

示例 2：温度转换（编辑框和滑动条的使用）



滚动条的 min 和 max 分别设置为 0 和 100，温度转换 $F=9/5(C)+32$ 和 $C=5/9(F-32)$

editF 的 Callback 函数：

```
deg_f=str2num(get(hObject,'String'));
deg_f=max([32 deg_f]);
deg_f=min([212 deg_f]);
deg_c=(5/9)*(deg_f-32);

set(handles.editF,'String',sprintf('%.1f',deg_f));
set(handles.editC,'String',sprintf('%.1f',deg_c));
set(handles.slider1,'Value',deg_c);
```

editC 的 Callback：

```
deg_c=str2num(get(hObject,'String'));
deg_c=max([0 deg_c]);
deg_c=min([100 deg_c]);
deg_f=(9/5)*deg_c+32;;

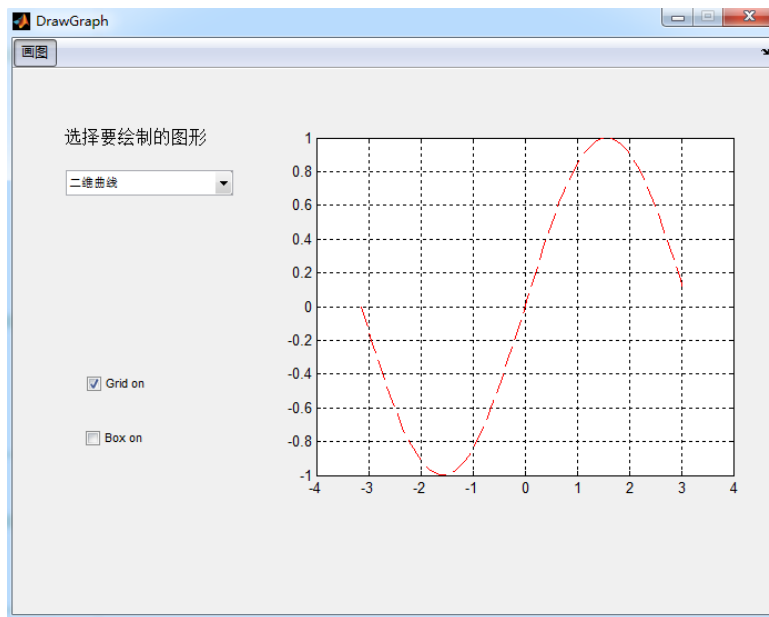
set(handles.editF,'String',sprintf('%.1f',deg_f));
set(handles.editC,'String',sprintf('%.1f',deg_c));
set(handles.slider1,'Value',deg_c);
```

slider 的 Callback：

```
deg_c=get(hObject,'Value');
deg_f=to_c(deg_c);

set(handles.editF,'String',sprintf('%.1f',deg_f));
set(handles.editC,'String',sprintf('%.1f',deg_c));
set(handles.slider1,'Value',deg_c);
```

示例三：图形绘制（坐标系、复选框和弹出式组合框的使用）



初始化弹出式组合框的代码（在 `createFcn` 函数中）：

```
set(hObject,'String','二维曲线|三维曲线|三维曲面');
```

弹出式组合框的 Callback(部分)：

```
selectValue=get(hObject,'Value');
if selectValue==1
    x=-pi:0.1:pi;
    plot(handles.axes1,x,sin(x),'ro');
end
```

复选框 `Grid on` 的代码：

```
value=get(hObject,'Value')
if value==1
    set(handles.axes1,'XGrid','on','YGrid','on');
end
```

6、Matlab 在数值模拟中的几个应用

6.1 蒙特卡洛方法

蒙特卡洛方法也称为**计算机随机模拟方法**，它源于世界著名的赌城—摩纳哥的 Monte Carlo（蒙特卡洛）。其雏形是 Buffon 随机投针实验。它是基于对大量事件的统计结果来实现一些确定性问题的计算。尤其适用于模拟随时间变化的随机过程。使用**蒙特卡洛方法必须使用计算机生成相关分布的随机数**，Matlab 给出了生成各种随机数的命令。

示例 1： π 值计算的蒙特卡洛模拟

（1）首先绘制一个长宽各为 1 的正方体内画一个以坐标原点为圆心，半径为 1 的 $1/4$ 圆。

绘图程序如下：

```
clc;
x=0:0.01:1;
y=sqrt(1-x.^2);
figure;
h=plot(x,y);
active.linestyle='-';
active.linewidth=3;
active.color='k';
set(h,active);
set(gcf,'color','y');
set(gca,'color','b');
xlabel('x');
ylabel('y');
grid on;
axis square;
```

（2）因为正方形的面积为 1, $1/4$ 圆和坐标轴围成的区域面积为 $\pi/4$ ，即它们的比值为 $\pi/4$ ，下面用计算机产生 $[0, 1]$ 区间内的随机数，并计算 π 值，程序如下：

```
clc;
clear all;
n=1000000;
a=rand(n,1);
%必须用 rand 指令而不能用 randn 指令因为产生的随机数必须是均匀的
b=rand(n,1);
c=find(a.^2+b.^2<=1);
d=length(c);
```


pi_value=d/n*4

示例 2: $y = x^2, y = 12 - x$ 与 x 轴在第一象限围成一个曲边三角形。设计一个随机实验，求该图形面积的近似值。

解 设计的随机试验的思想如下，在矩形区域 $[0,12] \times [0,9]$ 上产生服从均匀分布的 10^7 个随机点，统计随机点落在曲边三角形的频数，则曲边三角形的面积近似为上述矩形的面积乘以频率。

计算的 Matlab 程序如下

```
clc, clear
x=unifrnd(0,12,[1,10000000]);
y=unifrnd(0,9,[1,10000000]);
pinshu=sum(y<x.^2 & x<=3)+sum(y<12-x & x>=3);
area_appr=12*9*pinshu/10^7
```

运行结果在 49.5 附近，由于是随机模拟，每次的结果都是不一样的。

蒙特卡洛方法的其他应用有：

1、单重积分计算方法

设区间 (a,b) 中的随机变量 ξ 的概率密度函数为 $f_{\xi}(x)$ ， $g(x)$ 是区间 (a,b) 上的连续函数，数学期望

$$E[g(\xi)] = \int_a^b g(x)f_{\xi}(x)dx \quad (12)$$

存在，则 (12) 式的积分可用下面方法近似计算。

设随机变量 ξ 的一系列可取值为 x_1, x_2, \dots ，由 $y_i = f(x_i)$ 形成的随机变量 $\eta = g(\xi)$ 的可能取值的数列为 y_1, y_2, \dots 。显然由此产生的随机抽样 y_i 是相互独立的，且具有相同分布，若其数学期望为 M ，则根据大数定理有

$$\lim_{n \rightarrow \infty} P\left(\left|\frac{1}{n} \sum_{i=1}^n y_i - M\right| < \varepsilon\right) = 1,$$

可见，只要 n 充分大，积分 (12) 式有近似值：

$$\bar{M} = \frac{1}{n} \sum_{i=1}^n g(x_i). \quad (13)$$

下面讨论用上述方法计算积分：

$$J = \int_a^b h(x)dx \quad (14)$$

的值。

为此，选择某种密度函数 $f(x)$ 满足

$$\int_a^b f(x)dx = 1,$$

且能很方便地生成密度函数为 $f(x)$ 的随机抽样。同时将积分写成如下形式：

$$J = \int_a^b \frac{h(x)}{f(x)} \cdot f(x)dx = \int_a^b g(x)f(x)dx.$$

在很多情况下，往往取 $f(x)$ 为区间 (a,b) 上均匀分布的概率密度函数：

$$f(x) = \begin{cases} \frac{1}{b-a}, & x \in (a,b), \\ 0, & \text{其它}. \end{cases}$$

$$\text{这样, } J = (b-a) \int_a^b h(x) \frac{1}{b-a} dx.$$

现在在区间 (a,b) 上均匀分布的随机数总体中选取 x_i ，对每个 x_i 计算 $h(x_i)$ 的值，然

后计算平均值： $\bar{M} = \frac{1}{n} \sum_{i=1}^n h(x_i)$ 于是积分（14）式的值可近似地取为

$$J \approx (b-a)\bar{M} = \frac{b-a}{n} \sum_{i=1}^n h_i(x).$$

示例 3： 计算积分： $\int_{-1}^1 \frac{x dx}{\sqrt{5-4x}}.$

解 随机模拟时取区间 $(-1, 1)$ 上的均匀分布，其概率密度函数：

$$f(x) = \begin{cases} \frac{1}{2}, & x \in (-1,1) \\ 0, & \text{其它} \end{cases}$$

随机模拟的 Matlab 程序如下

```
clc, clear
y=@(x) x./sqrt(5-4*x); %定义被积函数
n=1000000; %生成随机数的个数
x=unifrnd(-1,1,[1,n]); %生成 n 个区间 (-1,1) 上均匀分布的随机数
h=y(x); %计算被积函数的一系列取值
junzhi=sum(h)/n; %计算取值的平均值
```

jifen=2*junzhi %计算积分的近似值

2、多重积分计算

假设要求多重积分

$$J = \int \cdots \int_{\Omega} f(x_1, \cdots, x_n) dx_1 \cdots dx_n \quad (15)$$

的值。积分区域 Ω 是有界区域，被积函数 f 在区域 Ω 中是有界的。

设 $g(x_1, \cdots, x_n)$ 为区域 Ω 上的概率密度函数，且当 $f(x_1, \cdots, x_n) \neq 0$ 时 $g(x_1, \cdots, x_n)$ 亦不为零。令

$$h(x_1, \cdots, x_n) = \begin{cases} \frac{f(x_1, \cdots, x_n)}{g(x_1, \cdots, x_n)}, & \text{当 } g(x_1, \cdots, x_n) \neq 0 \text{ 时,} \\ 0, & \text{当 } g(x_1, \cdots, x_n) = 0 \text{ 时.} \end{cases}$$

则积分(15)可改写为

$$J = \int \cdots \int_{\Omega} h(x_1, \cdots, x_n) g(x_1, \cdots, x_n) dx_1 \cdots dx_n .$$

若 (X_1, \cdots, X_n) 是 n 维空间区域 Ω 中的随机变量，概率密度函数为 $g(x_1, \cdots, x_n)$ ，则随机变量 $h(X_1, \cdots, X_n)$ 的数学期望为

$$E[h(X_1, \cdots, X_n)] = \int \cdots \int_{\Omega} h(x_1, \cdots, x_n) g(x_1, \cdots, x_n) dx_1 \cdots dx_n = J .$$

即积分 J 是随机变量 $h(X_1, \cdots, X_n)$ 的数学期望。如果选取 N 个点 $P_i(x_1^i, \cdots, x_n^i)$ ($i=1, \cdots, N$)服从分布 $g(x_1, \cdots, x_n)$ ，则根据大数定理，其算术平均值

$$\bar{J} = \frac{1}{N} \sum_{i=1}^N h(x_1^i, \cdots, x_n^i)$$

即为积分 J 的近似值。通常可选取 $g(x_1, \cdots, x_n)$ 为 Ω 上的均匀分布

$$g(x_1, \cdots, x_n) = \begin{cases} \frac{1}{V}, & (x_1, \cdots, x_n) \in \Omega, \\ 0, & \text{其它.} \end{cases}$$

其中 V 表示区域 Ω 的体积，则 $h(x_1, \cdots, x_n) = V \cdot f(x_1, \cdots, x_n)$

积分(15)的近似值可取为

$$\bar{J} = \frac{V}{N} \sum_{i=1}^N f(x_1^i, \dots, x_n^i).$$

示例 4: 计算 $I = \iiint_{\Omega} (x+y+z)^2 dx dy dz$, 其中 Ω 为 $z \geq x^2 + y^2$ 与

$x^2 + y^2 + z^2 \leq 2$ 所围成的区域。

解 随机模拟时首先要计算 Ω 的体积, 设 Ω 的体积为 V , 区域 Ω 上均匀分布的密度

函数为:
$$f(x, y, z) = \begin{cases} \frac{1}{V}, & (x, y, z) \in V, \\ 0, & \text{其它}. \end{cases}$$

求 Ω 的体积 V 时, 在立体区域 $[-1,1] \times [-1,1] \times [0, \sqrt{2}]$ 上产生服从均匀分布的 10^6 个随机点, 统计随机点落在 Ω 的频数, 则 Ω 的体积 V 近似为上述立体的体积乘以频率。

仿真一的程序如下

```
clc, clear
h=@(x,y,z) (x+y+z).^2; %定义被积函数
n=1000000; %生成随机数的个数
x=unifrnd(-1,1,[1,n]); %生成 n 个区间 (-1,1) 上均匀分布的随机数
y=unifrnd(-1,1,[1,n]);
z=unifrnd(0,sqrt(2),[1,n]);
f=sum(z>=x.^2+y.^2 & x.^2+y.^2+z.^2<=2); %计算落在区域 V 上的频数
V=f/n*4*sqrt(2) %计算体积
hh=h(x,y,z); %计算被积函数一系列的取值
jifen=V*sum(hh.*(z>=x.^2+y.^2 & x.^2+y.^2+z.^2<=2))/f
```

仿真二的程序如下 (为了说明和所取的积分区间的无关性)

```
clc, clear
h=@(x,y,z) (x+y+z).^2; %定义被积函数
n=1000000; %生成随机数的个数
x=unifrnd(-sqrt(2),sqrt(2),[1,n]); %生成 n 个区间 (-1,1) 上均匀分布的随机数
y=unifrnd(-sqrt(2),sqrt(2),[1,n]);
z=unifrnd(-sqrt(2),sqrt(2),[1,n]);
f=sum(z>=x.^2+y.^2 & x.^2+y.^2+z.^2<=2); %计算落在区域 V 上的频数
V=f/n*16 * sqrt(2) %计算体积
hh=h(x,y,z); %计算被积函数一系列的取值
jifen=V*sum(hh.*(z>=x.^2+y.^2 & x.^2+y.^2+z.^2<=2))/f
```

3、非线性整数规划的随机模拟

示例 5: 已知非线性整数规划为

$$\begin{aligned} \max \quad & z = x_1^2 + x_2^2 + 3x_3^2 + 4x_4^2 + 2x_5^2 - 8x_1 - 2x_2 - 3x_3 - x_4 - 2x_5, \\ \text{s.t.} \quad & \begin{cases} 0 \leq x_i \leq 99, & (i = 1, \dots, 5), \\ x_1 + x_2 + x_3 + x_4 + x_5 \leq 400, \\ x_1 + 2x_2 + 2x_3 + x_4 + 6x_5 \leq 800, \\ 2x_1 + x_2 + 6x_3 \leq 200, \\ x_3 + x_4 + 5x_5 \leq 200. \end{cases} \end{aligned}$$

解 (1) 首先编写 M 文件 mente.m 定义目标函数 f 和约束向量函数 g，程序如下

```
function [f,g]=mengte(x);
f=x(1)^2+x(2)^2+3*x(3)^2+4*x(4)^2+2*x(5)-8*x(1)-2*x(2)-3*x(3)-...
x(4)-2*x(5);
g=[sum(x)-400
x(1)+2*x(2)+2*x(3)+x(4)+6*x(5)-800
2*x(1)+x(2)+6*x(3)-200
x(3)+x(4)+5*x(5)-200];
```

(2) 编写如下 Matlab 程序求问题的解。

```
rand('state',sum(clock)); %初始化随机数发生器
p0=0;
tic %计时开始
for i=1:10^6
x=randint(1,5,[1,99]); %产生一行五列的区间[1,99]上的随机整数
[f,g]=mengte(x);
if all(g<=0)
if p0<f
x0=x; p0=f; %记录下当前较好的解
end
end
end
x0, p0
toc %计时结束
```

6.2 微分方程组的数值模拟

使用数值模拟微分方程组主要从两个角度考虑：(1) 有些微分方程组很难求出解析解或者没有解析解，只能使用数值解；(2) 可以考察模型未来一段时间的状况，如衰减、震荡或混沌等现象。

示例 1：某湖泊中有机物新城代谢系统模型的状态方程组如下所示：

$$\begin{cases} X_s = 95.9(1 + 0.635 \sin(2\pi t)) \\ \frac{dX_p}{dt} = X_s - 4.03X_p \\ \frac{dX_h}{dt} = 0.48X_p - 17.87X_h \\ \frac{dX_r}{dt} = 4.85X_h - 4.65X_r \\ \frac{dX_o}{dt} = 2.55X_p + 6.12X_h + 1.95X_r \\ \frac{dX_e}{dt} = 1.10X_p + 6.90X_h + 2.70X_r \end{cases}$$

其中, $X_s(0) = 95.9, X_p(0) = 0.83, X_h(0) = 0.003,$
 $X_r(0) = 0.0001, X_o(0) = 0.0, X_e(0) = 0.0$

时间 t 是以年为单位, X_s 表示 t 时刻太阳提供的能量, X_p 表示 t 时刻植物生长的数量, X_h 表示吞食植物的虫类生成数量; X_r 为 t 时刻食虫植物的生长数量; X_o 表示 t 时刻湖底有机物的沉淀量; X_e 表示 t 时刻已扩散到周围环境的总能量。使用 Matlab 的 ode 命令求解并模拟 1900 年到 2020 年该湖泊每隔 10 年有机物新城代谢情况, Matlab 程序如下所示:

```
%微分方程组求解主程序
clc;clear all;clf;close all;
%Windows 时钟自动计时
T1=clock;%Clock 函数返回的值是 clock = [year month day hour minute seconds]
disp('计算机正在准备输出湖泊有机物新陈代谢结果, 请耐心等待……');
[tt,y]=ode45('lbwfun',[0:10:2020],[95.9,0.83,0.003,0.0001,0.0,0.0]);
t=tt(191:end,:);
ys=y(191:end,1)
yp=y(191:end,2)
yh=y(191:end,3)
yr=y(191:end,4)
yo=y(191:end,5)
ye=y(191:end,6)
T2=clock;
API_elapsed_time=T2-T1;
if API_elapsed_time(6)<0
    API_elapsed_time(6)=API_elapsed_time(6)+60;
    API_elapsed_time(5)=API_elapsed_time(5)-1;
end
if API_elapsed_time(5)<0
```

```

        API_elapsed_time(5)=API_elapsed_time(5)+60;
        API_elapsed_time(4)=API_elapsed_time(4)-1;
    end
    if API_elapsed_time(4)<0
        API_elapsed_time(4)=API_elapsed_time(4)+60;
        API_elapsed_time(3)=API_elapsed_time(4)-1;
    end
    str=sprintf('湖泊新陈代谢模拟程序共运行 %d 小时 %d 分钟 %.4f 秒',...
        API_elapsed_time(4),API_elapsed_time(5),API_elapsed_time(6));
    disp(str);
    微分方程组的子函数：
    function ydot=lbwfun(t,y);
    ydot=[121.793*pi*cos(2*pi*t);
        y(1)-4.03*y(2);
        0.48*y(2)-17.87*y(3);
        4.85*y(3)-4.65*y(4);
        2.55*y(2)+6.12*y(3)+1.95*y(4);
        1.10*y(2)+6.90*y(3)+2.70*y(4)];
    end

```

6.3 服从概率分布的随机模拟

1、模型假设：

- (1) 顾客到达收银台的时间间隔服从负指数分布；
- (2) 收银系统连续工作性能仍然稳定；
- (3) 超市的所有收银台均是相互独立的，各个收银台之间不会因为忙碌的程度不同导致顾客的流动，即外界环境对收银台没有扰动。

2、模型组建

以超市的某一收银台为研究对象，顾客到达收银台的时间间隔服从平均时间为 10s 的负指数分布，分布函数为：

$$f(x) = \begin{cases} \frac{1}{\lambda} e^{-\frac{\lambda}{x}}, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

每个顾客的服务时间服从均值为 6.5s，标准差为 1.2s 的正态分布，使用计算机模拟顾客在收银台的平均逗留时间和收银系统的服务强度（服务占总时间的比率）。

设第 i 个人到达时刻为 a_i ，开始接受服务的时刻为 b_i ，离开时刻为 c_i ，设总共考虑 n 个顾客， n 个顾客到达的时间间隔为服从均值为 10s 的负指数分布序列 $\{dt(n)\}$ ，每个人

接受服务的时间服从正态分布 $N(6.5, 1.2 \times 1.2)$ 的序列 $\{st(n)\}$ ，每个人的到达时刻计算公式为：

$$a_1=0, a_i=a_{i-1}+dt_{i-1}, \quad i=2,3,\dots,n$$

第一个顾客接受服务的时刻为 $b_1=0$ ，第 1 个顾客离开的时刻为 $c_1=st_1$ ，第 i 个顾客开始接受服务的时刻为：

$$b_i = \begin{cases} a_i, & a_i > c_{i-1} \\ c_{i-1}, & a_i \leq c_{i-1} \end{cases}, \quad \text{说明当第 } i \text{ 个顾客到达时间比第 } i-1 \text{ 个顾客离开时间早，则开始}$$

接受服务的时间为第 $i-1$ 个顾客离开的时间；当第 i 个顾客到达时间比第 $i-1$ 个顾客离开的时间完，则开始接受服务的时间就是到达时间，第 i 个顾客离开的时间为：

$$c_i=b_i+st_i, \quad i=2,3,\dots,n$$

每个人在收银台的逗留时间为 $wt_i=c_i-a_i$ ， $i=2,3,\dots,n$ ，到第 n 个人离开时刻为 $T=c_n$ ，则系统的工作强度占总时间的比值为：

$$P = \frac{\sum_{i=1}^n st_i}{T-0} \cdot \frac{\sum_{i=1}^n st_i}{T}$$

3、模型求解，分别计算 $n=10, 100, 500$ 等 10 种不同的情况，模拟收银系统的工作强度和顾客平均逗留时间，每种情况都模拟 100 次以便尽量减少随机因素，程序如下所示：

```
clc;close all;clear all;
disp('计算机大概需要两分钟的运行时间，请等待……');
p=zeros(10,100);avert=zeros(10,100);
%分别在顾客人数为 10、100、500 等情况时，模拟系统工作强度和顾客平均逗留时间
nn=[10 100 500 1000 5000 10000 20000 50000 100000 500000];
for d=1:10 %length(nn)=10
    for s=1:100 %每种情况重复模拟 100 次以便消除随机因素
        n=nn(d);%模拟顾客数目
        dt=exprnd(10,1,n);%到达时间间隔
        st=normrnd(6.5,1.2,1,n);%服务台服务时间
        a=zeros(1,n);%每个顾客到达时刻
        b=zeros(1,n);%每个顾客开始接受服务时刻
        c=zeros(1,n);%每个顾客离开时刻
        a(1)=0;
        for i=2:n
            a(i)=a(i-1)+dt(i-1);%第 i 顾客到达时刻
        end
        b(1)=0;%第一个顾客开始接受服务的时刻是其到达的时刻
```



```

        c(1)=b(1)+st(1);%第一个顾客的离开时刻为其接受的服务时间加上开始接受服务的时刻
        for i=2:n
            %如果第 i 个顾客到达时间比前一个顾客离开时间早，则接受服务时间为前一人离开时间
            if(a(i)<=c(i-1))b(i)=c(i-1);
            %如果第 i 个顾客到达时间比前一个顾客离开时间晚，则接受服务时间为其到达时间
            else b(i)=a(i);
            end
            %第 i 个顾客离开时间为其开始接受服务的时刻+接受服务的时间长度
            c(i)=b(i)+st(i);
            end
            cost=zeros(1,n);%记录每个顾客在系统逗留时间
            for i=1:n
                cost(i)=c(i)-a(i);%第 i 个顾客在系统逗留时间
            end
            T=c(n);%总时间
            p(d,s)=sum(st)/T;
            avert(d,s)=sum(cost)/n;
        end
    end
    pc=sum(p)/100;avertc=sum(avert)/100;
    figure(1);subplot(2,1,1);%分区画图
    plot(pc,'color','g','linestyle','-','linewidth',2.5,'marker','*','markersize',5);
    text(1,pc(:,1),texlabel('10 人'),'fontsize',11);
    text(2,pc(:,2),texlabel('10^2 人'),'fontsize',11);
    text(3-0.15,pc(:,3)-0.004,texlabel('5x10^2 人'),'fontsize',11);
    text(4-0.15,pc(:,4)-0.004,texlabel('10^3 人'),'fontsize',11);
    text(5-0.15,pc(:,5)-0.004,texlabel('5x10^3 人'),'fontsize',11);
    text(6-0.15,pc(:,6)-0.004,texlabel('10^4 人'),'fontsize',11);
    text(7-0.15,pc(:,7)-0.004,texlabel('2x10^4 人'),'fontsize',11);
    text(8-0.15,pc(:,8)-0.004,texlabel('5x10^4 人'),'fontsize',11);
    text(9-0.15,pc(:,9)-0.004,texlabel('10^5 人'),'fontsize',11);
    text(10-0.15,pc(:,10)-0.004,texlabel('5x10^5 人'),'fontsize',11);
    xlim([1 11]);xlabel('顾客数量/个','fontsize',11);ylabel('系统工作强度','fontsize',11);
    subplot(2,1,2);
    plot(avertc,'color','r','linestyle','-','linewidth',2.5,'marker','s','markersize',5);
    text(1,avertc(:,1)+0.4,texlabel('10 人'),'fontsize',11);
    text(2-0.15,avertc(:,2)-0.4,texlabel('10^2 人'),'fontsize',11);
    text(3-0.15,avertc(:,3)-0.4,texlabel('5x10^2 人'),'fontsize',11);
    text(4-0.15,avertc(:,4)-0.4,texlabel('10^3 人'),'fontsize',11);
    text(5-0.15,avertc(:,5)-0.4,texlabel('5x10^3 人'),'fontsize',11);
    text(6-0.15,avertc(:,6)-0.4,texlabel('10^4 人'),'fontsize',11);

```

```

text(7-0.15,avertc(:,7)-0.4,texlabel('2x10^4 人'),'fontsize',11);
text(8-0.15,avertc(:,8)-0.4,texlabel('5x10^4 人'),'fontsize',11);
text(9-0.15,avertc(:,9)-0.4,texlabel('10^5 人'),'fontsize',11);
text(10-0.15,avertc(:,10)-0.4,texlabel('5x10^5 人'),'fontsize',11);
xlim([1 11]);xlabel('顾客数量/个','fontsize',11);ylabel('顾客逗留时间/秒','fontsize',11);

```

6.4 随机型动态系统仿真

1、时间步长法

在进行系统模拟时把整个仿真的时间过程分为许多相等的时间间隔，步长的长度根据实际问题决定，通过循环程序控制时间步进的过程。在每一个时间步长上仿真系统动态时，引入随机数以模拟随机因素的影响。

示例 1： 一周 7 天的假期到了，但是当地的气象预报说这一周每天都有 50% 的可能下雨，问在这一周内连续三天下雨的可能性有多大？

解：使用蒙特卡洛模拟方法仿真在这 7 天内每天下雨这个随机时间发生的过程来分析一周内连续三天下雨的可能性。由于天气下雨是个随机时间，只模拟一次不能说明什么问题，需要在相同的条件下模拟多次这个过程，从而得到有关连续三天下雨的统计规律。

用随机变量 y 表示一周内连续三天下雨事件： $y=1$ 表示事件发生， $y=0$ 表示不发生，下面通过频率来确定 $y=1$ 的概率。

假设：每天的天气变化是相互独立的。

下雨事件用随机变量 $x(t)$ 表示，每天下雨的概率 $p=0.5$ 。取 $[0,1]$ 区间上均匀分布的随机数 rand ，当 $\text{rand}>0.5$ 时，令 $x(t)=1$ ，表示第 t 天下雨，否则令 $x(t)=0$ ，表示第 t 天不下雨。让 t 从 1 到 7 循环 7 次，用来模拟 7 天内每天下雨的情况。用变量 C 记录连续下雨的天数，令 $C=3$ ，循环中止，令 $y=1$ ；否则 $y=0$ ，这样就得出了一周之内是否出现了连续三个雨天的状况。Matlab 程序如下：

```

function y=rain(p)
z=rand(1,7);
x=(z>p);
y=0;c=0;
for t=1:7
    if x(t)==1
        c=c+1;
    else
        c=0;
    end
end

```

```

        if c>=3
            y=1;
        end
    end
end

```

重复上面的计算 n 次，即模拟 n 周（例如 $n=100$ ）的天气变化，统计 $y=1$ 的周数 s ，就可以得到一周内连续三天下雨的可能性的频率 s/n ，Matlab 程序如下：

```

function s=rain3(p,n)
s=0;
for k=1:n
    y0=rain(p);
    s=s+y0;
end
s=s/n

```

结果的可信度分析：

因为仿真的是随机现象，需要对使用 100 次实验所得到的下雨周的频率作为下雨周出现的概率是否恰当进行分析。根据大数定律可知，**一个随机变量总体的随机样本的平均数随着样本量的增加将依概率收敛于这个总体的期望**。我们所研究的随机变量 y 是 0-1 分布的变量，期望值就是下雨周发生的概率 μ ，因此，只要 n 充分大前面的结论应该是可信的。但是 n 取多大才是做够大呢？需要对所得到的结论的置信程度作进一步的分析。根据中心极限定理：当 n 充分大时， $(\bar{y} - \mu) / \sqrt{\sigma^2 / n}$ 近似于标准正态分布 $N(0,1)$ ，其中 $\sigma^2 = Dy = 0.43 \times (1 - 0.43) = 0.2451$ ，由此得到 μ 的 95% 的置信区间为

$0.43 \pm 2\sqrt{\sigma^2 / n} \approx [0.3, 0.5]$ ，为了验证这个结论，可以再把上面的仿真运行 40 次分别统计下雨周的次数，结果表面，在 40 次的百周仿真过程中出现的雨周数全部都落在前面估计的 95% 的置信区间内，与前面的理论分析基本一致。但是这个区间的范围比较大，表示雨天概率估计值的可信度不高，可以用 $n=4000$ 或 100000 周的频率来计算，从而缩减置信区间。

其他分析：可进行灵敏性分析，改变降雨概率的值，分析结论对 p 的依赖关系；

改变出现下雨情况是独立同分布的假设，会有什么结果？

示例 2： 市场收款服务问题

某超级市场有两个出口，在出口处的服务有两项，收款和将顾客所购的商品装入袋内，商店只有两名职工从事出口处的服务工作，有两种安排方案：(1)只开一个出口，一人收款，一人装袋；(2)开两个出口，每人既收款又装袋。问商店经历应该选择哪一种出

口处的服务方案。

解：解决这类问题，首先需要确定评价方案好坏的标准。有下面几个选择：（1）把顾客在出口处平均等待时间的长短作为标准；（2）用每分钟时间服务的顾客数作为标准；（3）用服务员的工作效率作为标准等；我们选择第（1）种标准讨论。

在这个问题中，服务员为每一位顾客收款服务的时间，装袋服务的时间以及顾客到达出口处的规律使我们最关心的，为简化问题，作如下假设：

- （1）服务员为每一位顾客服务的收款和装袋时间都是相同的；
- （2）在第一种方案中，收款和装袋同时进行；
- （3）顾客到达出口是随机的。

例如设收款和装袋的时间都是 1 分钟，顾客到达出口的规律是：有 40% 的时间没有顾客到达，有 30% 的时间有一个顾客到达，30% 的时间有两个顾客到达出口。

取仿真步长为 $\Delta t = 1 \text{ min}$ ，引入参量和变量：

$n(t)$ 为在 $(t, t + \Delta t]$ 时间区间内到达收款处的顾客人数， $n(t)$ 是随机变量，可用蒙特卡洛方法获取；

$L(t)$ 表示 $(t, t + \Delta t]$ 时间区间内在收款处排队等待交款的顾客人数；

$L1(t)$ 表示到时刻 t 为止收款处所有顾客人数的累计；

$T1(t)$ 为到时刻 t 为止所有排队顾客等待时间的总和；

$T2(t)$ 为到时刻 t 为止所有已交款顾客接受服务的总时间

$\tau = 1$ 为收款或装袋的时间。

对两个服务员在一个收款台服务的情形，可以写出如下的平衡关系：

当 $L(t) = 0$ ，且 $n(t) = 0$ 时 $L(t+1) = 0, T2(t+1) = T2(t)$ ；否则 $L(t+1) = L(t) + n(t) - 1$ ，

$T2(t+1) = T2(t) + 2$ ；相应的有： $L1(t+1) = L1(t) + n(t)$ ； $T1(t+1) = T1(t) + L(t+1)$

仿真 t_{\min} 内收款台处顾客排队情况的 Matlab 程序如下：

```
function y=paidui(t)
L=zeros(1,t+1); %收款处等待的顾客人数
T1=zeros(1,t+1); %等待时间的累加
T2=zeros(1,t+1); %服务时间的累加
L1=zeros(1,t+1); %到达顾客人数累加
x=0:t;
r=rand(1,t);
for i=1:t
    if 0<=r(i)&&r(i)<0.4
        n=0;
```

```

elseif 0.4<r(i)&&r(i)<0.7
    n=1;
else
    n=2;
end
if L(i)==0&&n==0
    L(i+1)=L(i);
    T1(i+1)=T1(i);
    T2(i+1)=T2(i);
    L1(i+1)=L1(i);
else
    L(i+1)=L(i)+n-1;L1(i+1)=L1(i)+n;
    T1(i+1)=T1(i)+L(i+1);T2(i+1)=T2(i)+1;
end
end
r=[0 r];
a=[x' r' L' L1' T1' T2']%数值模拟结果
%为了分析市场效率，下面计算顾客的平均等待时间，平均逗留时间（等待+被服
务）和平均
%没分时间服务的顾客人数
eL=T2(end)/2; %已被服务的人数
L2=(find(L1>eL));l1=L1(L2);
l2=[eL l1(1:end-1)]; %未被服务的人
k=length(L2);
L3=sum(l1-l2).*(k:-1:1); %扣除的未被服务顾客等待时间总和
g1=(T1(end)-L3)/eL; %在[0,t]时间区间内顾客的平均等待时间
g2=g1+1; %在[0,t]区间内顾客的平均逗留时间
g3=eL/t; %平均每分时间服务的顾客人数
y=[g1 g2 g3];

```

同样考虑两个服务员分别在两个收款台，来到收款台处的顾客排成一队的情形。由于收款和装袋共需要 2min，一个顾客在收款台付费需要逗留 2min，引入变量 t_{ti} 描述一个顾客在第 i 个收款台的逗留时间， $i=1,2$ 。 $t_{ti}=0,1,2$ 分别表示第 i 个收款台没有顾客，一个顾客在那已逗留了 1min 和 2min，写出关于队列长度 $L(t)$ ，顾客累计人数 $L1$ ，排队总时间 $T1(t)$ 和服务总时间 $T2(t)$ 的平衡关系：

当 $L(t)=0$ 且 $n(t)=0$ 时没有人排队 $L(t+1)=0$;

当 $L(t)+n(t)>1$ 时，两个收款台只要有空，即在前一步 $t_{ti}=0$ 或 2，简单表示为 $\text{mod}(t_{ti},2)=0$,队列中就可以减少一位顾客，所以 $L(t+1)=L(t)+n-(2-\text{mod}(t_{t1},2)-\text{mod}(t_{t2},2))$

当 $L(t)+n(t)=1$ 时，即使两个收款台都有空，也只有一位顾客上前接受服务，因此 $L(t+1)=L(t)+n-(1-\min(\text{mod}(t_{t1},2),\text{mod}(t_{t2},2)))$;

相应的 $L1(t+1)=L1(t)+n$; $T1(t+1)=T1(t)+L(t+1)$

在确定 $t+1$ 时刻的收款台状态 tti 后, 只要 $tti>0$, 用符号表示为 $sign(tti)=1$, 服务总时间就增加, $T2(t+1)=T2(t)+sign(tt1)+sign(tt2)$

仿真 $tmin$ 内收款台处得顾客排队情况的 Matlab 程序为:

```
function y=paidui2(t)
L=zeros(1,t+1); %收款处等待的顾客人数
T1=zeros(1,t+1); %等待时间的累加
T2=zeros(1,t+1); %服务时间的累加
L1=zeros(1,t+1); %到达顾客人数累加
tt1=0; %顾客在收款台 1 的逗留时间
tt2=0; %顾客在收款台 2 的逗留时间
x=0:t;r=rand(1,t);
for i=1:t
    if 0<=r(i)&&r(i)<0.4
        n=0;
    elseif 0.4<r(i)&&r(i)<0.7
        n=1;
    else n=2;
    end
    if L(i)==0&&n==0
        L(i+1)=0;
        if tt1==1
            tt1=tt1+1;
        else tt1=0;
        end
        if tt2==1;
            tt2=tt2+1;
        else tt2=0;
        end
    elseif L(i)+n>1
        L(i+1)=L(i)+n-(2-mod(tt1,2)-mod(tt2,2));
        if tt2==1
            tt2=tt2+1;
        else tt2=1;
        end
        if tt1==1
            tt1=tt1+1;
        else tt1=1;
        end
    else L(i+1)=L(i)+n-(1-min(mod(tt1,2),mod(tt2,2)));
        if tt1==1&&tt2==1
            tt1=tt1+1;tt2=tt2+1;
```

```

elseif tt1==1 && tt2~=1
    tt1=tt1+1;tt2=1;
elseif tt1~=1&&tt2==1
    tt1=1;tt2=tt2+1;
else tt1=1;tt2=0;
end
end
L1(i+1)=L1(i)+n;T1(i+1)=T1(i)+L(i+1);
T2(i+1)=T2(i)+sign(tt1)+sign(tt2);
end
r=[0 r];
b=[x' r' L' L1' T1' T2']%数值模拟结果
%为了分析市场效率，下面计算顾客的平均等待时间，平均逗留时间（等待+被服
务）和平均
%没分时间服务的顾客人数
eL=T2(end)/2;%已被服务的人数
L2=(find(L1>eL));
l1=L1(L2);l2=floor(eL) l1(1:end-1)];%未被服务的人
k=length(L2);
L3=sum((l1-l2).*(k:-1:1));%扣除的未被服务顾客等待时间总和
g1=(T1(end)-L3)/eL;%在[0,t]时间区间内顾客的平均等待时间
g2=g1+1;%在[0,t]区间内顾客的平均逗留时间
g3=eL/t;%平均每分时间服务的顾客人数
y=[g1 g2 g3];

```

2、事件表法

从前面的例子可知，使用时间步长法仿真前面的排队过程有很多的冗余过程，当到来的顾客和等待交款的顾客队列的长度都为 0 时，整个系统无需任何计算。而在时间步长法中仍然需要计算和打印输出一次，浪费了许多时间，因此用时间步长法处理这一类问题并不是一个好办法，通常可以用事件表法来仿真。

事件表法就是以**事件**为中心来安排的算法，每处理一个事件就步进一步（每一次步进的时间可能不同）。其主要思路是将系统的仿真过程看成一个事件点的序列。根据事件出现的先后，用一个称之为“事件表”的表格来调度时间执行的顺序。对于那些当前需要处理的事件列入事件表中，从中取出最靠前的事件进行处理。完毕后自动退出事件表。在处理当前事件的过程中，往往会产生后继事件。因此必须预测出后继事件的发生时间，并将它列入事件表中，以使得系统的仿真过程能有条不紊的进行下去。

例 62 再次考虑市场服务问题：当到达收款台的顾客的人数和每个顾客交款的时间都是随机变量时前面的时间步长法不再适用，可以使用事件表的方法。

假设：（1）顾客到达收款台是随机的，平均事件间隔为 0.5min，即间隔时间服从 λ

=2 的指数分布;

(2) 对不同的顾客收款和装袋的时间服从正态分布 $N(1,1/3)$

对两个服务员在一个收款台的情形, 引入变量:

$t(i)$ 表示第 i 位顾客到达;

$t1(i)=t(i+1)-t(i)$: 两位顾客到达收款台的间隔时间服从 $\lambda=2$ 的指数分布的随机变量;

$t2(i)$: 第 i 位顾客接受服务的时间是服从正态分布 $N(1,1/3)$ 的随机变量

$T(i)$: 第 i 位顾客离去时间

将第 i 位顾客到达作为第 i 件事发生, 建立平衡关系: 当 $t(i+1) \geq T(i)$ 时, 即后一位顾客到达时, 前一位顾客已经离开收款台, 所以后者马上接受服务 $T(i+1)=t(i+1)+t2(i+1)$

否则 $T(i+1)=T(i)+t2(i+1)$

模拟 n 位顾客到达收款台前的排队情况的 Matlab 程序为:

```
function y=paidui3(n)
t=zeros(1,n+1);%每位顾客到达时间
T=zeros(1,n+1);%每位顾客离去时间
w=zeros(1,n+1);%顾客等待时间累加
ww=zeros(1,n+1);%收款台空闲时间累加
t1=exprnd(2,1,n+1);
t2=normrnd(1,1/3,1,n+1);
for i=1:n
    t(i+1)=t(i)+t1(i);
    if t(i+1)>=T(i)
        T(i+1)=t(i+1)+t2(i+1);w(i+1)=w(i);
        ww(i+1)=t(i+1)-T(i)+ww(i);
    else
        T(i+1)=T(i)+t2(i+1);w(i+1)=T(i)-t(i+1)+w(i);
        ww(i+1)=ww(i);
    end
end
end
c=[t',T',w',ww'];
g1=w(end)/n;%平均等待时间
g2=sum(T-t)/n;%平均逗留时间
g3=n/T(end);%平均每分钟服务的额顾客人数
y=[g1 g2 g3];
```


7、Matlab 在高等数学中的应用

7.0 Matlab 中的符号表示

1、建立符号变量

Matlab 提供了 **sym** 和 **syms** 两个建立符号对象的函数

1) **sym** 函数，用来建立单个符号变量，调用格式为：

符号变量名=sym('符号字符串')

注：符号字符串可以是常量、变量、函数或者表达式。

2) **syms** 函数，依次可以定义多个符号变量，调用格式为：

syms 符号变量 1 符号变量 2 符号变量 n

注意：符号变量名之间不能使用任何标点符号，只能用空格隔开。

例如：

```
f=sym('cos(x)'), f=sym('sin(x)^2=0');
```

```
syms x;
```

```
f=sin(x)+cos(x),
```

2、建立符号表达式

- 利用单引号来生成表达式
- 利用 **sym** 函数建立符号表达式
- 使用已经定义过的符号变量组成符号表达式

注意 1：符号表达式包括符号函数和符号方程，区别在于是否带有等号。

注意 2：独立变量：是当符号表达式中含有多于一个的变量时，只有一个变量是独立变量。缺省的独立变量是 **x**，如果没有 **x**，则选择最靠近 **x** 的作为独立变量。

可利用函数 **symvar** 询问 **MATLAB** 在符号表达式中哪一个变量它认为是独立变量。例如：**symvar('a*x+y');** **symvar('a*t+s/(u+3)');**

3、符号表达式的运算

(1)四则运算

四则运算分别使用 **symadd**, **symsub**, **symmul** 和 **symdiv** 来实现加减乘除运算，使用 **sympow** 实现幂运算。（注：6.5 以后的版本中已经没有了这些函数，而直接使用 + - * /

即可)

(2)提取分子和分母

如果符号表达式是一个有理分式或可以展开为有理分式,可利用 **numden** 函数来提取符号表达式中的分子或者分母,调用格式为:

`[n,d]=numden(s)`

`[n,d] = numden(sym(4/5))` returns `n = 4` and `d = 5`.

`[n,d] = numden(x/y + y/x)` returns `n = x^2+y^2` , `d = y*x`

(3)因式分解与展开

1) **factors**(s)分解因式

2) **expand**(s)展开

3) **collect**(s)合并同类项

4) **collect**(s,v)按变量 v 进行合并同类项

(4)符号表达式的化简

1) **simplify**(S)应用函数规则对 S 进行化简

2) **simple**(S) 进行综合化简

(5) 符号表达式与数值表达式之间的转换

1) 利用函数 **sym** 可以将数值表达式表示成符号表达式

2) **numeric** 或者 **eval** 函数可以将符号表达式转换成数值表达式

3) 函数 **digits** 和 **vpa** 配合替换函数 **subs** 进行转换。

digits 函数, **digits(D)**函数设置有效数字个数为 D 的近似解精度。

vpa 函数 , `R=vpa(S)`符号表达式 S 在 **digits** 函数设置下的精度的数值解。`vpa(S,D)` 符号表达式 S 在 **digits(D)**精度下的数值解。

subs 函数, `subs(S, OLD, NEW)`

(6) 使用 **latex()**函数可以将不好表达式转换成科学排版语言 LATEX 能支持的字符串,该字符串可以直接嵌入 Latex 文档。

例: `syms s z; P=(s+3)^2*(s^2+3*s+2)*(s^3+12*s^2+48*s+64);`

`P1=simple(subs(P,s,(z-1)/(z+1)));`

`latex(P1)`

4、符号函数的相关运算

(1) 复合函数运算。 **compose** 函数

(2) 反函数运算。 **finverse** 函数

5、符号代数方程求解

(1)线性方程组的求解，函数 **linsolve**，**solve**，可以得到方程的精确解

(2)多变量的非线性方程的符号解法，使用函数 **fsolve**，调用格式有：

- $X = \text{fsolve}('fun', X0)$
- $X = \text{fsolve}('fun', X0, options)$ options 为选择参数输入向量
- $X = \text{fsolve}('fun', X0, options, 'gradfun')$, gradfun 为输入函数在 X 处的偏导数
- $X = \text{fsolve}('fun', X0, options, 'gradfun', P1, P2, \dots)$ P1, P2 为问题定性参数
- $[X, options] = \text{fsolve}('fun', X0, \dots)$ 返回使用的优化方法的参数

注意：fun 必须使用@指定，即 fun 必须是函数句柄。

示例 1： 定义一个函数

```
function F = myfun(x,c)
```

```
    F = [ 2*x(1) - x(2) - exp(c*x(1))  
        -x(1) + 2*x(2) - exp(c*x(2))];
```

```
end
```

求解该函数：

```
c = -1; %定义输入参数
```

```
x = fsolve(@(x) myfun(x,c),[-5;-5])
```

6、多项式的表示方法

(1) 多项式的表示方法——转化为向量问题

对于多项式 $P(x) = a_0x^n + a_1x^{n-1} + \dots + a_{n-1}x + a_n$

用行向量表示： $P = [a_0, a_1, \dots, a_{n-1}, a_n]$

①系数向量直接输入法，MATLAB 自动将向量元素按**降幂顺序**分配给各系数值。函数 **poly2sym** 可以将向量表示的多项式转化为符号多项式表示。

注：由特征多项式生成的多项式首项系数一定为 1；n 阶矩阵一般产生 n 次多项式。

③由根创建多项式，由函数 **poly** 实现。注：若要生成实系数多项式，则根中的复数必定对应共轭；生成的多项式向量包含很小的虚部时可用 **real** 命令将其过滤掉。

(2) 多项式的运算

①多项式求值。输入变量值代入多项式计算时以数组为单元的使用函数 **polyval**（对应元素计算）；以矩阵(必须为方阵)为计算单元求多项式的值用函数 **polyvalm**；

②多项式求根。两种方法，一种是调用函数 **roots**，另一种是通过建立多项式的伴随矩阵再求其特征值的方法得到多项式的所有根。（使用 **compan** 和 **eig** 函数）

③多项式的乘除法运算。乘法使用函数 **conv**（向量卷积），除法使用函数 **deconv**

④多项式微分。微分函数 **polyder**

⑤多项式拟合。两种方法，一种是由矩阵的除法求解超定方程来进行，另一种是用拟合函数 **polyfit**,调用方式为 **polyfit(X,Y,n)**和**[p,s]=polyfit(X,Y,n)**

7.1 求极限

Matlab 求极限的命令为

limit(expr, x, a)

limit(expr, a)

limit(expr)

limit(expr, x, a, 'left')

limit(expr, x, a, 'right')

其中 **limit(expr, x, a)**表示求符号表达式 **expr** 关于符号变量 **x** 趋近于 **a** 时的极限，**limit(expr)**求表示缺省变量趋近于 0 时的极限。

示例 1： 求下列表达式的极限

$$(1) \lim_{x \rightarrow 0} \frac{3 \sin x + x^2 \cos \frac{1}{x}}{(1 + \cos x) \ln(1 + x)}, \quad (2) \lim_{x \rightarrow \infty} \left(\frac{x + 2a}{x - a} \right)^x$$

解 (1) **clc, clear**

syms x

f=(3*sin(x)+x^2*cos(1/x))/((1+cos(x))*log(1+x));

s=limit(f,x,0)

s=simplify(s)

(2) **clc, clear**

syms x a

s1=limit(((x+2*a)/(x-a))^x,x,inf)

s2=limit(((x+2*a)/(x-a))^x,x,-inf)

7.2 求导数

Matlab 的求导数命令为

`diff(expr)`

`diff(expr, v)`

`diff(expr, sym('v'))`

`diff(expr, n)`

`diff(expr, v, n)`

`diff(expr, n, v)`

其中 `diff(expr)` 表示求表达式 `expr` 关于默认变量的 1 阶导数，`diff(expr, v, n)` 和 `diff(expr, n, v)` 都表示求表达式，`expr` 关于符号变量 `v` 的 `n` 阶导数。

- 例 16 (1) 求函数 $y = \ln(x + \sqrt{1+x^2})$ 的二阶导数；
(2) 求向量 $a = [1 \ 0.5 \ 3.5 \ 6]$ 的一阶向前差分。

解 (1) `clc, clear`

`syms x`

`y=log(x+sqrt(1+x^2));`

`s=diff(y,x,2)`

`s=simple(s) %对符号函数进行化简`

(2) `a=[1,0.5,3.5,6]; da=diff(a)`

示例 2: 编写求隐函数的偏导数的 Matlab 函数。

`function dy=impldiff(f,x,y,n)`

`if mod(n,1)~=0`

`error('n should positive integer,please correct');`

`else`

`F1=-simple(diff(f,x)/diff(f,y));`

`dy=F1;`

`for i=2:n`

`dy=simple(diff(dy,x)+diff(dy,y)*F1);`

`end`

`end`

`syms x y;`

`f=(x^2-2*x)*exp(-x^2-y^2-x*y);`

`F1=impldiff(f,x,y,1)`

7.3 求一元函数极值

例 17 求函数 $f(x) = x^3 + 6x^2 + 8x - 1$ 的极值点，并画出函数的图形。

解 对 $f(x)$ 求导，然后令 $\frac{df(x)}{dx} = 0$ ，解方程则可求得函数 $f(x)$ 的极值点。

计算的 Matlab 程序如下

```
syms x
y=x^3+6*x^2+8*x-1; dy=diff(y);
dy_zero=solve(dy), dy_zero_num=double(dy_zero) %变成数值类型
ezplot(y) %符号函数画图
```

7.4 求积分

1. 求不定积分

Matlab 求符号函数不定积分的命令为

```
int(expr)
int(expr, v)
```

例 18 求不定积分

$$\int \frac{1}{1 + \sqrt{1 - x^2}} dx .$$

解 syms x

```
I=int(1/(1+sqrt(1-x^2)));
pretty(I) %写成符号表达式的形式
```

2. 求定积分

(1) 求定积分的符号解

Matlab 求符号函数的定积分命令为

```
int(expr, a, b)
int(expr, v, a, b)
```

例 19 求定积分

$$\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \cos x \cos 2x dx$$

解 syms x

I=int(cos(x)*cos(2*x),-pi/2,pi/2)

(2) 求定积分的数值解

例 20 求下列积分的数值解

i) $\int_2^{+\infty} \frac{dx}{x \cdot \sqrt[3]{x^2 - 3x + 2}};$

ii) $\iint_D \sqrt{1-x^2-y^2} d\sigma, D = \{(x,y) | x^2+y^2 \leq x\};$

iii) $\iiint_{\Omega} \frac{z^2 \ln(x^2+y^2+z^2+1)}{x^2+y^2+z^2+1} dv, \Omega = \{(x,y,z) | 0 \leq z \leq \sqrt{1-x^2-y^2}\}.$

解 i) 做变量替换 $x = \frac{1}{t}, dx = -\frac{1}{t^2} dt$, 得

$$\int_2^{+\infty} \frac{dx}{x \cdot \sqrt[3]{x^2 - 3x + 2}} = -\int_{\frac{1}{2}}^0 \frac{dt}{t \cdot \sqrt[3]{\frac{1}{t^2} - \frac{3}{t} + 2}} = \int_0^{\frac{1}{2}} \frac{dt}{\sqrt[3]{t - 3t^2 + 2t^3}}$$

输入:

方法一:

I=**quadl**(@(t) (t-3*t.^2+2*t.^3).^(-1/3),eps,0.5) %对积分函数进行了倒数变换

或者 I=**quad**(@(t) (t-3*t.^2+2*t.^3).^(-1/3),eps,0.5)

得到 I=1.4396。

方法二:

I=**integral**(@(t) (t-3*t.^2+2*t.^3).^(-1/3),eps,0.5)

结果 I=1.4396;

注: **quad()**函数使用 simpson 法,即用二次曲线逼近被积函数, **quadl()**采用更高阶的逼近方法。它们的调用方法是: q=quad(fun,a,b,tol)或者 quadl(fun,a,b,tol), fun 可以是符号方程,也可以是匿名函数。integral 函数是自 Matlab8.0 版开始引入的新的自适应变步长数值积分函数,调用格式为: I=integral(fun,a,b,属性设置对), a, b 是积分限,属性设置对用来进行选项控制,如 ReTol, AbsTol 等。

ii) 输入

方法一:

I=**dblquad**(@(x,y)sqrt(1-x.^2-y.^2).*(x.^2+y.^2<=x),0,1,-0.5,0.5)

得到 $I = 0.6028$ 。

方法二：

$I = \text{integral2}(@(x,y)\text{sqrt}(1-x.^2-y.^2).*(x.^2+y.^2 \leq x), 0, 1, -0.5, 0.5)$

注：Matlab 中计算二重积分的函数形式为 $q = \text{dblquad}(\text{fun}, x_{\min}, x_{\max}, y_{\min}, y_{\max}, \text{tol})$ ，一般只用于积分区域为矩形的情况。为了使用该函数，有时需要将非矩形区域化成矩形区域。

示例 1： 计算二重积分 $\iint_{\Omega} (x^2 + y^2) dx dy$ 积分区域 Ω 由 $x=1, y=x$ 及 $y=0$ 围成。

解：首先绘制积分区域，Matlab 程序如下：

```
clear,clc
fill([0,1,1,0],[0,0,1,0],'y');hold on%绘制积分区域
fill([0.55,0.6,0.6,0.55,0.55],[0,0,0.6,0.55,0],'r') %绘制单元条
gtext('y=x');pause;
gtext('x=1');pause
gtext('y=0')
```

按照矩形区域调用 `dblquad` 函数，程序如下：

$I = \text{dblquad}(@(x,y)(x.^2+y.^2).*(y-x < 0), 0, 1, 0, 1)$

或者 $I = \text{integral2}(@(x,y)(x.^2+y.^2).*(y-x < 0), 0, 1, 0, 1)$

求得 $I = 0.3333$

(3) 输入

```
fun3=@(x,y,z)z.^2.*log(x.^2+y.^2+z.^2+1)./(x.^2+y.^2+z.^2+1).*(z>=0 &...
z<=sqrt(1-x.^2-y.^2)); %该语句使用了续行符...
```

$I = \text{triplequad}(\text{fun3}, -1, 1, -1, 1, 0, 1)$ %三重积分计算函数

得到 $I = 0.1273$ 。

例 21 计算 $I = \iiint_V (x + y + z)^2 dx dy dz$ ，其中 V 为 $z \geq x^2 + y^2$ 与 $x^2 + y^2 + z^2 \leq 2$ 所围成的区域。

解法一 积分区域 V 显然关于 yOz 坐标平面对称, 也关于 xOz 坐标平面对称。又

$$(x+y+z)^2 = x^2 + y^2 + z^2 + 2xy + 2xz + 2yz,$$

由于 $2xy + 2yz$ 是关于 y 的奇函数, 故

$$\iiint_V (2xy + 2yz) dx dy dz = 0.$$

又 $2xz$ 是关于 x 的奇函数, 故 $\iiint_V 2xz dx dy dz = 0$ 。所以

$$I = \iiint_V (x+y+z)^2 dx dy dz = \iiint_V (x^2 + y^2 + z^2) dx dy dz.$$

在柱面坐标系中, V 的边界曲面 $z = x^2 + y^2$ 和 $x^2 + y^2 + z^2 = 2$ 可分别表示为 $z = r^2$, $z = \sqrt{2-r^2}$, 故

$$\begin{aligned} I &= \iiint_V (x^2 + y^2 + z^2) dx dy dz = \int_0^{2\pi} d\theta \int_0^1 dr \int_{r^2}^{\sqrt{2-r^2}} (r^2 + z^2) r dz \\ &= \left(\frac{8}{5} \sqrt{2} - \frac{89}{60} \right) \pi. \end{aligned}$$

Matlab 符号求解的程序如下

```
clc, clear
```

```
syms r z theta
```

```
I1=int((r^2+z^2)*r, z, r^2,sqrt(2-r^2)); %求最内层积分
```

```
I2=int(I1,r,0,1); %求中间层的积分
```

```
I=int(I2,theta,0,2*pi) %求最终的积分结果
```

```
pretty(I) %分数线中间显示的格式
```

```
Inum=double(I) %把符号解化成数值解
```

解法二 求数值解的 Matlab 程序如下

```
clc, clear
```

```
f=@(x,y,z) (x+y+z).^2.*(z>=x.^2+y.^2 & x.^2+y.^2+z.^2<=2);
```

```
I=triplequad(f,-sqrt(2),sqrt(2),-sqrt(2),sqrt(2),0,sqrt(2))
```

求得积分的值为 2.4486。

注: quad 与 int 的区别, int 可以求具有解析解的不定积分和定积分, quad 系列函数只能求解定积分。

7.5 求解微分方程(组)

1、微分方程(组)的解析解

在 MATLAB 中用 D 表示导数,例如 Dy 表示 y' , D2y 表示 y'' , Dy(0)=5 表示 $y'(0)=5$ 等, 符号常微分方程求解通过函数 **dsolve** 实现, 新版调用格式为:

S=dsolve(eqn)

S=dsolve(eqn, cond)

S=dsolve(eqn, cond, Name, Value)

旧版用法: **dsolve('e','c','v')** 包括微分方程, 初始条件和指定变量三个部分, 求解常微分方程 e 在初始条件 c 下的特解, v 描述方程中的自变量, 省略则默认以 t 为自变量, 若没有初始条件, 则获得通解。

例 1: 求 $\frac{du}{dt} = 1 + u^2$ 的通解

解: Matlab 程序为:

>>syms u(t)

>>eqn=diff(u, t)==1+u^2;

>>S=dsolve(eqn);

或者:

S=dsolve('Du=1+u^2','t')

例 2: 求微分方程 $\begin{cases} \frac{d^2y}{dx^2} + 4\frac{dy}{dx} + 29y = 0 \\ y(0) = 0, y'(0) = 15 \end{cases}$ 的特解

解: Matlab 程序为:

dsolve('D2y+4*Dy+29*y=0','y(0)=0,Dy(0)=15','x')

或者:

syms y(t)

eqn=diff(y, t, 2)+4*diff(y,t)+29*y==0

Dy=diff(y, t)

cond=[y(0)==0,Dy(0)==15]

S=dsolve(eqn, cond)

例 3: 带有参数的微分方程组求解

$$\frac{d^2y}{dt^2} = a^2y, \quad y(0) = b \text{ and } y'(0) = 1$$

Matlab 程序为:

```
syms y(t) a b
```

```
eqn = diff(y,t,2) == a^2*y;
```

```
Dy = diff(y,t);
```

```
cond = [y(0)==b, Dy(0)==1];
```

```
ySol(t) = dsolve(eqn,cond)
```

例 4: 求常微分方程组的通解

$$\begin{cases} \frac{dx}{dt} = 2x - 3y + 3z \\ \frac{dy}{dt} = 4x - 5y + 3z \\ \frac{dz}{dt} = 4x - 4y + 2z \end{cases}$$

解: Matlab 程序为:

```
[x,y,z]=dsolve('Dx=2*x-3*y+3*z','Dy=4*x-5*y+3*z','Dz=4*x-4*y+2*z')
```

新版解法:

```
syms x(t) y(t) z(t)
```

```
eqns=[diff(x,t)==2*x-2*y+3, diff(y,t)==4*x-5*y+3, diff(z,t)=4*x-4*y+2]
```

```
Sol=dsolve(eqns)
```

一阶齐次线性常微分方程组和非齐次线性方程组可以直接通过矩阵操作求解。

对于一阶齐次线性微分方程组: $X' = AX, X = [x_1, \dots, x_n], A = (a_{ij})_{n \times n}$ 其解的形式为:

$$X(t) = e^{A(t-t_0)} X_0$$

对于非齐次线性方程组: $X' = AX + f(t), X(t_0) = X_0$ 解的形式为:

$$X(t) = e^{A(t-t_0)} X_0 + \int e^{A(t-s)} f(s) ds$$

例 5: $X' = \begin{bmatrix} 2 & 1 & 3 \\ 0 & 2 & -1 \\ 0 & 0 & 3 \end{bmatrix} X, X(0) = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$

解: `syms t;a=[2,1,3;0,2,-1;0,0,2];`

`x0=[1;2;1];x=expm(a*t)*x0`

`pretty(x); %化简结果`

$$\text{例 5: } X' = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & -2 \\ 3 & 2 & 1 \end{bmatrix} X + \begin{bmatrix} 0 \\ 0 \\ e^t \cos(t) \end{bmatrix}, X(0) = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

解: syms t s;

a=[1 0 0;2 1 -2;3 2 1]; fs=[0;0;exp(s)*cos(2*s)];

x0=[0;1;1];

tx=int(expm(a*(t-s))*fs,s); %先求不定积分, 再计算定积分的结果, 提高速度

xstar=subs(tx,s,t)-subs(tx,s,0);

x=expm(a*t)*x0+xstar;

x=simple(s),pretty(x) %simple 直接显示化简结果, simplify 显示化简过程, pretty 修改显示格式

2、微分方程（组）的数值解

当难以求得微分方程的解析解时, 可以求其数值解。Matlab 中求解微分方程数值解的方法有以下几个: **ode45**, **ode23**, **ode113**, **ode23s**, **ode15s**。调用形式如下:

[t,x]=solver('f', ts, x0, options) 其中, t 是自变量的值, x 是函数值, solver 表示上述 5 种求解器的一种; 'f' 是由待解方程写成的 m 文件名, ts=[t0, tf] 表示自变量的初值和终值, options 用于设定误差限, 通过函数 **options=odeset('reltol', rt, 'abstol', at)** 分别设定相对误差和绝对误差。

不同的函数代表不同的内部算法, **ode45** 表示 4/5 阶龙格-库塔-费尔贝格算法 (RK 算法), 是解非刚性常微分方程的首选方法, ode23 采用 2/3 阶 RK 方法, ode113 采用多步法, 效率一般比 ode45 高。**ode15s**, **ode23s**, **ode23t**, **ode23tb** 用来解刚性常微分方程。

注 1: 在解 n 个未知函数的的方程组时, x0 和 xn 均为 n 维向量, m 文件中的待解方程组应以 x 分量形式写出;

注 2: 使用 Matlab 求数值解时, 高阶微分方程必须等价的变换成一阶微分方程组。

$$\text{例 1: 求解 van der Pol 刚性微分方程 } \begin{cases} \frac{d^2 x}{dt^2} - 1000(1-x^2) \frac{dx}{dt} - x = 0 \\ x(0) = 2; x'(0) = 0 \end{cases} \text{ 的数值解}$$

解: 令 y1=x, y2=y1', 则微分方程变为一阶微分方程组:

$$\begin{cases} y1' = y2 \\ y2' = 1000(1-y1^2)y2 - y1 \\ y1(0) = 2; y2(0) = 0 \end{cases}$$

写成 m 文件如下所示：

```
function dy=fun1(t,y)
dy=zeros(2,1);
dy(1)=y(2);
dy(2)=1000*(1-y(1)^2)*y(2)-y(1);
取[t0,tf]=[0,3000], 输入命令:
[t,y]=ode15s('fun1',[0,3000],[2,0]);
plot(t,y(:,1),'-')
```

例 2：求解微分方程组
$$\begin{cases} y_1' = y_2 y_3 \\ y_2' = -y_1 y_3 \\ y_3' = -0.51 y_1 y_2 \\ y_1(0) = 0, y_2(0) = 1; y_3(0) = 1 \end{cases}$$

解：建立方程组的 m 文件：

```
function dy=fun2(t,y)
dy=zeros(3,1); %必须初始化为列向量
dy(1)=y(2)*y(3);
dy(2)=y(1)*y(3);
dy(3)=-0.51*y(1)*y(2);
取 t0=0, tf=12, Matlab 命令如下:
[t,y]=ode45('fun2',[0 12],[0 1 1])
plot(t,y(:,1),'-',t,y(:,2),'*',t,y(:,3),'+')
```

注意：定义的微分方程组也可以含有附加参数，附加参数需要写到微分方程函数中作为形参存在，例如 `function dy=fun(t,y,a1,a2)`，在调用时，需要在 `ode45` 函数中给形参赋予具体的数值，如 `[t,x]=ode45(@fun,tspan,t0,a1,a2)`。

例 3：求解 Lorez 方程
$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = \beta x - y - xz \\ \dot{z} = -\lambda z + xy \end{cases}$$
 在初始条件 $[x_0, y_0, z_0] = [5, 13, 17]$ 时的解，其

中 $\sigma = 10, \beta = 28, \lambda = 8/3$ 。

解：rho=10;beta=29;lamda=8/3;
f=@(t,Y)[rho*(Y(2)-Y(1));beta*Y(1)-Y(2)-Y(1)*Y(3);-lamda*Y(3)+Y(1)*Y(2)];%方程

组必须使用列向量表示

```
[t,y]=ode45(f,[0,30],[5,13,17])
subplot(2,2,1)
plot(t,y(:,1),'*') %x 曲线
subplot(2,2,2)
plot(t,y(:,2),'X')
subplot(2,2,3)
plot(t,y(:,3),'O')
subplot(2,2,4)
plot3(y(:,1),y(:,2),y(:,3)) %绘制空间轨迹图，可以用 comet3 绘制动画式轨迹。
```

3、边值问题的 Matlab 数值解

Matlab 中使用 **bvp4c** (**bvp5c**) 和 **bvpinit** 命令求解常微分方程中的两点边值问题，其调用格式为：

```
sol = bvp4c(odefun,bcfun,solinit,options)
solinit = bvpinit(x,yinit,parameters) %该函数用来给 sol 提供初始猜测解，x 可以用 linspace(a,b,M)获得。
```

示例：求解非线性微分方程边值问题 $y''=F(x,y,y')=2yy'$, $y(0)=-1$, $y(\pi/2)=1$
解：

```
%令 x1=y,x2=y', 则方程化简为: x1'=x2,x2'=2x1x2
f1=@(t,x)[x(2);2*x(1)*x(2)]; %微分方程的 Matlab 函数
f2=@(xa,xb)[xa(1)+1;xb(1)-1]; %边界条件的 Matlab 函数
sinit=bvpinit(linspace(0,pi/2,5),rand(2,1));
sol=bvp5c(f1,f2,sinit);
plot(sol.x,sol.y)
```

注意：如果上面的边界条件改为： $y'(\pi/2)=1$ 和 $y(\pi/2)-y(0)=1$ ，则对应的边值函数修改为： $f2=@(xa,xb)[xb(2)-1;xb(1)-xa(1)-1]$

4、时延微分方程求解函数

常数时延微分方程函数 **dde23()**

时变延迟的微分方程函数 **ddesd()**

中立型延迟微分方程函数 **ddensd()**

例 1：接下面的时延微分方程组

$$\begin{cases} x_1'(t) = 1 - 3x_1(t) - x_2(t-1) - 0.2 * x_1^3(t-0.5) - x_1(t-0.5) \\ x_2'(t) = x_3(t) \\ x_3'(t) = 4x_1(t) - 2x_2(t) - 3x_3(t) \end{cases}$$

```
f=@(t,x,Z)[1-3*x(1)-Z(2,1)-0.2*Z(1,2)^3-Z(1,2);x(3);4*x(1)-2*x(2)-3*x(3)];
```

```
tau=[1,0.5];
```

```
sol=dde23(f,tau,zeros(3,1),[0,10]);
```

```
plot(sol.x,so.y(1,:))
```

5、偏微分方程函数

pdepe()函数可以直接求解偏微分方程，其入口为[c,f,s]=pdefun(x,t,u,ux);

边值函数可以由[pa,qa,pb,qb]=pdebc(x,t,u,ux)描述;

初始函数可以由 u0=pdeic(x)来描述;

sol=pdepe(m,@pdefun,@pdeic,@pdebc,x,t);

可以求解的表达式形式为:

$$c\left(x, t, u, \frac{\partial u}{\partial x}\right) \frac{\partial u}{\partial t} = x^{-m} \frac{\partial}{\partial x} \left[x^m f\left(x, t, u, \frac{\partial u}{\partial x}\right) \right] + s\left(x, t, u, \frac{\partial u}{\partial x}\right)$$

边界条件可以用下面的函数描述为

$$p(x, t, u) + q(x, t, u) \cdot f\left(x, t, u, \frac{\partial u}{\partial x}\right) = 0$$

并由此派生出 (a, b) 区间端点上的边界条件为

$$p(a, t, u) + q(a, t, u) \cdot f\left(a, t, u, \frac{\partial u}{\partial x}\right) = 0$$
$$p(b, t, u) + q(b, t, u) \cdot f\left(b, t, u, \frac{\partial u}{\partial x}\right) = 0$$

例：计算下面的偏微分方程

解：

```
function [c,f,s]=exmpde(x,t,u,du)
c=[1;1];
y=u(1)-u(2);
F=exp(-5.73*y)-exp(-11.46*y);
s=[-F;F];
f=[0.024*du(1);0.17*du(2)];
end
function [pa,pb,qa,qb]=exmpbc(xa,ua,xb,ub,t)
pa=[0;ua(2)];
qa=[1;0];
pb=[ub(1)-1;0];
qb=[0;1];
end
u0=@(x)[1;0];
x=0:0.05:1;
```

```

t=0:0.05:2;
m=0;
sol=pdepe(m,@exmpde,u0,@exmpbc,x,t);
surf(x,t,sol(:,:,1))
figure;
surf(x,t,sol(:,:,2))

```

7.6 级数求和

Matlab 级数求和的命令为

r = symsum(expr, v)

r = symsum(expr, v, a, b)

其中 **expr** 为级数的通项表达式，**v** 是求和变量，**a** 和 **b** 分别为求和变量的起始点和终止点，若没有指明 **a** 和 **b**，**a** 的默认值为 0，**b** 的默认值为 **v-1**。

无穷数列的累加称为级数，当取其前面若干有限项时，得到的是部分和，数列 **a** 累加形成的新序列可用 **s=cumsum(a)** 实现。**a** 的长度为 **n**，则 **s** 的长度也为 **n**，即每一个 **s(k)** 是数组 **a** 中前 **k** 项的和。注意 **cumsum** 与 **sum** 的区别，**sum(a)** 得到的是一个数，即序列 **s** 中最后一项。

示例：

```

clc,clear
k=1:10;
a=1./k.^2;
s=cumsum(a)
ss=sum(a)

```

例 22 求如下级数的和

$$(1) \sum_{n=1}^{\infty} \frac{2n-1}{2^n}, (2) \sum_{n=1}^{\infty} \frac{1}{n^2}.$$

解 (1) syms n

f1=(2*n-1)/2^n;

s1=symsum(f1,n,1,inf)

(2) syms n

f2=1/n^2;

s2=symsum(f2,n,1,inf)

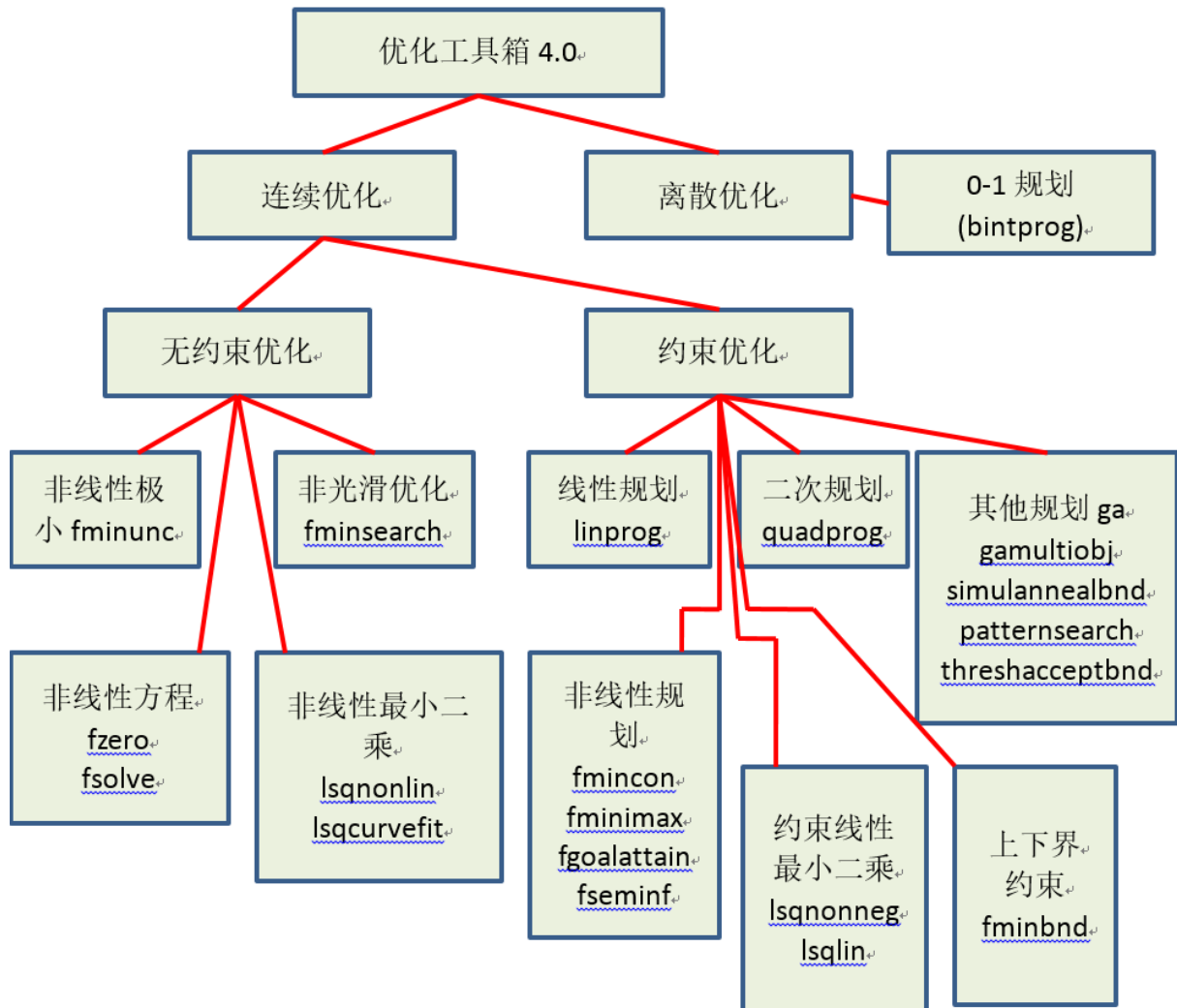
补充一点: Matlab 求数组元素乘积的函数 **B=prod(A)** 如果 **A** 是一个 **m** 行一列的(向量)，则这种用法即返回这 **m** 个元素的乘积； 如果 **A** 是一个 **m** 行 **n** 列的矩阵，则 **A** 的每一列都被看做一个 **m** 行 1 列的向量，分别计算每个向量中元素的乘积，返回给 **B**，

因此 B 是一个 1 行 n 列的数组。

7.7 Matlab 求解优化问题

Matlab 求解各种形式的极值问题被集成在优化工具箱中。

下面是优化工具箱 4.0 版本的功能示意图：



| 类型 | 模型 | 基本函数 |
|---------|---|---|
| 一元函数极小值 | $\text{Min } F(x)$ $s.t. x_1 < x < x_2$ | <code>x=fminbnd('F',x1,x2)</code> |
| 无约束极小值 | $\text{Min } F(x)$ | <code>x=fminunc('F',x0)</code> <code>x=fminsearch('F',x0)</code> |
| 线性规划 | $\text{Min } c^T x$ $s.t. AX \leq b$ | <code>x=linprog(c,A,b)</code> |
| 0-1整数规划 | $\text{min } F(x)$ $s.t. \begin{cases} AX \leq b \\ Aeq \cdot X = beq \\ X_i = 0 \text{ 或 } 1 \end{cases}$ | <code>x=bintprog(F)</code> |
| 二次规划 | $\text{Min } \frac{1}{2} x^T H x + c^T x$ $s.t. Ax \leq b$ | <code>x=quadprog(H,c,A,b)</code> |
| 约束极小值 | $\text{Min } F(x)$ $s.t. G(x) \leq 0$ | <code>x=fmincon('FG',X0)</code> |
| 非线性最小二乘 | $\text{min } F^2(X) = \sum_{i=1}^n F_i^2(X)$ | <code>x=lsqnonlin(F,X0)</code> |
| 目标达到问题 | $\text{Min } r$ $s.t. F(x) - wr \leq goal$ | <code>x=fgoalattain('F',x,goal,w)</code> |
| 极小极大问题 | $\text{Min}_x \max_{(F(x))} \{ F_i(x) \}$ $s.t. G(x) \leq 0$ | <code>x=fminimax('FG',x0)</code> |

| 变量 | 描述 | 调用函数 |
|-----------|--|--|
| f | 线性规划目标函数 f^*X 或二次规划的目标函数 $X^*H*X + F^*X$ 中线性项的系数向量 | <code>linprog,quadprog</code> |
| fun | 非线性优化的目标函数. <code>fun</code> 必须为行命令对象或 M 文件、嵌入函数、或 MEX 文件的名称 | <code>fminbnd,fminsearch,fminunc,fmincon,lsqcurvefit,lsqnonlin,fgoalattain,fminimax</code> |
| H | 二次规划目标函数 $X^*H*X + f^*X$ 中的二次项的系数矩阵 | <code>quadprog</code> |
| A,b | A 矩阵和 b 向量分别为线性不等式约束： $AX \leq b$ 中的系数矩阵和右端向量 | <code>linprog,quadprog,fgoalattain,fmincon,fminimax</code> |
| Aeq,beq | Aeq 矩阵和 beq 向量分别为线性等式约束 $Aeq \cdot X = beq$ 中的系数矩阵和右端向量 | <code>linprog,quadprog,fgoalattain,fmincon,fminimax</code> |
| vlb,vub | X 的下限和上限向量 | <code>linprog,quadprog,fgoalattain,fmincon,fminimax,lsqcurvefit,lsqnonlin</code> |

| | | |
|----------------|----------|-----------------|
| <i>X0</i> | 迭代初始点坐标 | 除 fminbnd 外所有函数 |
| <i>x1,x2</i> | 函数最小化的区间 | fminbnd |
| <i>options</i> | 优化选项参数结构 | 所有优化函数 |

| 变量 | 描述 | 调用函数 |
|-----------------|---|---|
| <i>exitflag</i> | 描述退出条件： exitflag>0 表示目标函数收敛于 x exitflag=0 表示已达到函数评价或迭代的最大次数 exitflag<0 表示目标函数不收敛 | |
| <i>x</i> | 由优化函数求的 x 值 | 所有优化函数 |
| <i>feval</i> | 解 x 处的目标函数值 | linprog,quadprog,fgoalattain fmincon,fminimax,lsqcurvefit lsqnonlin,fminbnd |
| <i>output</i> | 包含优化结果信息的输出结构 iterations,Algorithm,FuncCount（函数评价次数） | 所有优化函数 |

进行极值计算时有两种方式，通过命令 `optimtool` 打开优化工具箱 GUI 求解和使用上述提供的命令函数进行求解。

示例 1：使用单纯形计算最小值

$$\begin{aligned} \min f &= -4x_1 - x_2 \\ \text{s.t.} \quad &\begin{cases} -x_1 + 2x_2 \leq 4 \\ 2x_1 + 3x_2 \leq 12 \\ x_1 - x_2 \leq 3 \\ x_1, x_2 \geq 0 \end{cases} \end{aligned}$$

方法一：通过优化工具箱求解

方法二：输入 Matlab 命令：

```
clc,clear
```

```
f=[-4;-1];
```

```
A=[-1 2;2 3;1 -1];
```

```
b=[4;12;3];
```

```
[x,fval,exitflag,output]=linprog(f,A,b)
```

注意：有些线性规划问题中自变量的下标是双下标或者多重下标的，这样就不能直

接用示例 1 的方法求解,而应该首先将原问题转化为单下标自变量的最优化问题,为此,需要重新选定变量。

示例 2:

$$\begin{aligned} \min \quad & 2800(x_{11} + x_{21} + x_{31} + x_{41}) + 4500(x_{12} + x_{22} + x_{32}) + 6000(x_{13} + x_{23}) + 7200 \\ \text{s.t.} \quad & \begin{cases} x_{11} + x_{12} + x_{13} + x_{14} \geq 15 \\ x_{12} + x_{13} + x_{14} + x_{21} + x_{22} + x_{23} \geq 10 \\ x_{13} + x_{14} + x_{22} + x_{23} + x_{31} + x_{32} \geq 20 \\ x_{14} + x_{23} + x_{32} + x_{41} \geq 20 \\ x_{ij} \geq 0 (i = 1, 2, 3, 4, j = 1, 2, 3, 4) \end{cases} \end{aligned}$$

解: 令 $x_1 = x_{11}, x_2 = x_{12}, x_3 = x_{13}, \dots$ 即可, 程序如下:

```
f=2800*[1 0 0 0 1 0 0 1 0 1]+4500*[0 1 0 0 0 1 0 0 1 0]+...
    6000*[0 0 1 0 0 0 1 0 0 0]+7300*[0 0 0 1 0 0 0 0 0 0];
A=-[1 1 1 1 0 0 0 0 0 0;0 1 1 1 1 1 1 1 0 0 0;
    0 0 1 1 0 1 1 1 1 0;0 0 0 1 0 0 1 0 1 1];
B=[15;10;20;12];
xm=[0 0 0 0 0 0 0 0 0 0];
Aeq=[];beq=[];
x=linprog(f,A,B,Aeq,beq,xm)
```

注意: 对于一般的非线性规划问题, 可以使用 `fmincon()` 函数进行求解。

示例 3: 非线性优化

$$\begin{aligned} f(x) &= -x_1 x_2 x_3 \\ \text{s.t.} \quad & \begin{cases} -x_1 - 2x_2 - 2x_3 \leq 0 \\ x_1 + 2x_2 + 2x_3 \leq 72 \end{cases} \end{aligned}$$

方法一: 通过优化工具箱求解

方法二: 输入 Matlab 命令:

```
clc,clear
f=@(x)(-x(1)*x(2)*x(3));
A=[-1 -2 -2;1 2 2];
b=[0;72];
[x,fval,exitflag,output]=fmincon(f,[10;10;10],A,b)
```

示例 4:

$$\begin{aligned} \min \quad & 1000 - x_1^2 - 2x_2^2 - x_3^2 - x_1x_2 - x_1x_3 \\ \text{s.t.} \quad & \begin{cases} x_1^2 + x_2^2 + x_3^2 - 25 = 0 \\ 8x_1 + 14x_2 + 7x_3 - 56 = 0 \\ x_1, x_2, x_3 \geq 0 \end{cases} \end{aligned}$$

解：约束条件中含有非线性条件，需要使用非线性规划的方式来求解

方法一：

```
f=@(x)1000-x(1)*x(1)-2*x(2)*x(2)-x(3)*x(3)-x(1)*x(2)-x(1)*x(3)
% ff=optimset;
% ff.LargeScale='off';
% ff.TolFun=1e-30;
% ff.TolX=1e-15;
% ff.TolCon=1e-20;
x0=[1;1;1];
xm=[0;0;0];
xM=[];
A=[];
B=[];
Aeq=[];
Beq=[];
[x,f_opt,c,d]=fmincon(f,x0,A,B,Aeq,Beq,xm,xM,@opt_con1);
% 定义非线性约束
function [c ceq]=opt_con1(x)
ceq=[x(1)*x(1)+x(2)*x(2)+x(3)*x(3)-25;8*x(1)+14*x(2)+7*x(3)-56];
c=[];
end
```

方法二、使用结构体的方法求解

%非线性约束条件如下定义：

```
function [c ceq]=opt_con1(x)
ceq=x(1)*x(1)+x(2)*x(2)+x(3)*x(3)-25;
c=[];
end
```

求解程序：

```
clear P;

P.objective=f;
P.nonlcon=@opt_con2;
P.x0=x0;
P.lb=xm;
```

```

P.options=ff;
P.solver='fmincon'
[x,f_opt,c,d]=fmincon(P);
方法三、也可以将线性约束条件单独拿出来
x0=[1;1;1];
Aeq=[8,14,7];
Beq=56;
[x,f_opt,c,d]=fmincon(f,x0,A,B,Aeq,Beq,xm,xM,@opt_con2);

```

注意：如果一次求解不能搜索到最优结果，则可以多次运行或者通过编写程序使寻优过程多次执行。

其他求解极值的函数主要有：

(1) $x=fminbnd(fun,x1,x2)$ ，求函数在区间 $[x1,x2]$ 上的极小值对应的自变量值

示例 5：求 $f(x)=x^4-x^2+x-1$ 在区间 $[-2,1]$ 上的极小值。

解： Matlab 命令如下：

```
[x,feval,exitflag,output]=fminbnd('x^4-x^2+x-1',-2,1)
```

(2) $x=fminsearch(fun,x0)$ 无约束非线性极小值求解， $x0$ 为起始搜索点。

示例 6： **fminsearch** 函数求 $f(x)=1/((x1-2)^2+3)-1/(2(x2+1)^2-5)$,初始点 $[0,0]$,

(3) $x=fminunc(fun,x0,options)$ 求解函数的一个局部极小值

(4) $x=fminimax(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon)$ nonlcon 包括非线性约束;

(5) $x=lsqnonlin(fun,x0,lb,ub)$;非线性最小二乘优化

(6) $x=linprog(f,A,b,Aeq,beq,lb,ub,x0)$ 求解线性规划问题

(7) $x=bintprog(f,A,b,Aeq,beq,x0)$ 求解 0-1 规划

(8) $x=quadprog(H,f,A,b,Aeq,beq,lb,ub,x0)$;求解二次规划

(9) $x = intlinprog(f,intcon,A,b,Aeq,beq,lb,ub,options)$ 混合整数线性规划问题

示例 7： 二次规划问题

数学模型为：

$$\min \quad \frac{1}{2} x^T H x + f^T x$$

$$s.t. \quad \begin{cases} Ax \leq B \\ A_{eq} x \leq B_{eq} \\ x_m \leq x \leq x_M \end{cases}$$

求下面的二次规划问题：

$$\begin{aligned} \min \quad & (x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 - 3)^2 + (x_4 - 4)^2 \\ \text{s.t.} \quad & \begin{cases} x_1 + x_2 + x_3 + x_4 \leq 5 \\ 3x_1 + 3x_2 + 2x_3 + x_4 \leq 10 \\ x_1, x_2, x_3, x_4 \geq 0 \end{cases} \end{aligned}$$

解：需要将目标函数转换成标准形式，然后调用函数 `quadprog` 进行计算

```
f=[-2,-4,-6,-8];
H=diag([2 2 2 2]);
Opt=optimset;
Opt.LargeScale='off' %不适用大规模问题的算法
A=[1,1,1,1;3,3,2,1];
B=[5;10];
Aeq=[];
Beq=[];
LB=zeros(4,1);
[x,f_opt]=quadprog(H,f,A,B,Aeq,Beq,LB,[],[],Opt);
```

注意：不要忘记 **H** 矩阵的生成中有一个 $1/2$ 项，另外，此题中还有一个常数 30，在计算时省去了，编程得出结果后还要再加上常数项。

7.8 代数方程的求解

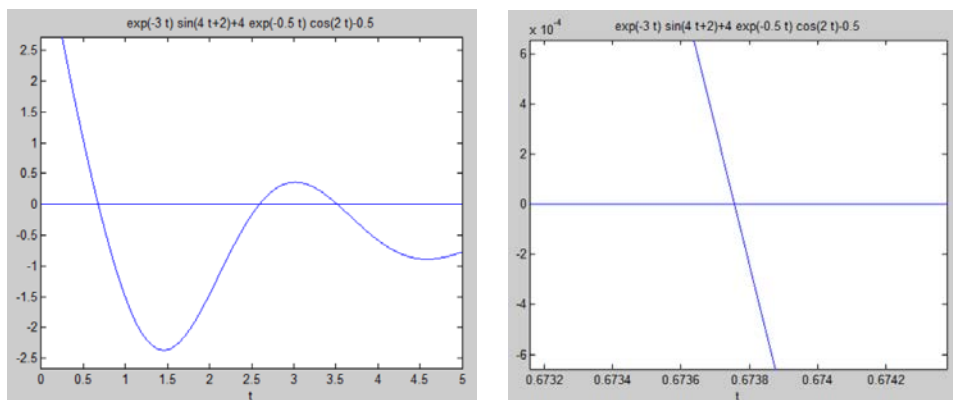
7.8.1 代数方程的图解法

1、一元方程的图解法。利用 `ezplot` 绘制隐函数 $f(x)=0$ 的曲线，可以用图解法从给出的曲线和直线 $y=0$ 的交点得到所有的实数解。

示例 1：解方程 $\exp(-3t)\sin(4t+2)+4\exp(-0.5t)\cos(2t)=0.5$

解： `ezplot('exp(-3*t)*sin(4*t+2)+4*exp(-0.5*t)*cos(2*t)-0.5',[0,5]);`

`line([0,5],[0,0]); %绘制横轴`



从绘制的图形可以看出方程可能有多个实根，如果要用图解法求得某个根，可以对该点进行局部放大，直到曲线穿越横轴，且横轴给出的各个标点的数值完全一致时，可以

断定原方程的解即为 t 轴标度, 局部放大结果如上图所示, 可得方程的一个解为 $t=0.6738$, 对应得到函数值使用下面的语句获得:

```
t=0.6738,vpa(exp(-3*t)*sin(4*t+2)+4*exp(-0.5*t)*cos(2*t)-0.5)
```

2、二元方程的图解法

二元方程也可以通过 `ezplot` 函数将第一个方程对应的曲线绘制出来, 在使用 `hold on` 命令保持图形不被刷新, 最后调用 `ezplot` 函数将第二个曲线在同一坐标下绘制出来, 得到曲线后就可以通过读取交点坐标的方式得出联立方程的根。

示例: 用图解法求方程组
$$\begin{cases} x^2 e^{-xy^2/2} + e^{-x/2} \sin(xy) = 0 \\ y^2 \cos(x+y^2) + x^2 e^{x+y} = 0 \end{cases}$$
 的解

解:

```
ezplot('x^2*exp(-x*y^2/2)+exp(-x/2)*sin(x*y)')
hold on
ezplot('y^2*cos(y+x^2)+x^2*exp(x+y)')
```

7.8.2 多项式型方程的准解析解法

Matlab 符号运算工具箱中给出了 `solve()` 函数对多项式类方程求解十分有效, 可以找出多项式方程所有的根, 其定义格式为:

`S=solve(eqn1,eqn2,...,eqnn)` 或者

`[x,y,...]= solve(eqn1,eqn2,...,eqnn,'x,y,...')` %指定变量

示例 1: 解二元方程组
$$\begin{cases} x^2 + y^2 - 1 = 0 \\ 0.75x^3 - y + 0.9 = 0 \end{cases}$$

解:

```
[x,y]=solve('x^2+y^2-1=0','0.75*x^3-y+0.9=0')
```

```
[eval('x.^2+y.^2-1') eval('0.75*x.^3-y+0.9')]
```

 %用来验证结果是否满足方程。

`solve()` 可以直接实现带有变量的方程的解, 这样的求解用普通的数值解方法是不能实现的。

注意: 对于没有 `=` 的表达式, `solve()` 会将其置为 0; 对其他非缺省变量求解, 需要指定该求解变量。

示例 2: 解带有参数的方程
$$\begin{cases} x^2 + ax^2 + 6b + 3y^2 = 0 \\ y = a + x + 3 \end{cases}$$
 的解

解:

`syms a b`

`[x,y]=solve('x^2+a*x^2+6*b+3*y^2=0','y=a+x+3','x,y')` %指定 x,y 为求解变量

7.9 Matlab 求解插值和拟合问题

设 $f(x)$ 是给定的函数，本身未知，仅知道一组自变量和因变量的值，插值是根据已知样本点的信息获得在其他点上函数值的方法，如果在这些给定的点的范围内进行插值，称为内插，否则称为外插，如果从时间概念上来理解，则外插也称为预测或者预报。

7.9.1 Matlab 插值工具箱

1、一维插值函数 **interp1**，命令格式为：

`yi=interp1(x,Y,xi,'method')` 其中 `method` 指定的方法为 `nearest,linear,spline,cubic,pchip`（分段三次 Hermite 插值），所有的插值方法都要求 x 是单调的。 x 与 Y 是具有相同大小的向量，求在 xi 点处的插值函数值 yi

一维快速插值函数 `interp1q`，命令格式为：

`yi = interp1q(x,Y,xi)`，其中 x 是单调递增的列向量， Y 是列向量或者行数为 `length(x)` 的矩阵， xi 是一个列向量。

`yi = interp1(x,Y,xi,method,'extrap')` %对于超出 x 范围的 xi 中的分量将执行特殊的外插值法 `extrap`。

示例 1：在一 天 24 小时内，从零点开始每间隔 2 小时测得的环境温度数据分别为 12, 9, 9, 10, 18, 24, 28, 27, 25, 20, 18, 15, 13，推测中午 12 点（即 13 点）时的温度。

解：

`x=0:2:24;`

`y=[12 9 9 10 18 24 28 27 25 20 18 15 13];`

`a=13;`

`y1=interp1(x,y,a,'spline')`

`xi=0:1/3600:24;` %绘制一天内的温度变化曲线

`yi=interp1(x,y,xi, 'spline');`

`plot(x,y,'o',xi,yi)`

示例 2：

`x=0:0.12:1;`

`y=(x.^2-3*x+5).*exp(-5*x).*sin(x);`

`plot(x,y,'ro',x,y);`

`x1=0:0.02:1;` %要插值点

`y1=(x1.^2-3*x1+5).*exp(-5*x1).*sin(x1);`

`y2=interp1(x,y,x1);y3=interp1(x,y,x1,'spline');`

```

y4=interp1(x,y,x1,'nearest');y5=interp1(x,y,x1,'cubic');
plot(x1,[y2' y3' y4' y5'],'.',x,y,'ro',x1,y1);
legend('linear','spline','nearest','cubic','样本点','原函数');
%计算各插值的最大误差
[max(abs(y1-y2)),max(abs(y1-y3)),max(abs(y1-y4)),max(abs(y1-y5))];

```

2、二维数据内插值 **interp2**，插值函数为二元函数。

```

ZI = interp2(X,Y,Z,XI,YI,'method')

```

X 和 Y 分别是 m 维和 n 维的向量，表示节点，Z 为 n*m 的矩阵，表示节点值，XI，YI 是为二维数组，表示插值点，XI 与 YI 是方向不同的向量，一个是行向量，一个是列向量，ZI 是行数为 YI 的维数，列数为 XI 的维数的矩阵，表示得到的插值。若 XI 与 YI 中有在 X 与 Y 范围之外的点，则相应地返回 nan (Not a Number)。method 为插值方法。

如果在某区域测量了若干个节点的高程(节点值)，为了画出比较精确的等高线图，需要先插入更多的点(插值点)，然后计算这些点的高程。

如果是三次样条插值，可以使用命令

```

pp=csape({x0,y0},z0,conds,valconds);z=fnval(pp,{x,y})

```

示例 1:

```

years = 1950:10:1990;
service = 10:10:30;
wage = [150.697 199.592 187.625
179.323 195.072 250.287
203.212 179.092 322.767
226.505 153.706 426.730
249.633 120.281 598.243];
w = interp2(service,years,wage,15,1975) %计算在 1975 年工作了 15 年的员工的工资

```

示例2: 由 $z = f(x, y) = (x^2 - 2x)e^{-x^2 - y^2 - xy}$ 生成一些稀疏的网格数据，

然后对整个函数曲面进行插值。

```

clear;[x,y]=meshgrid(-3:0.6:3,-2:0.4:2);
z=(x.^2-2*x).*exp(-x.^2-y.^2-x.*y);
surf(x,y,z);figure
[x1,y1]=meshgrid(-3:0.2:3,-2:0.2:2);
z1=interp2(x,y,z,x1,y1);
z2=interp2(x,y,z,x1,y1,'cubic');
z3=interp2(x,y,z,x1,y1,'spline');
surf(x1,y1,z1);figure
%比较误差
z0=(x1.^2-2*x1).*exp(-x1.^2-y1.^2-x1.*y1);
surf(x1,y1,abs(z0-z2));axis([-3,3,-2,2,0,0.08])
figure;surf(x1,y1,abs(z0-z3));
axis([-3,3,-2,2,0,0.08])

```

interp2 函数能够较好的进行二维插值运算，但是它只能处理以网格形式给出的数据，

如果数据是随机给出的，此时可以使用 **griddata** 函数。其调用格式为：

格式： $z = \text{griddata}(x0, y0, z0, x, y, 'method')$

'v4': MATLAB4.0 提供的插值算法，公认效果较好；

'linear': 双线性插值算法（缺省算法）；

'nearest': 最临近插值；

'cubic': 双三次插值。

示例3：已知 $z = f(x, y) = (x^2 - 2x)e^{-x^2 - y^2 - xy}$ 在 $x: [-3, 3], y: [-2, 2]$ 矩形区域内随机给出一组坐标，使用 **griddata** 进行插值并作误差分析。

```
clear
x=-3+6*rand(199,1);y=-2+4*rand(199,1);
z=(x.^2-2*x).*exp(-x.^2-y.^2-x.*y);
plot(x,y,'*');figure;plot3(x,y,z,'*');figure;
[x1,y1]=meshgrid(-3:0.2:3,-2:0.2:2);
z1=griddata(x,y,z,x1,y1,'cubic');
surf(x1,y1,z1);figure;
z2=griddata(x,y,z,x1,y1,'v4');surf(x1,y1,z2);
%误差比较
z0=(x1.^2-2*x1).*exp(-x1.^2-y1.^2-x1.*y1);
surf(x1,y1,abs(z0-z1));axis([-3,3,-2,2,0,0.15])
figure;
surf(x1,y1,abs(z0-z2));axis([-3,3,-2,2,0,0.15])
```

3、三维插值运算函数 **interp3**, n 维网格插值 **interp**n, 其调用格式同 **interp1** 和 **interp2**, 对应的三维网格生成函数为 $[x, y, z] = \text{meshgrid}(x1, y1, z1)$ 和非网格生成函数 **griddata3()**, **griddatan()**, 他们同 **griddata** 调用格式相同。

示例 1:

$V(x, y, z) = e^{x^2z + y^2x + z^2y} \cos(x^2yz + z^2yx)$ 通过函数生成一些网格型样本

点，试根据样本点进行拟合，并给出拟合误差。

```
解: [x,y,z]=meshgrid(-1:0.2:1); [x0,y0,z0]=meshgrid(-1:0.05:1);
V=exp(x.^2.*z+y.^2.*x+z.^2.*y).*cos(x.^2.*y.*z+z.^2.*y.*x);
V0=exp(x0.^2.*z0+y0.^2.*x0+z0.^2.*y0).*cos(x0.^2.*y0.*z0+z0.^2.*y0.*x0);
V1=interp3(x,y,z,V,x0,y0,z0,'spline'); err=V1-V0; max(err(:))
slice(x0,y0,z0,V1,[-0.5,0.3, 0.9],[0.6,-0.1],[-1,-0.5,0.5,1])
title('Slives for Four Dim Figures');
```

4、三次样条插值函数

已知平面上 n 个点 $x_i, y_i (i=1 \dots n)$ ，其中 $x_1 < x_2 < \dots < x_n$ ， $S(x)$ 为三次样条函数需要满足三个条件：

(1) $S(x_i) = y_i (i=1 \dots n)$ ，即函数经过样本点；

(2) $S(x)$ 在每个子区间 $[x_i, x_{i+1}]$ 上为三次多项式：

$$S(x) = c_{i1}(x - x_i)^3 + c_{i2}(x - x_i)^2 + c_{i3}(x - x_i) + c_{i4}$$

(3) $S(x)$ 在整个区间 $[x_1, x_n]$ 上有连续的一阶及二阶导数。

Matlab 中有如下函数：

$y = \text{interp1}(x, Y, xi, 'spline');$

$y = \text{spline}(x, y, xi);$ % 对于给定的离散的测量数据 x, y （称为断点），要寻找一个三项多项式 $y = p(x)$ ，以逼近每对数据 (x, y) 点间的曲线。

示例 1: 已知 $z = f(x, y) = (x^2 - 2x)e^{-x^2 - y^2 - xy}$ 用三次参数样条插值

的方法进行拟合并绘制曲面

解: $x_0 = -3:0.6:3; y_0 = -2:0.4:2; [x, y] = \text{ndgrid}(x_0, y_0);$

$z = (x.^2 - 2*x) .* \exp(-x.^2 - y.^2 - x.*y);$

$sp = \text{csapi}(\{x_0, y_0\}, z);$

$\text{fnplt}(sp);$

示例 2: 对离散分布在 $y = \exp(x)\sin(x)$ 函数曲线上的数据点进行样条插值计算

解: $x = [0 \ 2 \ 4 \ 5 \ 8 \ 12 \ 12.8 \ 17.2 \ 19.9 \ 20];$

$y = \exp(x) .* \sin(x);$

$xx = 0:0.25:20;$

$yy = \text{spline}(x, y, xx);$

$\text{plot}(x, y, 'o', xx, yy)$

$pp = \text{csape}(x, y, conds);$ % conds 是边界条件选项，可以取以下值：

'complete', 给定边界一阶导数.

'not-a-knot', 非扭结条件, 不用给边界值.

'periodic', 周期性边界条件, 不用给边界值.

'second', 给定边界二阶导数.

'variational', 自然样条(边界二阶导数为 0)

$pp = \text{csape}(x, y, conds, valconds); y = \text{ppval}(pp, x);$ 求插值点的函数值必须调用 ppval 函数。

示例 3：给定数据 $x=[1\ 2\ 4\ 5]$ ， $y=[1\ 3\ 4\ 2]$ ，边界条件 $S''(1)=2.5$ ， $S''(5)=-3$ ，计算插值。

```
解： x=[1 2 4 5];y=[1 3 4 2];  
pp=csape(x,y,'second',[2.5,-3]);  
pp.coefs %显示每个区间上三次多项式的系数  
xi=1:0.1:5;  
yi=ppval(pp,xi); %计算插值点对应的函数值  
plot(x,y,'o',xi,yi)
```

Matlab 还提供了一个生成三次参数样条类的函数 $S=csapi(x,y)$

其中 $x=[x_1,x_2,\dots,x_n]$ ， $y=[y_1,y_2,\dots,y_n]$ 为样本点。 S 返回样条函数对象的插值结果，包括子区间点、各区间点三次多项式系数等。

可用 `fnplt()`绘制出插值结果，其调用格式：`fnplt(S)`

对给定的向量 x_p ，可用 `fnval()`函数计算，其调用格式：`yp=fnval(S,xp)`

其中得出的 y_p 是 x_p 上各点的插值结果。

示例 4：

$f(x) = (x^2 - 3x + 5)e^{-5x}\sin(x)$ 的一些数据点，用三次参数样条插值的方法对这些数据进行拟合。

```
解： x=0:.12:1; y=(x.^2-3*x+5).*exp(-5*x).*sin(x);
```

```
sp=csapi(x,y); fnplt(sp)
```

```
c=[sp.breaks(1:4)' sp.breaks(2:5)' sp.coefs(1:4,:),...
```

```
    sp.breaks(5:8)' sp.breaks(6:9)' sp.coefs(5:8,:)]
```

处理多个自变量的网格数据三次样条插值类的函数：

```
S=csapi({x1,x2,...,xn},z)
```

其中 x_i 为自变量标识， z 是网格数据的样本点， S 是三次样条曲线的对象。

7.9.2 Matlab 中的拟合方法

1、线性最小二乘法拟合曲线

当用 m 次多项式拟合给定数据时，一般使用函数 **polyfit**，命令格式为：

$a=polyfit(x,y,m)$ ，输入参数 x ， y 为要拟合的数据， m 为拟合多项式的次数，输出参数 a 为拟合多项式的系数向量。

多项式在 x 处的函数值 y 可以使用函数 `polyval`，格式为：

$y = \text{polyval}(a, x)$, a 为向量表示的多项式

示例：某乡镇企业 1990-1996 年的生产利润如下表所示，试预测 1997 年和 1998 年的利润。

| 年份 | 1990 | 1991 | 1992 | 1993 | 1994 | 1995 | 1996 |
|----|------|------|------|------|------|------|------|
| 利润 | 70 | 122 | 144 | 152 | 174 | 196 | 202 |

解：

```
x=[1990 1991 1992 1993 1994 1995 1996];
```

```
y=[70 122 144 152 174 196 202];
```

```
a=polyfit(x,y,1); %根据利润增长的情况，使用一次线性拟合函数
```

```
y97=polyval(a,1997);
```

```
y98=polyval(a,1998);
```

注意：多项式拟合相当于对已知函数用 Taylor 幂级数表示，但是 Taylor 幂级数展开的前提条件是函数应该已知。

2、最小二乘优化

常用的求解最小二乘优化的函数 **lsqlin**, **lsqcurvefit**, **lsqnonlin**, **lsqnonneg**，这些函数都位于 Matlab 的优化工具箱中，下面只举例说明它们的用法。

```
x=lsqlin(C,d,A,b,Aeq,beq,lb,ub,x0)
```

求解的模型为：
$$\min_x \frac{1}{2} \|C \cdot x - d\|_2^2 \text{ such that } \begin{cases} A \cdot x \leq b, \\ Aeq \cdot x = beq, \\ lb \leq x \leq ub. \end{cases}$$

示例 1: $x=[19 \ 25 \ 31 \ 38 \ 44]'$;

$y=[19 \ 32.3 \ 49 \ 73.3 \ 97.8]'$;

$r=[\text{ones}(5,1), x.^2]$; %拟合形如 $y=a+b \cdot x^2$ 的多项式，如果拟合的多项式的形式为 $y=a_0+a_1 \cdot x+a_2 \cdot x^2$ ，则 r 应该写成 $r=[\text{ones}(5,1), x, x.^2]$

```
ab=lsqlin(r,y); %计算多项式的系数
```

```
x0=19:0.1:44;
```

```
y0=ab(1)+ab(2)*x0.^2;
```

```
plot(x,y,'o',x0,y0,'r') %绘制对比曲线查看结果
```

lsqcurvefit 函数求解的数学模型为

$$\min_x \|F(x, xdata) - ydata\|_2^2 = \min_x \sum_i (F(x, xdata_i) - ydata_i)^2,$$

函数的调用格式为：

`x=lsqcurvefit(fun,x0,xdata,ydata,lb,ub,options)`; `fun` 参数是定义函数 $F(x, xdata)$ 的 M 文件, `x0` 为初始点, `xdata` 是输入数据, `ydata` 是观测数据, 它们都是向量或者矩阵, $F(x, xdata)$ 的维数和 `ydata` 一致。

示例 2: 用给定的观测数据拟合函数 $y = e^{-k_1 x_1} \sin(k_2 x_2) + x_3^2$ 中的参数 k_1 和 k_2 。

解：(1) 首先编写 M 文件，定义拟合函数

```
function y=fun(k,xdata)
f=exp(-k(1)*xdata(:,1)).*sin(k(2)*xdata(:,2))+xdata(:,3).^2;
end
```

(2) 调用函数 `lsqcurvefit`

`x0=[23.73 5.49 1.21;...];` %`x0` 是 25 行 3 列的输入样本

`y0=[15.02;...];` %`y0` 是 25 行 1 列的观测值

`k0=rand(2,1);` %初始化待求参数的值

`lb=zeros(2,1);ub=[20;2];` %给定参数的上下界，没有边界时用[]表示

`k=lsqcurvefit(@fun,k0,x0,y0,lb,ub)`

`lsqnonlin` 函数用来解决的数学模型是： $\min_x \|f(x)\|_2^2 = \min_x (f_1(x)^2 + f_2(x)^2 + \dots + f_n(x)^2)$

命令格式为：

`x=lsqnonlin(fun,x0,lb,ub,options)`;其中 `fun` 是定义向量函数 $f(x)$ 的 M 文件或者函数句柄。

示例 3: 用最小二乘法拟合 $y = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$ 中的未知参数 μ, σ ，假设数据值

`x0,y0` 已知。

解：`x0=-10:0.01:10;`

`y0=normpdf(x0,0,1);` %计算标准正态分布密度函数在 `x0` 处的取值

`F=@(xs)1/sqrt(2*pi)/xs(2)*exp(-(x0-xs(1)).^2)/xs(2)^2/2-y0;` %定义函数

`xs0=rand(2,1);`%非线性拟合要有初始值

xs=lsqnonlin (F,xs0) ;

示例 4: 示例 3:体重约 70kg 的某人在短时间内喝下 2 瓶啤酒后, 隔一定时间测量他的血液中的酒精含量(mg/100ml), 得到如下数据, 使用所给数据用函数 $f=at^be^{ct}$ 进行拟合, 求出常数 a,b,c。

解: clear clc;
t=[0.25 0.5 0.75 1 1.5 2 2.5 3 3.5 4 4.5 5:16];
h=[30 68 75 82 82 77 68 68 58 51 50 41 38 35 28 25 18 15 12 10 7 7 4];
h1=log(h); %进行对数变换
f=inline('a(1)+a(2).*log(t)+a(3).*t','a','t');
[x,r]=lsqcurvefit(f,[1,0.5,-0.5],t,h1);
plot(t,h,'*');
hold on;
ezplot('exp(4.4834+0.4709*log(t)-0.2663*t)',[0.2 16]);
grid on;

示例 5: 多目标优化问题
一般的数学模型:

$$J = \min_{s.t. G(x) \leq 0} F(x)$$

无约束多目标函数的最小二乘解法, 假设多目标规划问题的目标函数

$F(x)=[f_1(x), f_2(x), \dots, f_n(x)]$, 则可以按照下面的方式转化成单目标问题:

$J = \min_{s.t. x_m \leq x \leq x_M} f_1^2(x) + f_2^2(x) + \dots + f_n^2(x)$, 然后调用 fmincon()函数求解。

求解下面的无约束非线性多目标规划问题的最小二乘解:

$$\min \begin{bmatrix} (x_1 + 2x_2 + 3x_3) \sin(x_1 + x_2) e^{-x_1^2 - x_3^2} + 5x_3 \\ e^{-x_2^2 - 4x_3^2} \cos(4x_1 + x_2) \end{bmatrix}$$

$$s.t. \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \leq x \leq \begin{bmatrix} 3 \\ \pi \\ 5 \end{bmatrix}$$

解:

f=@(x)[(x(1)+2*x(2)+3*x(3))*sin(x(1)+x(2))*exp(-x(1)^2-x(3)^2)+5*x(3);
exp(-x(2)^2-4*x(3)^2)*cos(4*x(1)+x(2))];
xm=[0;0;0];
xM=[3;pi;5];
x0=xm;
G=@(x)f(x)*f(x)
x=fmincon(G,x0,[],[],[],[],xm,xM)

lsqnonneg 函数求解的数学模型为: $\min_x \|C \cdot x - d\|_2^2$, where $x \geq 0$.

命令格式为:

$x = \text{lsqnonneg}(C, d, \text{options})$, 该函数的用法同 lsqlin 用法类似。

示例 6: 线性加权变换纠结多目标规划问题

对不同的指标添加权值, 然后将多个目标改写成下面的形式:

$$f(x) = w_1 f_1(x) + w_2 f_2(x) + \dots + w_p f_p(x), \text{ 其中 } w_1 + w_2 + \dots + w_p = 1 \text{ 且 } 0 \leq w_i \leq 1$$

求解下面的问题:

某商店有 A1, A2, A3 三种糖果, 单价分别是 4, 2.8, 2.4 元/kg, 现在筹办一个茶话会, 要求买糖果的钱不超过 20 元, 糖果总量不少于 6kg, A1 和 A2 两种糖果总量不少于 3kg, 应该如何确定最好的买糖方案。

解: 设三种糖果购买量分别是 x_1, x_2, x_3 , 则两个目标函数如下:

$$\text{花费 } \min \quad f_1(x) = 4x_1 + 2.8x_2 + 2.4x_3;$$

$$\text{重量 } \max \quad f_2(x) = x_1 + x_2 + x_3;$$

写成数学模型如下所示:

$$\begin{aligned} \min \quad & \begin{bmatrix} 4x_1 + 2.8x_2 + 2.4x_3 \\ -(x_1 + x_2 + x_3) \end{bmatrix} \\ \text{s.t.} \quad & \begin{cases} 4x_1 + 2.8x_2 + 2.4x_3 \leq 20 \\ x_1 + x_2 + x_3 \geq 6 \\ x_1 + x_2 \geq 3 \\ x_1, x_2, x_3 \geq 0 \end{cases} \end{aligned}$$

将上面的模型改写为:

$$\begin{aligned} \min \quad & (w_1[4, 2.8, 2.4] - w_2[1, 1, 1])x \\ \text{s.t.} \quad & \begin{cases} 4x_1 + 2.8x_2 + 2.4x_3 \leq 20 \\ -(x_1 + x_2 + x_3) \leq 6 \\ -(x_1 + x_2) \leq 3 \\ x_1, x_2, x_3 \geq 0 \end{cases} \end{aligned}$$

Matlab 程序:

```
f1=[4,2.8,2.4];f2=[1,1,1];
Aeq=[];Beq=[];
xm=[0;0;0];C=[];
A=[4.2 2.8 2.4;-1 -1 -1;-1 -1 0];
B=[20;-6;-3];
ww1=[0:0.1:1];
for w1=ww1,w2=1-w1;
    x=linprog(w1*f1+w2*f2,A,B,Aeq,Beq,xm);
    C=[C;w1 w2 x' f1*x -f2*x]
```

end

这样可以得到不同加权系数下的最优方案。

示例 7：极小极大问题求解

极小极大问题也是多目标优化中的一类问题，假设有 p 个目标函数，它们中每一个均可以取出最大值，然后从这些最大值中进行最小化搜索，或者说极小极大问题是在最不利的条件下寻找最有利决策方案的一种方法。数学模型表示为：

$$J = \min \left[\max_{s.t. G(x) \leq 0} f_i(x) \right]$$

fminimax()函数提供了求解方法，调用格式与 fmincon()类似。

Matlab 程序示例：

```
f=@(x)[x(1)^2*sin(x(2))+x(2)-3*x(1)*x(2)*cos(x(1));  
-x(1)^2*exp(-x(2))-x(2)^2*exp(-x(1))+x(1)*x(2)*cos(x(1)*x(2));  
x(1)^2+x(2)^2-2*x(1)*x(2)+x(1)-x(2);  
-x(1)^2-x(2)^2*cos(x(1)*x(2))];  
A=[4.3 3.8;1 1];  
B=[4.9;3];  
x=fminimax(f,rand(2,1),A,B)
```

示例 8：目标规划问题，fgoalattain()

有些最优化问题找不到可行解，这就需要放松约束条件，例如，将不等式约束条件 $Ax \leq B$ 改写为 $Ax \leq B + d^- - d^+$ ，将偏差 $d^- - d^+$ 引入目标函数，使之最小。相应的，目标函数也应该给出某一个满意指标。对应的数学模型为：

$$\begin{aligned} \min \quad & \gamma \\ s.t. \quad & \begin{cases} F(x) - w\gamma \leq g \\ Ax \leq B \\ A_{eq}x \leq B_{eq} \\ c(x) \leq 0 \\ c_{eq}(x) = 0 \\ x_m \leq x \leq x_M \end{cases} \end{aligned}$$

上面的 g 为人为引入的目标。

使用该方法求解示例 6 中的问题，Matlab 代码如下：

```
f=@(x)[[4,2.8,2.4]*x;[-1 -1 -1]*x];  
Aeq=[];Beq=[];  
w=[0.8,0.2];  
goal=[20;-6];  
A=[-1,-1,0]; B=[-3];
```

```
x=fgoalattain(f,x0,goal,w,A,B,[],[],xm);
f(x);
```

3、Matlab 曲线拟合的图形用户界面操作

拟合一元函数的命令是 `cftool`，拟合二元函数的命令 `sftool`，非线性拟合的命令是 `nlintool`。

主要拟合步骤是：

- (1)将数据导入工作空间；
- (2)运行 `cftool` 命令；
- (3)对数据进行预处理；
- (4)选择适当的模型进行拟合；
- (5)生成一些相关的统计量，并进行预测操作

示例 1：利用下表的数据拟合方程 $y = a_0 e^{a_1 x_1} e^{a_2 x_2}$

| x_1 | x_2 | y |
|-------|-------|-----|
| 5 | 2 | 1.5 |
| 5 | 5 | 2.3 |
| 7 | 10 | 3.5 |
| 8 | 4 | 5.2 |
| 8 | 6 | 1.8 |
| 3 | 3 | 2.9 |
| 7 | 8 | 3.8 |
| 7 | 3 | 4.5 |
| 2 | 6 | 4.7 |

解：

(1) 把数据加载到 Matlab 工作空间中，

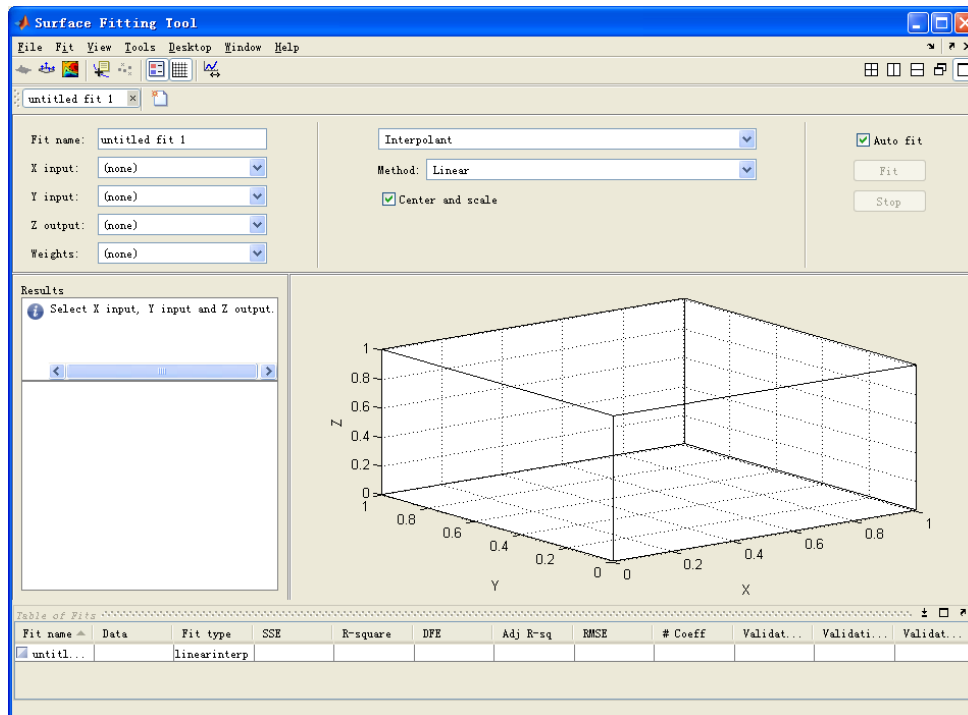
```
a=[5 2 1.5;5 5 2.3;7 10 3.5;8 4 5.2;8 6 1.8;3 3 2.9;7 8 3.8;7 3 4.5;2 6 4.7];
```

```
x1=a(:,1);
```

```
x2=a(:,2);
```

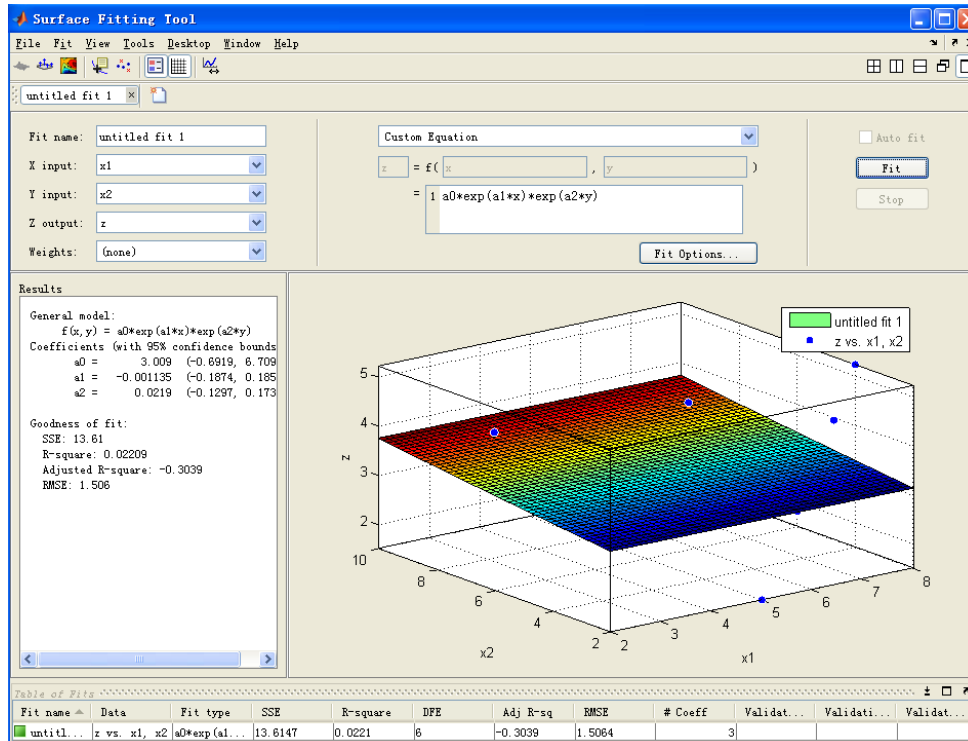
```
z=a(:,3);
```

(2)在命令窗口中输入 `sftool`，打开如下界面



(3) 输入自变量 X, Y 的数据 x1,x2, 因变量 Z 的数据 z, 并定义要拟合的函数 $a_0 \cdot \exp(a_1 \cdot x) \cdot \exp(a_2 \cdot y)$

(4) 用鼠标点击 fit 进行拟合, 结果如下



8、Matlab 在线性代数中的应用

8.1 向量组的线性相关性

求列向量组 A 的一个最大线性无关组可用命令 **rref(A)** 将 A 化成行最简形，其中单位向量对应的列向量即为最大线性无关组所含向量，其它列向量的坐标即为其对应向量用最大线性无关组线性表示的系数。

例 23 求下列矩阵列向量组的一个最大无关组。

$$A = \begin{bmatrix} 1 & -2 & -1 & 0 & 2 \\ -2 & 4 & 2 & 6 & -6 \\ 2 & -1 & 0 & 2 & 3 \\ 3 & 3 & 3 & 3 & 4 \end{bmatrix}$$

解 编写 Matlab 程序如下

format rat %数据是有理分数表示

a=[1,-2,-1,0,2;-2,4,2,6,-6;2,-1,0,2,3;3,3,3,3,4];

b=rref(a)

format %恢复到短小数的显示格式

求得

$$b = \begin{bmatrix} 1 & 0 & 1/3 & 0 & 16/3 \\ 0 & 1 & 2/3 & 0 & -1/9 \\ 0 & 0 & 0 & 1 & -1/3 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

记矩阵 A 的五个列向量依次为 α_1 、 α_2 、 α_3 、 α_4 、 α_5 ，则 α_1 、 α_2 、 α_4 是列向量组的一个最大无关组。且有

$$\alpha_3 = \frac{1}{3}\alpha_1 + \frac{2}{3}\alpha_2, \quad \alpha_5 = \frac{16}{3}\alpha_1 - \frac{1}{9}\alpha_2 - \frac{1}{3}\alpha_4.$$

例 24 设 $A = [a_1, a_2, a_3] = \begin{bmatrix} 2 & 2 & -1 \\ 2 & -1 & 2 \\ -1 & 2 & 2 \end{bmatrix}$, $B = [b_1, b_2] = \begin{bmatrix} 1 & 4 \\ 0 & 3 \\ -4 & 2 \end{bmatrix}$.

验证 a_1, a_2, a_3 是 R^3 的一个基，并把 b_1, b_2 用这个基线性表示。

解 编写 Matlab 程序如下

format rat

```
a=[2,2,-1;2,-1,2;-1,2,2];b=[1,4;0,3;-4,2];
```

```
c=rref([a,b])
```

```
format %恢复到短小数的显示格式
```

求得

```
c= 1 0 0 2/3 4/3
    0 1 0 -2/3 1
    0 0 1 -1 2/3
```

说明 a_1, a_2, a_3 是 R^3 的一个基, 且有 $b_1 = \frac{2}{3}a_1 - \frac{2}{3}a_2 - a_3$, $b_2 = \frac{4}{3}a_1 + a_2 + \frac{2}{3}a_3$ 。

8.2 齐次线性方程组

在 Matlab 中, 函数 **null** 用来求解零空间, 即满足 $Ax=0$ 的解空间, 实际上是求出解空间的一组基 (基础解系)。格式如下:

```
z=null(A) %z 的列向量为方程组的正交规范基, 满足。ZTZ=E
```

```
z=null(A,'r') %z 的列向量是方程 Ax=0 的有理基。
```

例 25 求方程组的通解

$$\begin{cases} x_1 + 2x_2 + 2x_3 + x_4 = 0, \\ 2x_1 + x_2 - 2x_3 - 2x_4 = 0, \\ x_1 - x_2 - 4x_3 - 3x_4 = 0. \end{cases}$$

解 编写程序如下

```
format rat
```

```
a=[1,2,2,1;2,1,-2,-2;1,-1,-4,-3]
```

```
b=null(a,'r') %求有理基
```

```
syms k1 k2
```

```
x=k1*b(:,1)+k2*b(:,2) %写出方程组的通解
```

```
format %恢复到短小数的显示格式
```

8.3 非齐次线性方程组

Matlab 中解非齐次线性方程组可以使用“\”。虽然表面上只是一个简简单单的符号, 而它的内部却包含许许多多的自适应算法, 如对超定方程 (无解) 用最小二乘法, 对欠定方程 (多解) 它将给出范数最小的一个解。

对于非齐次线性方程组 $Ax=b$ ，Matlab 的求解命令为

$x=A\backslash b$ 或者 $(x=inv(A)*b)$

无论 $Ax=b$ 是有唯一解，无穷多解还是无解时，Matlab 总是给出一个解。

(1) 当 $Ax=b$ 唯一解时，Matlab 给出的就是该唯一解。

(2) 当 $Ax=b$ 有无穷多解时，Matlab 给出的是最小范数解。

(3) 当 $Ax=b$ 无解时，Matlab 给出的是最小二乘解。

另外求解欠定方程组（多解）可以使用求矩阵 A 的行最简形命令 `rref(A)`，求出所有的基础解系。

例 26 求超定方程组

$$\begin{cases} 2x_1 + 4x_2 = 11, \\ 3x_1 - 5x_2 = 3, \\ x_1 + 2x_2 = 6, \\ 2x_1 + x_2 = 7. \end{cases}$$

解 编写程序如下

```
a=[2,4;3,-5;1,2;2,1];
```

```
b=[11;3;6;7];
```

```
solution=a\b
```

注：上面解超定方程组的“\”可以用伪逆命令 **pinv** 代替，且 `pinv` 的使用范围比“\”更加广泛，`pinv` 也给出最小二乘解或最小范数解。

例 27 用最小二乘解法解方程组

$$\begin{cases} x_1 + x_2 = 1, \\ x_1 + x_3 = 2, \\ x_1 + x_2 + x_3 = 0, \\ x_1 + 2x_2 - x_3 = -1. \end{cases}$$

解 编写程序如下

```
format rat
```

```
a=[1,1,0;1,0,1;1,1,1;1,2,-1];
```

```
b=[1;2;0;-1];
```

```
x1=a\b %这里\和 pinv 是等价的
```

```
x2=pinv(a)*b
```

format %恢复到短小数的显示格式

例 28 求解欠定方程组

$$\begin{cases} x_1 - x_2 - x_3 + x_4 = 0, \\ x_1 - x_2 + x_3 - 3x_4 = 1, \\ x_1 - x_2 - 2x_3 + x_4 = -1/2. \end{cases}$$

解 编写程序如下

```
format rat
```

```
a=[1,-1,-1,1,0;1,-1,1,-3,1;1,-1,-2,1,-1/2];
```

```
b=rref(a)
```

```
format %恢复到短小数的显示格式
```

例 29 利用表 1 的数据拟合函数

$$y = a_0 + a_1x_1^3 + a_2x_1x_2 + a_3x_2^2$$

中的参数 a_0, a_1, a_2, a_3 。

表 1

| x_1 | x_2 | y |
|-------|-------|-----|
| 5 | 2 | 1.5 |
| 5 | 5 | 2.3 |
| 7 | 10 | 3.5 |
| 8 | 4 | 5.2 |
| 8 | 6 | 1.8 |
| 3 | 3 | 2.9 |
| 7 | 8 | 3.8 |
| 7 | 3 | 4.5 |
| 2 | 6 | 4.7 |

解 拟合的 Matlab 程序如下

```
clc, clear
```

```
d=[5 2 1.5
```

```
5 5 2.3
```

```
7 10 3.5
```

```
8 4 5.2
```

```
8 6 1.8
```

```
3 3 2.9
```

```
7 8 3.8
```

```
7 3 4.5
```

```
2 6 4.7];
```



```
m=size(d,1); %获取 d 的行数
a=[ones(m,1),d(:,1).^3,d(:,1).*d(:,2),d(:,2).^2]; % 求 d(:,3)=xs*a 中的 xs;
xs=a\d(:,3) %求的多项式的系数
```

例 30 利用表 1 的数据拟合函数

$$y = a_0 e^{a_1 x_1} e^{a_2 x_2}$$

中的参数 a_0, a_1, a_2 。

解 把函数 $y = a_0 e^{a_1 x_1} e^{a_2 x_2}$ 两边取对数得 $\ln y = \ln a_0 + a_1 x_1 + a_2 x_2$, 拟合的 Matlab 程序如下

```
clc, clear
d=[5 2 1.5
5 5 2.3
7 10 3.5
8 4 5.2
8 6 1.8
3 3 2.9
7 8 3.8
7 3 4.5
2 6 4.7];
m=size(d,1);
a=[ones(m,1),d(:,1),d(:,2)]; b=log(d(:,3));
xs=a\b;
xs(1)=exp(xs(1))
```

例 31 使用用户图形界面解法命令 `sftool` 拟合例 29 和例 30 中的参数。

8.4 相似矩阵及二次型

例 32 求一个正交变换 $x = Py$, 把二次型

$$f = 2x_1x_2 + 2x_1x_3 - 2x_1x_4 - 2x_2x_3 + 2x_2x_4 + 2x_3x_4$$

化为标准形。

解 二次型的矩阵为

$$A = \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & -1 & 1 \\ 1 & -1 & 0 & 1 \\ -1 & 1 & 1 & 0 \end{bmatrix}$$

编写如下程序

```
A=[0,1,1,-1;1,0,-1,1;1,-1,0,1;-1,1,1,0];
[P,D]=eig(A)
```

P 就是所求的正交矩阵, 使得 $P^T A P = D$, 令 $x = Py$, 其中 $x = [x_1, x_2, x_3, x_4]^T$,

$$y = [y_1, y_2, y_3, y_4]^T$$

例 33 判别二次型 $f = 2x_1^2 + 4x_2^2 + 5x_3^2 - 4x_1x_2$ 的正定性，并求正交变换把二次型化成标准型。

解 Matlab 程序如下

```
a=[2 -2 0; -2 4 0; 0 0 5];
b=eig(a);
if all(b>0)
    fprintf('二次型正定\n');
else
    fprintf('二次型非正定\n');
end
[c,d]=eig(a) %c 为正交变换的变换矩阵
```

8.5 线性代数中的其它应用

例 34 已知三角形 $\triangle ABC$ 三点的坐标 $A(12,10)$ ， $B(2,6)$ ， $C(4,-10)$ ，求三角形 $\triangle ABC$ 的面积。

解法一 若三角形三顶点 (x_1, y_1) ， (x_2, y_2) ， (x_3, y_3) 呈逆时针排列，则三角形的面积为

$$S = \frac{1}{2} \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$$

计算的 Matlab 程序如下

```
clc, clear
t=[12 10 1; 2 6 1; 4 -10 1];
S=1/2*abs(det(t)) %不考虑顶点的排列，加上绝对值函数 abs。
```

解法二 利用复数运算的命令计算。

```
clc, clear
a=complex(12,10); b=complex(2,6); c=complex(4,-10); %注意复数的写法。
plot([a,b,c,a]) %画出三角形
theta=angle((b-a)/(c-a)) %利用复数运算求两个向量夹角
S=1/2*abs(b-a)*abs(c-a)*abs(sin(theta))
```

例 35 设 $\alpha_1 = \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}$, $\alpha_2 = \begin{bmatrix} -1 \\ 3 \\ 1 \end{bmatrix}$, $\alpha_3 = \begin{bmatrix} 4 \\ -1 \\ 0 \end{bmatrix}$ 试把这组向量规范正交化。

解法一 使用 Matlab 的命令 **orth**, Matlab 程序如下

```
clc, clear
```

```
format
```

```
a=[1 -1 4; 2 3 -1; -1 1 0];
```

```
b=orth(a)
```

解法二 用 Schimidt 正交化过程把 $\alpha_1, \dots, \alpha_r$ 规划正交化, 取

$$\beta_1 = \alpha_1;$$

$$\beta_1 = \alpha_2 - \frac{[\beta_1, \alpha_2]}{[\beta_1, \beta_1]} \beta_1;$$

.....

$$\beta_r = \alpha_r - \frac{[\beta_1, \alpha_r]}{[\beta_1, \beta_1]} \beta_1 - \frac{[\beta_2, \alpha_r]}{[\beta_2, \beta_2]} \beta_2 - \dots - \frac{[\beta_{r-1}, \alpha_r]}{[\beta_{r-1}, \beta_{r-1}]} \beta_{r-1}.$$

然后把 β_1, \dots, β_r 单位化, 即取

$$\gamma_1 = \frac{1}{\|\beta_1\|} \beta_1, \gamma_2 = \frac{1}{\|\beta_2\|} \beta_2, \dots, \gamma_r = \frac{1}{\|\beta_r\|} \beta_r$$

就是一组规范正交基。

编写程序实现Schimidt正交化过程, 程序如下

```
clc, clear
```

```
format
```

```
a=[1 -1 4; 2 3 -1; -1 1 0];
```

```
[m,n]=size(a); %求矩阵 a 的维数
```

```
b(:,1)=a(:,1); c(:,1)=b(:,1)/norm(b(:,1));
```

```
for i=2:n
```

```
    s=a(:,i);
```

```
    for k=1:i-1
```

```
        s=s-dot(a(:,i),c(:,k))*c(:,k); %正交化过程
```

```
    end
```

```
    c(:,i)=s/norm(s); %单位化
```

```
end
```

```
c
```

例 36: 特征多项式的提取, 在 Matlab 中提供了求矩阵特征多项式系数的函数 **c=charpoly(A)**, 返回的 c 为行向量, 其各个分量为矩阵 A 的降幂排列的特征多项式系数。可以用 poly2sym(c)获得特征多项式的解析形式。

```
A=[16 2 3 13;5 11 10 8;9 7 6 8;4 14 15 1];  
p1=charpoly(A)  
poly2sym(p1)
```

9、Matlab 在概率统计中的应用

9.1 相关概率函数

1、概率密度函数 **pdf** 系列。以 `normpdf` 为例，其调用格式为：

`y=normpdf(x,mu,sigma)`，计算参数为 `mu` 和 `sigma` 的样本数据 `x` 的正态概率密度函数值。

2、参数估计 **fit** 系列。以 `normfit` 为例，调用方法为：

`[muhat,sigmahat,muci,sigmaci]=normfit(x,alpha)`；对样本 `x` 进行参数估计，并计算置信度为 $100(1-\alpha)\%$ 的置信区间，如 `alpha=0.01` 时，则给出置信度为 99% 的置信区间，默认为 0.05。

3、`hist(x,m)` 函数，画出样本数据 `x` 的直方图，`m` 为直方图的条数，默认为 10。

4、`tabulate()` 函数，绘制频数表，返回 `table` 矩阵，第一列包含 `x` 的值，第二列包含该值出现的次数，最后一列包含每个值的百分比。

5、`h=ttest(x,m,alpha)` 函数，假设检验函数，此函数对样本数据 `x` 进行显著性水平为 `alpha` 的 `t` 假设检验，以检验正态分布样本 `x`（标准差未知）的均值是否为 `m`，`h=1` 表示拒绝零假设，`h=0` 表示不能拒绝零假设。

6、`normplot(x)` 或者 `weiplot(x)`，统计绘图函数，进行正态分布检验

7、`mean` 平均值(期望)函数，`nanmean` 求随机变量的算术平均，`geomean` 求随机变量的几何平均，`harmmean` 求随机变量的和谐平均，`trimmean` 求随机变量的调整平均。

8、`sort` 随机变量由小到大排序，`sortrows` 对随机矩阵按首行进行行排序，`range` 求随机变量值的范围，即最大与小值之差，

9、`var` 方差函数，`cov` 协方差函数，`std` 标准差函数，`skewness(x)` 偏度函数，`kurtosis(x)` 峰度函数，`corrcoef` 相关系数，

10、期望与方差函数，如 `betastat`, `chi2stat` 等，调用方法：`[M,V]=binostat(N,P)`，
`[M,V]=normstat(mu,sigma)`

11、分部名称 **name+cdf** 组成的概率累积分布函数，用来计算随机变量 $X \leq K$ 的概率之和（累积概率值）。例如 `binocdf`, `betacdf`, `chi2cdf`, `fcdf`, `normcdf` 等还有一种形式为
`P=cdf(name,X,A1,A2,A3)` 估算名称为 `name` 的累积分布函数

12、逆累积分布函数 `icdf()`，已知 $F(x) = P\{X \leq x\}$ ，求 `x`。用法：`icdf('name',P,`

a1, a2, a3), 返回分布为 name, 参数为 a1,a2,a3, 累积概率值为 P 的临界值。

13、分位点函数。已知分布及分布中的一点, 求此点处的概率值要用到概率累积函数(cdf); 当已知概率值而需要求对应该概率的分布点时, 需要用到逆概率累积函数 inv 逆概率累积函数的组成由分布的名称 name+inv 组合而成, 例如

binoinv,betainv,chi2inv,finv,norminv 等假设检验中经常用到求分位点的问题

14、随机数生成函数。常用的有: (1) betarnd (2) binornd (3) chi2rnd (4) frnd

(5)geornd (6)hygernd (7)mvnrnd 多元正态分布 (8)trnd (9)unifrnd

15、用于区间估计的函数。

(1) Gauss-Newton 法的非线性最小二乘数据拟合

nlinfit(X,Y,'MODEL',BETA0)返回 MODEL 描述的类型非线性方程的系数, MODEL 为用户提供形如 $y=f(\beta,x)$ 的函数。

[BETA,R,J]=nlinfit(X,Y,'MODEL',BETA0)返回拟合系数 BETA, 残差 R 和 Jacobi 矩阵 J

(2)非线性拟合和预测的交互图形工具 **nlintool(X,Y,MODEL,BETA0,ALPHA)**返回 (X,Y)数据的非线性曲面拟合的预测图, 它把预测值的(1-ALPHA)通用置信区间绘制成两个红曲面, BETA0 为参数的初始预测值向量, ALPHA 默认为 0.05

nlintool(X,Y,MODEL,BETA0,ALPHA,XNAME,YNAME)

(3) 非线性最小二乘预测的置信区间. **nlpredci** 非线性最小二乘预测的置信区间

[YPRED,delta]=nlpredci(MODEL,INPUTS,X,F,J)返回给定残差 F 和 Jacobi 矩阵的 X 点的最小二乘预测值 YPRED 和 95%的置信区间。

(4) 非线性模型的参数置信区间, **CI=nlparci(X,F,J)**返回置信区间

(5) 非负最小二乘法。**x=nnls(A,b)**返回在 $x \geq 0$ 的条件下使得 $\text{norm}(Ax-b)$ 最小的向量 x, A, b 必须为实矩阵或向量。 **[x,w]=nnls(A,b)**

9.2 常见的概率分布密度函数

常见概率分布密度函数有:

Beta(beta),Binomial(bio),Chisquare(chi2),Exponential(exp),F(f),Gamma(gam),Geometric(geo),Hypergeometric(hyge),Lognormal(对数正态分布 logn),Negative Binomial(nbin),Nocentral F(非中心 F 分布 ncf),Nocentral t(nct),Nocentral Chi-square(ncx2), Normal(正态分布 norm),Poisson, Rayleigh(瑞利分布 rayl),T,Uniform(unif),Discrete

Uniform(unid), Weibull(weib)。

示例 1：正态分布示例

```
clear all;close all;clc;
x=[-5:0.02:5]'
y1=[];y2=[];
mu1=[-1 0 0 0 1];
sig1=sqrt([1,0.1,1,10,1]);
for i=1:length(mu1)
    y1=[y1,normpdf(x,mu1(i),sig1(i))]; % 概率密度函数
    y2=[y2,normcdf(x,mu1(i),sig1(i))]; % 累积分布函数
end
plot(x,y1);
figure;
plot(x,y2);
```

示例 2：连续均匀分布

```
clear all;close all;clc;
x=-10:0.01:10;
r=1;
y=unifpdf(x,0,2*pi*r);
plot(x,y,'r-')
```

示例 3：指数分布

```
clear all;close all;clc;
x=0:0.1:30;
y=expdf(x,4);
plot(x,y,'m-')
```

示例 4：几何分布

```
clear all;close all;clc;
x=0:30;
y=geopdf(x,0.5);
plot(x,y,'b')
```

示例 5：二项分布

```
clear all;close all;clc;
x=0:50;
y=binopdf(x,0.5);
```

```
plot(x,y,'r*')
```

示例 6: 绘制参数=1,2,5,10 时 Poission 分布的概率密度函数与分布函数曲线

```
clear all;close all;clc;  
x=[0:15]';  
y1=[];  
y2=[];  
lam1=[1 2 5 10];  
n=length(lam1);  
for i=1:n  
    y1=[y1,poisspdf(x,lam1(i))];  
    y2=[y2,poisscdf(x,lam1(i))];  
end  
stem(x,y1);  
line(x,y1);  
figure;  
plot(x,y2);
```

示例 7: 离散均匀分布

```
clear all;close all;clc;  
n=20;x=1:n  
y=unidpdf(x,0,2*pi*r);  
plot(x,y,'o-')
```

示例 8: 卡方分布

```
clear all;close all;clc;  
x=0:0.1:20;y1=chi2pdf(x,4);figure(1);plot(x,y1);  
y2=chi2pdf(x,10);figure(2);  
plot(x,y2)
```

示例 9: F 分布

```
clear all;close all;clc;  
x=0.01:0.1:8.01;  
y=fpdf(x,10); plot(x,y,'r:');
```

示例 10: t 分布

```
clear all;close all;clc;  
x=-6:0.01:6;  
y=tpdf(x,10); plot(x,y,'r:');
```


示例 11: 分别绘制出 b 为 0.5,1,3,5 时 Rayleigh 分布的概率密度函数与分布函数曲线。

```
x=[-eps:0.02:-0.05;0:0.02:5];
x=sort(x');
b1=[.5,1,3,5];
y1=[];
y2=[];
for i=1:length(b1)
    y1=[y1,raylpdf(x,b1(i))];
    y2=[y2,raylcdf(x,b1(i))];
end
plot(x,y1);
for i=1:length(b1)
    str{i}=['b=',num2str(b1(i))];
end
legend(str);
figure,plot(x,y2)
```

示例 12: 在标准正态分布表中, 若已知 $F(x) = 0.975$, 求 x

$x = \text{icdf('norm',0.975,0,1)}$

示例 13: 参数估计函数 mle

$X = \text{binornd}(20,0.75);$

$[p,pci] = \text{mle('bino',X,0.05,20)}$ %求概率的估计值和置信区间, 置信度为 95%

9.3 一类概率问题的求解

随机变量 ξ 分布函数 $F(x)$ 的物理意义是该随机变量 ξ 落入 $(-\infty, x)$ 区间的概率, 故可以利用分布函数的概念求取满足条件的概率。要求出 ξ 落入区间 $[x_1, x_2]$ 的概率

$P[x_1 \leq \xi \leq x_2]$, 可以利用两个分布函数之差求出, 需要用的概率公式如下所示:

$$\begin{cases} P[\xi \leq x] = F(x) \\ P[x_1 \leq \xi \leq x_2] = F(x_2) - F(x_1) \\ P[\xi \geq x] = 1 - F(x) \end{cases}$$

示例 1: 假设某随机变量 x 满足 Rayleigh 分布, 且 $b=1$, 是分别求出该随机变量 x 值落入区间 $[0.2, 2]$ 及区间 $[1, \infty]$

```
b=1;
p1=raylcdf(0.2,b);
p2=raylcdf(2,b);
P=p2-p1
```

示例 2: 假设 A, B 两地间有 6 个交通岗, 在各个交通岗处遇到红灯的概率均相同, 为 $p=1/3$, 且中途遇到红灯的次数满足二项分布 $B(6,p)$, 是求出某人从 A 地出发到 B 地至

少遇到一次红灯的概率。若选择不同的 p 值，再绘制出至少遇到一次红灯的概率曲线。

解：

```
x=0:6;
y=binopdf(x,6,1/3);
P1=1-y(1); %1 减去不遇红灯的概率（遇到次数为 0）
P2=sum(y(2:end));% 另一种计算方法
p0=0.05:0.05:0.95;
y=[];
for p=p0
    y=[y,1-binopdf(0,6,p)];
end
plot(p0,y,1/3,P1,'o')
```

9.4 概率统计作图

1、通用函数 cdfplot()

格式: cdfplot(X) %作样本 X（向量）的累积分布函数图形

h = cdfplot(X) %h 返回曲线的句柄

[h,stats] = cdfplot(X) %stats 表示样本的一些特征

示例：

```
X=normrnd (0,1,50,1);
```

```
[h,stats]=cdfplot(X)
```

输出结果：

```
h =
```

```
3.0013
```

```
stats =
```

```
min: -1.8740 %样本最小值
```

```
max: 1.6924 %最大值
```

```
mean: 0.0565 %平均值
```

```
median: 0.1032 %中间值（50%处的数据）
```

```
std: 0.7559 %样本标准差
```

2、专用函数

1) 绘制正态分布概率图形 normplot()

格式: normplot(X) %若 X 为向量，则显示正态分布概率图形，若 X 为矩阵，则显示每一列的正态分布概率图形。

注意：说明样本数据在图中用“+”显示；如果数据来自正态分布，则图形显示为直线，而其它分布可能在图中产生弯曲。

示例:

```
X=normrnd(0,1,50,1);
```

```
normplot(X)
```

3、绘制多项式曲线 refcurve()

格式: `h = refcurve(p)` %在图中加入一条多项式曲线, `h` 为曲线的句柄, `p` 为多项式系数向量, `p=[p1,p2, p3,...,pn]`, 其中 `p1` 为最高幂项系数。

示例: 火箭的高度与时间图形, 加入一条理论高度曲线, 火箭初速为 100m/秒。

```
h = [85 162 230 289 339 381 413 437 452 458 456 440 400 356];
```

```
plot(h,'+')
```

```
refcurve([-4.9 100 0])
```

4、给当前图形加一条参考线 reffline()

格式: `reffline(slope,intercept)` % `slope` 表示直线斜率, `intercept` 表示截距

`reffline(slope)` `slope=[a b]`, 图中加一条直线: $y=b+ax$ 。

示例:

```
y = [3.2 2.6 3.1 3.4 2.4 2.9 3.0 3.3 3.2 2.1 2.6]';
```

```
plot(y,'+')
```

```
reffline(0,3)
```

5、样本的概率图形 capaplot

格式: `p = capaplot(data,specs)` %`data` 为所给样本数据, `specs` 指定范围, `p` 表示在指定范围内的概率。

注意: 该函数返回来自于估计分布的随机变量落在指定范围内的概率

示例:

```
>data=normrnd (0,1,30,1);
```

```
p=capaplot(data, [-2,2])
```

6、附加有正态密度曲线的直方图 histfit()

格式: `histfit(data)` %`data` 为向量, 返回直方图和正态曲线。

`histfit(data,nbins)` % `nbins` 指定 bar 的个数, 缺省时为 `data` 中数据个数的平方根。

示例:

```
r = normrnd (10,1,100,1);
```

```
histfit(r)
```

7、在指定的界线之间画正态密度曲线 normspec()

格式 `p = normspec(specs,mu,sigma) %specs` 指定界线，`mu,sigma` 为正态分布的参数 `p` 为样本落在上、下界之间的概率。

示例：

```
normspec([10 Inf],11.5,1.25)
```

10、Matlab 图论工具箱

Matlab 图论工具箱的命令见表 2。

表 2 Matlab 图论工具箱的相关命令

| 命令名 | 功能 |
|-----------------------|---------------------------|
| graphallshortestpaths | 求图中所有顶点对之间的最短距离 |
| graphconncomp | 找无向图的连通分支，或有向图的强（弱）连通分支 |
| graphisdag | 测试有向图是否含有圈，不含圈返回 1，否则返回 0 |
| graphisomorphism | 确定两个图是否同构，同构返回 1，否则返回 0 |
| graphisspanntree | 确定一个图是否是生成树，是返回 1，否则返回 0 |
| graphmaxflow | 计算有向图的最大流 |
| graphminspanntree | 在图中找最小生成树 |
| graphpred2path | 把前驱顶点序列变成路径的顶点序列 |
| graphshortestpath | 求图中指定的一对顶点间的最短距离和最短路径 |
| graphtopoorder | 执行有向无圈图的拓扑排序 |
| graphtraverse | 求从一顶点出发，所能遍历图中的顶点 |

下面给出图论工具箱在最短路，最小生成树和最大流中的应用例子。Matlab 图论工具箱使用的数据结构是稀疏矩阵。普通矩阵和稀疏矩阵相互转换的命令是 **sparse** 和 **full**。

10.1 图的矩阵表示与绘制

在图论中，图是由节点和边构成的，边是指两个节点的直接路径，如果边是有向的，则图称为有向图，否则称为无向图。假设一个图有 n 个节点，则可以用一个 $n*n$ 的矩阵 R 来表示它，设由节点 i 到节点 j 的边权值为 k ，即 $R(i,j)=k$ ，这样的矩阵称为关联矩阵。若 i 和 j 之间不存在边，则令 $R(i,j)=0$ 。如果图是稀疏的，Matlab 中可以使用稀疏矩阵来表示。

示例 1:

```
a=[1 2 3 4 5 2]
b=[2 3 4 1 2 5];
w=[1 2 3 4 5 6];
R=sparse(a,b,w)
F=full(R); %变成常规矩阵
```

生物信息学工具箱中提供了建立有向图的现成函数 `biograph()` 和 `view()`，调用方法如下：

```
BGobj = biograph(CMatrix)
```

```
BGobj = biograph(CMatrix, NodeIDs)
view(BGobj) %显示图形结构。
BGobjHandle = view(BGobj)
```

示例 2:

```
ab=[1 1 2 2 3 3 4 4 4 4 5 6 6 7 8];
bb=[2 3 5 4 4 6 5 7 8 6 7 8 9 9 9];
w=[1 2 12 6 3 4 4 15 7 2 7 7 15 3 10];
R=sparse(ab,bb,w);
R(9,9)=0;
P=biograph(R,[],'ShowWeights','on');
h=view(P);
```

10.2 几个函数的简要介绍

1、图的最短路径 **graphallshortestpaths** 函数的命令格式:

```
[dist] = graphallshortestpaths(G)
[dist] = graphallshortestpaths(G, ...'Directed', DirectedValue, ...)
[dist] = graphallshortestpaths(G, ...'Weights', WeightsValue, ...)
```

G 是代表一个图的 $N \times N$ 稀疏矩阵, 矩阵中的非零值代表一条边的权值; **DirectedValue** 属性表示图是否为有向图, **false** 代表无向图, **true** 代表有向图, 默认为 **true**。 **WeightsValue** 是矩阵 G 中边的自定义权值列向量, 该属性可以使我们使用零权值。输出 **[dist]** 是一个 $N \times N$ 的矩阵, 每一个元素代表两点之间最短距离, 对角线上的元素总为零, 不在对角线上的零表示起点和终点的距离为 0, **inf** 值表示没有路径。

示例 1: 有向图

(1) 创建并显示一个含有 6 个结点 11 条边的有向图

```
W = [.41 .99 .51 .32 .15 .45 .38 .32 .36 .29 .21]; %权值向量
DG = sparse([6 1 2 2 3 4 4 5 5 6 1],[2 6 3 5 4 1 6 3 4 3 5],W) %构造的稀疏矩阵表示图
h=view(biograph(DG,[],'ShowWeights','on')); %显示图的结构
```

(2) 寻找有向图中每对结点之间的最短路径

```
dist= graphallshortestpaths(DG)
```

示例 2: 无向图

(1) 创建无向图, 结点信息同上

```
UG=tril(DG+DG'); %要求生成的无向图是一个下三角矩阵
```

```
view(biograph(UG,[],'ShowArrows','off','ShowWeights','on'))
```

(2) 求每对顶点之间的最短路径

```
dist= graphallshortestpaths(UG,'directed',false)
```

2、最小生成树 **graphminspantree** 函数的调用格式:

```
[Tree, pred] = graphminspantree(G)
```

```
[Tree, pred] = graphminspantree(G, R)
```

```
[Tree, pred] = graphminspantree(..., 'Method', MethodValue, ...)
```

```
[Tree, pred] = graphminspantree(..., 'Weights', WeightsValue, ...)
```

该函数用来寻找一个无循环的节点集合，连接无向图的全部节点，并且总的权值最小。Tree 是一个代表生成树的稀疏矩阵，pred 是包含最小生成的祖先节点的向量。G 是无向图，R 代表根节点，取值为 1 到节点数目，Method 可以选择 'Kruskal'，'Prim' 算法。

示例 1:

(1) 构造无向图

```
W = [.41 .29 .51 .32 .50 .45 .38 .32 .36 .29 .21];
```

```
DG = sparse([1 1 2 2 3 4 4 5 5 6 6],[2 6 3 5 4 1 6 3 4 2 5],W);
```

```
UG = tril(DG + DG')
```

```
view(biograph(UG,[],'ShowArrows','off','ShowWeights','on'))
```

(2) 调用函数求无向图的最小生成树

```
[ST,pred] = graphminspantree(UG)
```

```
h=view(biograph(ST,[],'ShowArrows','off','ShowWeights','on'))
```

```
Edges=getedgesbynodeid(h); %提取无向图 h 的边集
```

```
set(Edges,'LineColor',[0 0 0]);
```

```
set(Edges,'LineWidth',1.5); %设置边值属性
```

3、计算有向图的最大流 **graphmaxflow**，函数格式为:

```
[MaxFlow, FlowMatrix, Cut]=graphmaxflow(G, SNode, TNode)
```

```
[...] = graphmaxflow(G, SNode, TNode, ...'Capacity', CapacityValue, ...)
```

```
[...] = graphmaxflow(G, SNode, TNode, ...'Method', MethodValue, ...)
```

G 是 N*N 的稀疏矩阵，表示有向图，SNode 和 TNode 都是 G 中的节点，分别表示起点和目标点，CapacityValue 为每条边自定义容量的列向量；MethodValue 可以取 'Edmonds' 和 'Goldberg' 算法。输出 MaxFlow 表示最大流，FlowMatrix 是表示每条边

数据流值的稀疏矩阵，Cut 表示连接 SNode 到 TNode 的逻辑行向量，如果有多个解时，Cut 是一个矩阵。

注意：该函数只能解决权值都为正值，并且两个顶点之间不能有两条弧的问题。有时对不满足该条件的有向图添加节点的方法进行转化。

示例 2:

(1) 构造带有节点和边的有向图

```
cm = sparse([1 1 2 2 3 3 4 5],[2 3 4 5 4 5 6 6],[2 3 3 1 1 1 2 3],6,6); %6 个节点，8 条边
```

(2) 计算第 1 个到第 6 个节点的最大流

```
[M,F,K]=graphmaxflow(cm,1,6);
```

(3)显示原始有向图的图结构

```
h0=view(biograph(cm,[],'ShowWeights','on'));
```

(4)显示计算最大流矩阵的图结构

```
h1=view(biograph(F,[],'ShowWeights','on'));
```

(5)在原始图结构中标注求解结果

```
set(h0.Nodes(K(1,:)),'Color',[1 0 0]);
```

4、图的遍历函数 **graphtraverse**，命令格式如下：

```
[disc, pred, closed]=graphtraverse(G,S)
```

```
[...] = graphtraverse(G, S, ...'Depth', DepthValue, ...)
```

```
[...] = graphtraverse(G, S, ...'Directed', DirectedValue, ...)
```

```
[...] = graphtraverse(G, S, ...'Method', MethodValue, ...)
```

G 是有向图，S 为起始节点，disc 是节点索引向量，pred 是祖先节点索引向量

示例 3:

(1) 创建一个有向图：

```
DG = sparse([1 2 3 4 5 5 5 6 7 8 8 9], [2 4 1 5 3 6 7 9 8 1 10 2],true,10,10)
```

```
h = view(biograph(DG))
```

(2) 使用深度优先算法从第 4 个节点开始遍历

```
order = graphtraverse(DG,4)
```

(3) 标记遍历结果

```
for i = 1:10
```

```
    h.Nodes(order(i)).Label =sprintf('%s:%d',h.Nodes(order(i)).ID,i);
```

```
end
```



```

h.ShowTextInNodes = 'label'
dolayout(h)
(4) 使用广度优先遍历
order = graphtraverse(DG,4,'Method','BFS')
(5) 标记遍历结果
for i = 1:10
    h.Nodes(order(i)).Label =sprintf('%s:%d',h.Nodes(order(i)).ID,i);
end
h.ShowTextInNodes = 'label'
dolayout(h)
(6) 标记与节点 4 邻近的深度为 2 的节点
node_idx = graphtraverse(DG,4,'depth',2)
set(h.nodes(node_idx),'Color',[1 0 0])

```

10.3 最短路径问题

示例：在 15 个顶点的无向图中，每对顶点之间以概率 0.65 存在一条权重为[2 10]上随机整数的边，首先生成该图。然后求解下列问题：

- (1) 求顶点 v1 到顶点 v11 的最短距离及最短路径。
- (2) 求所有顶点对之间的最短距离。

解 求解的 Matlab 程序如下

```

clc, clear
a=rand(15); a=tril(a); %截取下三角元素
a(1:16:end)=0; %对角线元素置 0;
randnum=randint(15,15,[2,10]); %生成 15×15 的随机整数矩阵
b=(a>=0.35).*randnum; %生成赋权图的邻接矩阵
c=b+b'; %生成完整的邻接矩阵
b=sparse(b); %生成邻接矩阵为下三角阵的稀疏矩阵
[d1,path]=graphshortestpath(b,1,11,'directed',false) %求顶点 1 到 11 的最短路
d2=graphallshortestpaths(b,'directed',false) %求所有顶点对之间的最短距离
save data1 b; %把稀疏矩阵 b 保存到 data1.mat 中，供以后使用
dlmwrite('data2.txt',c) %把矩阵 c 保存在纯文本文件 data2.txt，供以后调用

```

10.4 最小生成树问题

示例：对于例 46 生成的无向图，求它的最小生成树。

解 Matlab 程序如下

```

clc, clear
load data1

```

```
[st,pred]=graphminspantree(b,'Method','Kruskal') %求最小生成树
view(biograph(st,[],'ShowArrows','off','ShowWeights','on')) %画出最小生成树
```

10.5 最大流问题

例 48 图 3 的有向图中弧上的数字表示容量，求从 v_s 到 v_t 的最大流量。

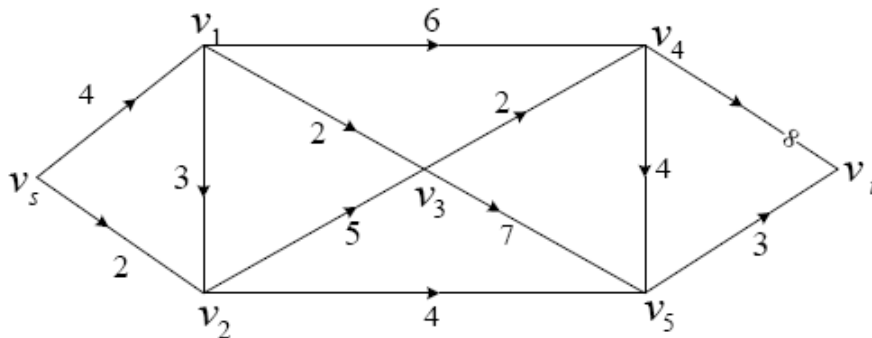


图 3 有向图

解 求最大流的 Matlab 程序如下

```
clc, clear
a=zeros(7);
a(1,[2 3])=[4 2];
a(2,[3:5])=[3 2 6];
a(3,[4 6])=[5 4];
a(4,[5 6])=[2 7];
a(5,[6 7])=[4 8];
a(6,7)=3;
b=sparse(a); %构造稀疏矩阵
[M,F,K]=graphmaxflow(b,1,7)
view(biograph(b,[],'ShowArrows','on','ShowWeights','on')) %画有向图
view(biograph(F,[],'ShowWeights','on')) %画最大流
```

11、评价方法

评价方法大体上可分为两类，其主要区别在确定权重的方法上。一类是**主观赋权法**，多数采取综合咨询评分确定权重，如**综合指数法**、**模糊综合评判法**、**层次分析法**、**功效系数法**等。另一类是**客观赋权法**，根据各指标间相关关系或各指标值变异程度来确定权重，如**主成分分析法**、**因子分析法**、**理想解法**（也称 TOPSIS 法）等。目前国内外综合评价方法有数十种之多，其中主要使用的评价方法有主成分分析法、因子分析、TOPSIS、秩和比法、灰色关联、熵权法、层次分析法、模糊评价法、灰色理论法、物元分析法、聚类分析法、价值工程法、神经网络法等。

11.1 理想解法

理想解法的具体算法如下

(1) 用向量规范化的方法求得规范决策矩阵

设多属性决策问题的决策矩阵 $A = (a_{ij})_{m \times n}$ ，规范化决策矩阵 $B = (b_{ij})_{m \times n}$ ，其中

$$b_{ij} = a_{ij} / \sqrt{\sum_{i=1}^m a_{ij}^2}, \quad i = 1, 2, \dots, m, \quad j = 1, 2, \dots, n. \quad (1)$$

(2) 构成加权规范阵 $C = (c_{ij})_{m \times n}$

设由决策人给定各属性的权重向量为 $w = [w_1, w_2, \dots, w_n]^T$ ，则

$$c_{ij} = w_j \cdot b_{ij}, \quad i = 1, 2, \dots, m, \quad j = 1, 2, \dots, n. \quad (2)$$

(3) 确定正理想解 C^* 和负理想解 C^0

设正理想解 C^* 的第 j 个属性值为 c_j^* ，负理想解 C^0 第 j 个属性值为 c_j^0 ，则

$$\text{正理想解 } c_j^* = \begin{cases} \max_i c_{ij}, & j \text{ 为效益型属性,} \\ \min_i c_{ij}, & j \text{ 为成本型属性,} \end{cases} \quad j = 1, 2, \dots, n. \quad (3)$$

$$\text{负理想解 } c_j^0 = \begin{cases} \min_i c_{ij}, & j \text{ 为效益型属性,} \\ \max_i c_{ij}, & j \text{ 为成本型属性,} \end{cases} \quad j = 1, 2, \dots, n. \quad (4)$$

(4) 计算各方案到正理想解与负理想解的距离

备选方案 d_i 到正理想解的距离为

$$s_i^* = \sqrt{\sum_{j=1}^n (c_{ij} - c_j^*)^2}, \quad i = 1, 2, \dots, m. \quad (5)$$

备选方案 d_i 到负理想解的距离为

$$s_i^0 = \sqrt{\sum_{j=1}^n (c_{ij} - c_j^0)^2}, \quad i = 1, 2, \dots, m. \quad (6)$$

(5) 计算各方案的排队指标值（即综合评价指数）

$$f_i^* = s_i^0 / (s_i^0 + s_i^*), \quad i = 1, 2, \dots, m. \quad (7)$$

(6) 按 f_i^* 由大到小排列方案的优劣次序。

例 49 研究生评估问题，数据如下表 1 所示

| | 人均专著 x1(本/人) | 生师比 x2 | 经费 x3(万元/年) | 逾期毕业率 x4 (%) |
|---|--------------|--------|-------------|--------------|
| 1 | 0.1 | 5 | 5000 | 4.7 |
| 2 | 0.2 | 6 | 6000 | 5.6 |
| 3 | 0.4 | 7 | 7000 | 6.7 |
| 4 | 0.9 | 10 | 10000 | 2.3 |
| 5 | 1.2 | 2 | 400 | 1.8 |

解：

第一步：数据预处理，即属性值的规范化，属性值一般具有三种类型，效益型、成本型和区间型。对多属性决策评估问题可能三种类型同时出现，因此需要进行规范化，去掉量纲并进行数据的归一化处理。常用的属性规范化方法有：

(1) 线性变换

原始的决策变换矩阵为 $A = (a_{ij})_{m \times n}$ ，变换后的决策矩阵为 $B = (b_{ij})_{m \times n}$ ，设 a_j^{\max} 是决策矩阵第 j 列的最大值， a_j^{\min} 对应最小值，若 x_j 是效益型，则 $b_{ij} = a_{ij} / a_j^{\max}$ ，若 x_j 为成本型属性，则令 $b_{ij} = 1 - a_{ij} / a_j^{\max}$

(2) 标准 0-1 变换

对效益型属性，令 $b_{ij} = \frac{a_{ij} - a_j^{\min}}{a_j^{\max} - a_j^{\min}}$ ；对成本型属性，令 $b_{ij} = \frac{a_j^{\max} - a_{ij}}{a_j^{\max} - a_j^{\min}}$

(3) 区间型属性的变换

设给定的最优属性区间为 $[a_j^0, a_j^*]$, a_j' 为无法容忍下限, a_j'' 为无法容忍上限, 则

$$b_{ij} = \begin{cases} 1 - (a_j^0 - a_{ij}) / (a_j^0 - a_j') & a_j' \leq a_{ij} < a_j^0 \\ 1, & a_j^0 \leq a_{ij} \leq a_j^* \\ 1 - (a_{ij} - a_j^*) / (a_j'' - a_j^*) & a_j^* < a_{ij} \leq a_j'' \\ 0, & \text{其他} \end{cases}$$

设生师比最优区间为 $[5, 6]$, $a_2'=2, a_2''=12$, 则属性 2 的数据处理结果为:

| | 生师比 x2 | 处理的结果 |
|---|--------|--------|
| 1 | 5 | 1 |
| 2 | 6 | 1 |
| 3 | 7 | 0.8333 |
| 4 | 10 | 0.3333 |
| 5 | 2 | 0 |

使用 Matlab 计算的程序为:

```
x2=@(qujian,lb,ub,x)((1-(qujian(1)-x)./(qujian(1)-lb))).*(x>=lb & x<qujian(1))+
(x>=qujian(1) & x<=qujian(2))+((1-(x-qujian(2))./(ub-qujian(2))).*(x>qujian(2) & x<=ub));
qujian=[5,6];
lb=2;ub=12;
x2data=[5,6,7,10,2]';
y2=x2(qujian,lb,ub,x2data)
```

(4) 向量的规范化, 变换公式为 $b_{ij} = a_{ij} / \sqrt{\sum_{i=1}^m a_{ij}^2}$, $i=1, \dots, m, j=1, \dots, n$, 规范化后,

各方案的同一属性值的平方和为 1, 这种规范化常用于计算各方案与某种虚拟方案的欧氏距离的场合。

(5) 标准化处理, 为了消除量纲的影响, 使每个变量都具有同等的表现力, 处理方法是:

$$b_{ij} = \frac{a_{ij} - \bar{a}_j}{s_j}, \text{其中 } \bar{a}_j = \frac{1}{m} \sum_{i=1}^m a_{ij}, s_j = \sqrt{\frac{1}{m-1} \sum_{i=1}^m (a_{ij} - \bar{a}_j)^2}$$

用于计算的 Matlab 程序为:

```
x=[0.1 5 5000 4.7;0.2 6 6000 5.6;0.4 7 7000 6.7;0.9 10 10000 2.3;1.2 2 400 1.8]
```

$y = \text{zscore}(x)$

第二步：设权向量为 $w = [0.2 \ 0.3 \ 0.4 \ 0.1]$ ，利用公式（2）计算加权的向量规范化属性矩阵；

第三步：根据公式（3）~（4）计算正负理想解 C^*, C^0

第四步：利用公式（5）~（6）计算各方案到正负理想解的距离 s_i^*, s_i^0

第五步：利用公式（7）计算排队指标，确定方案的优劣。

总的 Matlab 计算程序如下：

```
clc, clear
a=[0.1  5   5000   4.7
    0.2  6   6000   5.6
    0.4  7   7000   6.7
    0.9  10  10000   2.3
    1.2  2   400     1.8];
[m,n]=size(a);
x2=@(qujian,lb,ub,x)(1-(qujian(1)-x)./(qujian(1)-lb)).*(x>=lb &
x<qujian(1))+(x>=qujian(1) & x<=qujian(2)).*(x>qujian(2) & x<=ub); //定义匿名函数
qujian=[5,6]; lb=2; ub=12;
a(:,2)=x2(qujian,lb,ub,a(:,2)); %对属性 2 进行变换
for j=1:n
    b(:,j)=a(:,j)/norm(a(:,j)); %向量规范化
end
w=[0.2 0.3 0.4 0.1];
c=b.*repmat(w,m,1); %求加权矩阵
Cstar=max(c); %求正理想解
Cstar(4)=min(c(:,4)) %属性 4 为成本型的
C0=min(c); %q 求负理想解
C0(4)=max(c(:,4)) %属性 4 为成本型的
for i=1:m
    Sstar(i)=norm(c(i,:)-Cstar); %求到正理想解的距离
    S0(i)=norm(c(i,:)-C0); %求到负理想的距离
end
f=S0./(Sstar+S0);
[sf,ind]=sort(f,'descend') %求排序结果
```

例 50 已知 10 个人 4 门课的成绩见表 3，试对这 10 个人的总成绩进行排序。

表 3 成绩表

| 序号 | 语文 | 数学 | 英语 | 计算机 |
|----|----|----|----|-----|
|----|----|----|----|-----|

| | | | | |
|----|----|----|----|----|
| 1 | 88 | 61 | 81 | 83 |
| 2 | 92 | 94 | 56 | 56 |
| 3 | 60 | 94 | 89 | 66 |
| 4 | 92 | 74 | 93 | 56 |
| 5 | 80 | 87 | 82 | 58 |
| 6 | 58 | 60 | 86 | 88 |
| 7 | 66 | 72 | 85 | 83 |
| 8 | 77 | 92 | 71 | 68 |
| 9 | 94 | 87 | 81 | 93 |
| 10 | 94 | 94 | 62 | 56 |

解 求解的 Matlab 程序如下

```
clc, clear
a=load('data3.txt'); %将表格中的数据提前存入 data3.txt 中
amax=max(a); %求每一列的最大值,即求正理想
amin=min(a); %求每一列的最小值,即求负理想
m=size(a,1); %求矩阵 a 的行数
for i=1:m
    dstar(i)=norm(a(i,:)-amax); %求到正理想的距离
    d0(i)=norm(a(i,:)-amin); %求到负理想的距离
end
f=d0./(dstar+d0); %求评价的指标值
[sf,ind]=sort(f,'descend') %指标值按照从大到小排列
```

11.2 层次分析法

层次分析法（Analytic Hierarchy Process，简称 AHP）是对一些较为复杂、较为模糊的问题作出决策的简易方法，它特别适用于那些**难于完全定量分析**的问题。它是美国运筹学家 T. L. Saaty 教授于上世纪 70 年代初期提出的一种简便、灵活而又实用的多准则决策方法。但实际使用过程中有太多的主观成分，并不常用。

1、层次分析法的基本原理与步骤

运用层次分析法建模，大体上可按下面四个步骤进行：

- (i) 建立递阶层次结构模型；
- (ii) 构造出各层次中的所有**判断矩阵**；
- (iii) 层次单排序及一致性检验；
- (iv) 层次总排序及一致性检验。

下面分别说明这四个步骤的实现过程。

1) 递阶层次结构的建立与特点

应用 AHP 分析决策问题时，首先要把问题条理化、层次化，构造出一个有层次的结构模型。在这个模型下，复杂问题被分解为元素的组成部分。这些元素又按其属性及关系形成若干层次。上一层次元素作为准则对下一层次有关元素起支配作用。

这些层次可以分为三类：

(i) 最高层：这一层次中只有一个元素，一般它是分析问题的预定目标或理想结果，因此也称为**目标层**。

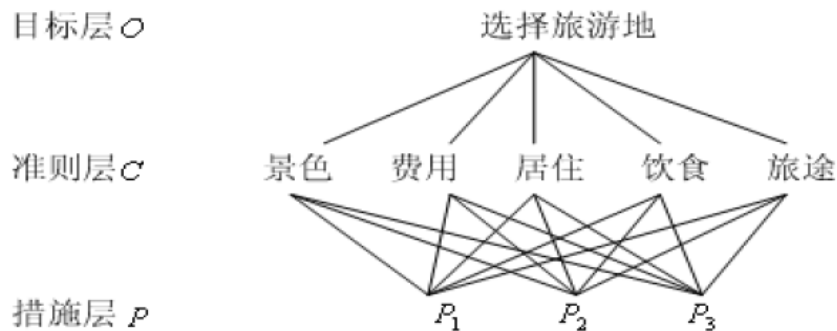
(ii) 中间层：这一层次中包含了为实现目标所涉及的中间环节，它可以由若干个层次组成，包括所需考虑的准则、子准则，因此也称为**准则层**。

(iii) 最底层：这一层次包括了为实现目标可供选择的各种措施、决策方案等，因此也称为**措施层或方案层**。

递阶层次结构中的层次数与问题的复杂程度及需要分析的详尽程度有关，一般地层次数不受限制。每一层次中各元素所支配的元素一般不要超过 9 个。这是因为支配的元素过多会给两两比较判断带来困难。

示例：

假期旅游有 P_1 、 P_2 、 P_3 3 个旅游胜地供你选择，试确定一个最佳地点。



2) 构造判断矩阵

层次结构反映了因素之间的关系，但准则层中的各准则在目标衡量中所占的比重并不一定相同，在决策者的心目中，它们各占有一定的比例。

在确定影响某因素的诸因子在该因素中所占的比重时，遇到的主要困难是这些比重常常**不易定量化**。此外，当影响某因素的因子较多时，直接考虑各因子对该因素有多大程度的影响时，常常会因考虑不周全、顾此失彼而使决策者提出与他实际认为的重要性程度不相一致的数据，甚至有可能提出一组隐含矛盾的数据。

设现在要比较 n 个因子 $X = \{x_1, \dots, x_n\}$ 对某因素 Z 的影响大小, 怎样比较才能提供可信的数据呢? Saaty 等人建议可以采取对因子进行两两比较建立成对比较矩阵的办法。即每次取两个因子 x_i 和 x_j , 以 a_{ij} 表示 x_i 和 x_j 对 Z 的影响大小之比, 全部比较结果用矩阵 $A = (a_{ij})_{n \times n}$ 表示, 称 A 为 $Z-X$ 之间的成对比较判断矩阵 (简称判断矩阵)。容易看出, 若 x_i 与 x_j 对 Z 的影响之比为 a_{ij} , 则 x_j 与 x_i 对 Z 的影响之比应为

$$a_{ji} = \frac{1}{a_{ij}}。$$

关于如何确定 a_{ij} 的值, Saaty 等建议引用数字 1~9 及其倒数作为标度。

| 标度 | 含 义 |
|------------|--|
| 1 | 表示两个因素相比, 具有相同重要性 |
| 3 | 表示两个因素相比, 前者比后者稍重要 |
| 5 | 表示两个因素相比, 前者比后者明显重要 |
| 7 | 表示两个因素相比, 前者比后者强烈重要 |
| 9 | 表示两个因素相比, 前者比后者极端重要 |
| 2, 4, 6, 8 | 表示上述相邻判断的中间值 |
| 倒数 | 若因素 i 与因素 j 的重要性之比为 a_{ij} , 那么因素 j 与因素 i 重要性之比为 $a_{ji} = 1/a_{ij}$ 。 |

3) 层次单排序及一致性检验

判断矩阵 A 对应于最大特征值 λ_{\max} 的特征向量 W , 经归一化后即为同一层次相应因素对于上一层次某因素相对重要性的排序权值, 这一过程称为层次单排序。

上述构造成对比较判断矩阵的办法虽能减少其它因素的干扰, 较客观地反映出一对因子影响力的差别。但综合全部比较结果时, 其中难免包含一定程度的非一致性。如果比较结果是前后完全一致的, 则矩阵 A 的元素还应当满足:

$$a_{ij}a_{jk} = a_{ik}, \quad \forall i, j, k = 1, 2, \dots, n$$

满足上述关系的**正互反矩阵**称为**一致矩阵**。

对判断矩阵的一致性检验的步骤如下:

(i) 计算一致性指标 CI

$$CI = \frac{\lambda_{\max} - n}{n - 1}$$

(ii) 查找相应的平均随机一致性指标 RI 。对 $n = 1 \sim 9$, Saaty 给出了 RI 的值, 如表 2 所示

表 2 RI 的值

| n | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|------|---|---|------|------|------|------|------|------|------|
| RI | 0 | 0 | 0.58 | 0.90 | 1.12 | 1.24 | 1.32 | 1.41 | 1.45 |

RI 的值是这样得到的, 用随机方法构造 500 个样本矩阵: 随机地从 1~9 及其倒数中抽取数字构造正互反矩阵, 求得最大特征根的平均值 λ'_{\max} , 并定义

$$RI = \frac{\lambda'_{\max} - n}{n - 1}。$$

(iii) 计算一致性比例 CR

$$CR = \frac{CI}{RI}$$

当 $CR < 0.10$ 时, 认为判断矩阵的一致性是可以接受的, 否则应对判断矩阵作适当修正。

4) 层次总排序及一致性检验

上面我们得到的是一组元素对其上一层中某元素的权重向量。我们最终要得到各元素, 特别是最底层中各方案对于目标的排序权重, 从而进行方案选择。总排序权重要自上而下地将单准则下的权重进行合成。

表 3 层次总排序合成表

| 层 A 层 B | 层 A | | | | B 层总排序权值 |
|------------|----------|----------|---------|----------|--------------------------|
| | A_1 | A_2 | \dots | A_m | |
| | a_1 | a_2 | \dots | a_m | |
| B_1 | b_{11} | b_{12} | \dots | b_{1m} | $\sum_{j=1}^m b_{1j}a_j$ |
| B_2 | b_{21} | b_{22} | \dots | b_{2m} | $\sum_{j=1}^m b_{2j}a_j$ |
| \vdots | \dots | \dots | \dots | \dots | \vdots |
| B_n | b_{n1} | b_{n2} | \dots | b_{nm} | $\sum_{j=1}^m b_{nj}a_j$ |

对层次总排序也需作一致性检验, 检验仍像层次总排序那样由高层到低层逐层进行。这是因为虽然各层次均已经过层次单排序的一致性检验, 各成对比较判断矩阵都已具有较为满意的一致性。但当综合考察时, 各层次的非一致性仍有可能积累起来, 引起最终分析结果较严重的非一致性。

设 B 层中与 A_j 相关的因素的成对比较判断矩阵在单排序中经一致性检验, 求得单排序一致性指标为 $CI(j)$, ($j = 1, \dots, m$), 相应的平均随机一致性指标为 $RI(j)$ ($CI(j)$ 、 $RI(j)$ 已在层次单排序时求得), 则 B 层总排序随机一致性比例为

$$CR = \frac{\sum_{j=1}^m CI(j)a_j}{\sum_{j=1}^m RI(j)a_j}$$

当 $CR < 0.10$ 时, 认为层次总排序结果具有较满意的一致性并接受该分析结果。

例 51 (续 50) 试用层次分析法对总成绩进行排序。

解 排序的 Matlab 程序如下

```
a=randint(10,4,[60 95]);
dlmwrite('data.txt',a); %将数据存入文件
A=[1 3 5 7;1/3 1 2 4;1/5 1/2 1 2;1/7 1/5 1/2 1];%判断矩阵
[vec,val]=eig(A);%求特征向量和特征值
w=vec(:,1)/sum(vec(:,1)); %求得每个指标的权重
zb1=a*w %求指标的得分
[szb1,ind1]=sort(zb1,'descend') %降序排列
```

11.3 灰色关联分析法

对于两个系统之间的因素, 其随时间或不同对象而变化的关联性大小的量度, 称为关联度。在系统发展过程中, 若两个因素变化的趋势具有一致性, 即同步变化程度较高, 即可谓二者关联程度较高; 反之, 则较低。因此, 灰色关联分析方法, 是根据因素之间发展趋势的相似或相异程度, 亦即“灰色关联度”, 作为衡量因素间关联程度的一种方法。

灰色关联分析具体步骤如下:

(1) 确定比较对象 (评价对象) 和参考数列 (评价标准)

设评价对象有 m 个, 评价指标有 n 个, 参考数列为 $x_0 = \{x_0(k) | k = 1, 2, \dots, n\}$, 比较数列为 $x_i = \{x_i(k) | k = 1, 2, \dots, n\}$, $i = 1, 2, \dots, m$ 。

(2) 确定各指标值对应的权重

可用层次分析法等确定各指标对应的权重 $w = [w_1, \dots, w_n]$, 其中 w_k , $k = 1, 2, \dots, n$ 为第 k 个评价指标对应的权重。

(3) 计算灰色关联系数

$$\xi_i(k) = \frac{\min_s \min_t |x_0(t) - x_s(t)| + \rho \max_s \max_t |x_0(t) - x_s(t)|}{|x_0(k) - x_i(k)| + \rho \max_s \max_t |x_0(t) - x_s(t)|}$$

为比较数列 x_i 对参考数列 x_0 在第 k 个指标上的关联系数，其中 $\rho \in [0,1]$ 为分辨系数。称式中 $\min_s \min_t |x_0(t) - x_s(t)|$ 、 $\max_s \max_t |x_0(t) - x_s(t)|$ 分别为两级最小差及两级最大差。

一般来讲，分辨系数 ρ 越大，分辨率越大； ρ 越小，分辨率越小。

(4) 计算灰色加权关联度

灰色加权关联度的计算公式为

$$r_i = \sum_{k=1}^n w_i \xi_i(k),$$

这里 r_i 为第 i 个评价对象对理想对象的灰色加权关联度。

(5) 评价分析

根据灰色加权关联度的大小，对各评价对象进行排序，可建立评价对象的关联序，关联度越大其评价结果越好。

例 52（续 50）试用灰色关联分析法对总成绩进行排序。

解 排序的 Matlab 程序如下

```
clc, clear
a=load('data.txt');
ck=max(a); %求参考序列
cha= repmat(ck,10,1)-a;
mmax=max(max(cha)); %求最大差
mmin=min(min(cha)); %求最小差
rho=0.5; %分辨系数
xishu=(mmin+rho*mmax)./(cha+rho*mmax) %计算灰色关联系数
guanliandu=mean(xishu) %取等权重，计算关联度
[gsort,ind]=sort(guanliandu,'descend') %对关联度按照从大到小排序
```

11.4 主成分分析法

主成分分析的主要目的是希望用较少的变量去解释原来资料中的大部分变异，将我们手中许多相关性很高的变量转化成彼此相互独立或不相关的变量。通常是选出比原始变量个数少，能解释大部分资料中的变异的几个新变量，即所谓**主成分**，并用以解释资料的综合性指标。由此可见，主成分分析实际上是一种**降维方法**。

其主要步骤为：

(1) 原始数据标准化：

$$\hat{a}_{ij} = \frac{a_{ij} - \mu_j}{s_j}, \text{ 其中 } \mu_j = \frac{1}{n} \sum_{i=1}^n a_{ij}, s_j = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (a_{ij} - \mu_j)^2}, \quad j = 1, 2, \dots, m \text{ 表示指标数量}$$

μ_j 和 s_j 分别为第 j 个指标的样本均值和样本标准差。标准化的指标变量表示形式如下：

$$\hat{x}_j = \frac{x_j - \mu_j}{s_j}, \text{ 标准化过程可用函数 } \mathbf{zscore}() \text{ 实现。}$$

(2) 计算相关系数矩阵 $R = (r_{ij})_{m \times n}$ 其中

$$r_{ij} = \frac{\sum_{k=1}^n \hat{a}_{ki} \hat{a}_{kj}}{n-1} (i, j = 1, 2, \dots, m) \text{ 可知 } r_{ii} = 1, r_{ij} = r_{ji}$$

r_{ij} 为第 i 个指标和第 j 个指标的相关系数，可用 Matlab 函数 **corrcoef()** 计算。

(3) 计算特征值和特征向量

求相关系数矩阵 R 的特征值并且排序 $\lambda_1 > \lambda_2 > \dots > \lambda_m \geq 0$ 及对应的特征向量

$\mu_1, \mu_2, \dots, \mu_m$, 其中 $\mu_j = [\mu_{1j}, \mu_{2j}, \dots, \mu_{mj}]^T$, 由特征向量组成 m 个新的指标变量，写出主成分分量：

$$\begin{cases} y_1 = \mu_{11} \hat{x}_1 + \mu_{21} \hat{x}_2 + \dots + \mu_{m1} \hat{x}_m \\ \dots \\ y_m = \mu_{1m} \hat{x}_1 + \mu_{2m} \hat{x}_2 + \dots + \mu_{mm} \hat{x}_m \end{cases}$$

(4) 选择 p 个主成分 ($p \leq m$)，计算综合评价

1) 计算 λ_i ($i=1, 2, \dots, m$) 的**信息贡献率**和**累积贡献率**

$$b_j = \frac{\lambda_j}{\sum_{k=1}^m \lambda_k} (j = 1, 2, \dots, m) \text{ 为主成分 } y_j \text{ 的信息贡献率}$$

$$a_p = \frac{\sum_{k=1}^p \lambda_k}{\sum_{k=1}^m \lambda_k} \text{ 称为 } y_1, y_2, \dots, y_p \text{ 的累积贡献率，当 } a_p \text{ 接近于 } 1 \text{ (取 } 0.85, 0.9, 0.95 \text{ 等) 时，}$$

选择前 p 个指标变量 y_1, y_2, \dots, y_p 作为 p 个主成分，代替原来的 m 个主成分，从而可对 p 个主成分进行综合分析。

2) 计算综合得分

$$Z = \sum_{j=1}^p b_j y_j$$

例 53（续 50）试用主成分分析法对总成绩进行排序。

解 Matlab 程序为：

```
fs=load('data.txt'); % 载入数据
fs2=zscores(fs); % 进行标准化处理
r=corrcoef(fs2); % 计算相关系数矩阵
[vec,val,rate]=pcacov(r); % 求归一化后的特征值，特征向量和贡献率
fh= repmat(sign(sum(vec)),4,1); % 构造与 r 同维数的元素为+1 和-1 的矩阵，
% 进行符号变换，处理特征向量矩阵，使第一列元素大于 1
vec2=vec.*fh; % 修改特征向量的正负号，每个特征向量乘以所有分量和的符号函数值
xishu=vec2(:,1:3); % 只取前 3 列
df=fs2*xishu; % 计算各成分得分
zb4=df*rate(1:3); % 计算综合得分
[szb4,ind4]=sort(zb4,'descend'); % 把得分按照从高到低的次序排列
```

11.5 秩和比法（RSR 法）

秩和比统计方法是我国统计学家田凤调在 1988 年提出的，是一种将多项指标综合成一个具有 0~1 连续变量特征的统计量，也可看成 0~100 的计分。多用于现成卫生统计资料的再分析。不论所分析的问题是什么，计算的 RSR 越大越好。为此，在编秩时要区分高优指标和低优指标，有时还要引进不分高低的情况。例如，评价预期寿命、受检率、合格率等可视为高优指标；发病率、病死率、超标率为低优指标。在疗效评价中，不变率、微效率等可看作不分高低的指标。指标值相同时应编以平均秩次。

样本秩的概念：

设 x_1, x_2, \dots, x_n 是从一元总体抽取的容量为 n 的样本，其从小到大顺序统计量是 $x(1), x(2), \dots, x(n)$ ，若 $x_i = x(k)$ ，则称 k 是 x_i 在样本中的秩，记做 R_i ，对每一个 $i=1, 2, \dots, n$ ，称 R_i 是第 i 的秩统计量。 R_1, R_2, \dots, R_n 总称为秩统计量。

例如：样本数据 -0.8, -3.1, 1, -5.2, 4.3

顺序统计量为：-5.2 -3.1 -0.8 1, 4.2

则秩统计量为：3 2 4 1 5

设有 m 个指标，对 n 组数据进行评价，形成 n 行 m 列的数据阵，则各行：

$$RSR_i = \frac{\sum_{j=1}^m R_{ij}}{(m * n)}$$

其中 RSR 为分别按列编秩后各行的秩次。最小 $RSR=1/n$ ，最大 $RSR=1$ 。秩和比法基本步骤：

1)编秩

将 n 个评价对象的 m 个指标排列成 n 行 m 列的原始数据表，编出每个指标各评价对象的秩，其中效益型指标从小到大编秩，成本型指标从大到小编秩，同一指标数据相同者编平均秩，得到的秩矩阵为 $(R_{ij})_{n \times m}$

2)计算秩和比（RSR）

根据公式 $RSR_i = \frac{\sum_{j=1}^m R_{ij}}{(m * n)}$ 计算，当各评价指标的权重不同时，计算加权秩和

比（ $WRSR$ ），计算公式为： $WRSR_i = \frac{1}{n} \sum_{j=1}^m w_j R_{ij}, \sum_{j=1}^m w_j = 1$

3)计算概率单位。编制 RSR （ $WRSR$ ）频率分布表，列出各组频数 f_i ，计算各组累积频数 cf_i ，计算累积频率 $p_i = cf_i/n$ ，将 p_i 转化为概率单位 $Probit_i$ ， $Probit_i$ 为标准正态分布的 p_i 分位数加 5。

4) 计算直线回归方程。以累积频率所对应的概率单位 $Probit_i$ 为自变量，以 RSR 值为因变量，计算直接回归方程即 $RSR=a+b*Probit$

5)分档排序。按照回归方程推算所对应的 RSR 估计值对评价对象进行分档排序

例 53：某医院 1983-1992 年工作质量统计指标及权重系数如下表所示，其中 x_1 为治愈率， x_2 为病死率， x_3 为周转率， x_4 为平均病床工作日， x_5 为病床使用率， x_6 为平均住院日， x_2 和 x_6 为成本型指标，其余为效益型指标。

| 年度 | x_1 | x_2 | x_3 | x_4 | x_5 | x_6 |
|------|-------|-------|-------|-------|-------|-------|
| 1983 | 75.2 | 3.5 | 38.2 | 370.1 | 101.5 | 10 |

| | | | | | | |
|------|-------|-------|-------|-------|-------|-------|
| 1984 | 76.1 | 3.3 | 36.7 | 369.6 | 101 | 10.3 |
| 1985 | 80.4 | 2.7 | 30.5 | 309.7 | 84.8 | 10 |
| 1986 | 77.8 | 2.7 | 36.3 | 370.1 | 101.4 | 10.2 |
| 1987 | 75.9 | 2.3 | 38.9 | 369.4 | 101.2 | 9.61 |
| 1988 | 74.3 | 2.4 | 36.7 | 335.3 | 91.9 | 9.2 |
| 1989 | 74.6 | 2.2 | 37.5 | 356.2 | 97.6 | 9.3 |
| 1990 | 72.1 | 1.8 | 40.3 | 401.7 | 101.1 | 10 |
| 1991 | 72.8 | 1.9 | 37.1 | 372.8 | 102.1 | 10 |
| 1992 | 72.1 | 1.5 | 33.2 | 358.1 | 97.8 | 10.4 |
| 权重系数 | 0.093 | 0.418 | 0.132 | 0.1 | 0.098 | 0.159 |

解：编秩，加权秩和比计算结果如下表所示：

| 年度 | x1 | x2 | x3 | x4 | x5 | x6 | WRSR |
|------|-----|-----|-----|-----|----|-----|---------|
| 1983 | 6 | 1 | 7.5 | 7.5 | 9 | 5.5 | 0.4539 |
| 1984 | 8 | 2 | 6 | 6 | 5 | 2 | 0.3582 |
| 1985 | 10 | 3.5 | 1 | 1 | 1 | 5.5 | 0.3598 |
| 1986 | 9 | 3.5 | 7.5 | 7.5 | 8 | 3 | 0.4707 |
| 1987 | 7 | 6 | 5 | 5 | 7 | 8 | 0.6805 |
| 1988 | 4 | 5 | 2 | 2 | 2 | 10 | 0.5042 |
| 1989 | 5 | 7 | 3 | 3 | 3 | 9 | 0.634 |
| 1990 | 1.5 | 9 | 10 | 10 | 6 | 5.5 | 0.7684 |
| 1991 | 3 | 8 | 9 | 9 | 10 | 5.5 | 0.71695 |
| 1992 | 1.5 | 10 | 4 | 4 | 4 | 1 | 0.5535 |

计算各组的频数 f_i ，累积频数 cf_i ，累积频率 pi ，概率单位 $Probit_i$ ，最后一个累积频率 0.975 按照 $1-1/(4n)$ 估计。计算结果如下表所示：

| 年度 | f_i | cf_i | pi | $Probit_i$ | $WRSRfit_i$ | 排序 |
|------|-------|--------|-------|------------|-------------|----|
| 1983 | 1 | 1 | 0.1 | 3.7184 | 0.4093 | 10 |
| 1984 | 1 | 2 | 0.2 | 4.1584 | 0.4512 | 9 |
| 1985 | 1 | 3 | 0.3 | 4.4756 | 0.4814 | 8 |
| 1986 | 1 | 4 | 0.4 | 4.7467 | 0.5072 | 7 |
| 1987 | 1 | 5 | 0.5 | 5 | 0.5313 | 6 |
| 1988 | 1 | 6 | 0.6 | 5.2533 | 0.5555 | 5 |
| 1989 | 1 | 7 | 0.7 | 5.2544 | 0.5813 | 4 |
| 1990 | 1 | 8 | 0.8 | 5.8416 | 0.6115 | 3 |
| 1991 | 1 | 9 | 0.9 | 6.2816 | 0.6534 | 2 |
| 1992 | 1 | 10 | 0.975 | 6.95996 | 0.71797 | 1 |

求得的一元线性回归方程为 $WRSR=0.0552+0.0954Probit$ ，计算的估计值为 $WRSRfit_i$ ，排序结果为上表的最后一列。

Matlab 计算程序如下所示：

`aw=load('zhb.txt');` %把 x1,...,x6 的数据和权重数据保存在纯文本文件 zhb.txt 中


```

w=aw(end,:); %提取权重向量
a=aw([1:end-1],:); %提取指标数据
a(:,[2,6])=-a(:,[2,6]); %把成本型指标转换成效益型指标
ra=tiedrank(a) %对每个指标值分别编秩，即对 a 的每一列分别编秩
[n,m]=size(ra); %计算矩阵 sa 的维数
RSR=mean(ra,2)/n %计算秩和比
W=repmat(w,[n,1]);
WRSR=m*mean(ra.*W,2)/n %计算加权秩和比
p=[1:n]/n; %计算累积频率
p(end)=1-1/(4*n) %修正最后一个累积频率，最后一个累积频率按 1-1/(4*n)估计
Probit=norminv(p,0,1)+5 %计算标准正态分布的 p 分位数+5
X=[ones(n,1),Probit']; %构造一元线性回归分析的数据矩阵
[ab,abint,r,rint,stats]=regress(WRSR,X) %一元线性回归分析
WRSRfit=ab(1)+ab(2)*Probit %计算 WRNR 的估计值
[sWRSRfit,ind]=sort(WRSRfit,'descend') %对 WRNR 的估计值按照从大到小排序

```

12、预测方法

12.1 多项式回归分析

例 54 根据表 4 某猪场 25 头育肥猪 4 个胴体性状的数据资料, 试进行瘦肉量 y 对眼肌面积 (x_1)、腿肉量(x_2)、腰肉量(x_3)的多元回归分析。

表 4 某养猪场数据资料

| 序号 | 瘦肉量 $y(kg)$ | 眼肌面积 $x_1(cm^2)$ | 腿肉量 $x_2(kg)$ | 腰肉量 $x_3(kg)$ | 序号 | 瘦肉量 $y(kg)$ | 眼肌面积 $x_1(cm^2)$ | 腿肉量 $x_2(kg)$ | 腰肉量 $x_3(kg)$ |
|----|----------------|---------------------|------------------|------------------|----|----------------|---------------------|------------------|------------------|
| 1 | 15.02 | 23.73 | 5.49 | 1.21 | 14 | 15.94 | 23.52 | 5.18 | 1.98 |
| 2 | 12.62 | 22.34 | 4.32 | 1.35 | 15 | 14.33 | 21.86 | 4.86 | 1.59 |
| 3 | 14.86 | 28.84 | 5.04 | 1.92 | 16 | 15.11 | 28.95 | 5.18 | 1.37 |
| 4 | 13.98 | 27.67 | 4.72 | 1.49 | 17 | 13.81 | 24.53 | 4.88 | 1.39 |
| 5 | 15.91 | 20.83 | 5.35 | 1.56 | 18 | 15.58 | 27.65 | 5.02 | 1.66 |
| 6 | 12.47 | 22.27 | 4.27 | 1.50 | 19 | 15.85 | 27.29 | 5.55 | 1.70 |
| 7 | 15.80 | 27.57 | 5.25 | 1.85 | 20 | 15.28 | 29.07 | 5.26 | 1.82 |
| 8 | 14.32 | 28.01 | 4.62 | 1.51 | 21 | 16.40 | 32.47 | 5.18 | 1.75 |
| 9 | 13.76 | 24.79 | 4.42 | 1.46 | 22 | 15.02 | 29.65 | 5.08 | 1.70 |
| 10 | 15.18 | 28.96 | 5.30 | 1.66 | 23 | 15.73 | 22.11 | 4.90 | 1.81 |
| 11 | 14.20 | 25.77 | 4.87 | 1.64 | 24 | 14.75 | 22.43 | 4.65 | 1.82 |
| 12 | 17.07 | 23.17 | 5.80 | 1.90 | 25 | 14.35 | 20.04 | 5.08 | 1.53 |
| 13 | 15.40 | 28.57 | 5.22 | 1.66 | | | | | |

要求

(1) 求 y 关于 x_1, x_2, x_3 的线性回归方程

$$y = c_0 + c_1x_1 + c_2x_2 + c_3x_3,$$

计算 c_0, c_1, c_2, c_3 的估计值;

(2) 对上述回归模型和回归系数进行检验 (要写出相关的统计量)

(3) 试建立 y 关于 x_1, x_2, x_3 的二项式回归模型, 并根据适当统计量指标选择一个较好的模型。

解 (1) 记 y, x_1, x_2, x_3 的观测值分别为 $b_i, a_{i1}, a_{i2}, a_{i3}$, $i = 1, 2, \dots, 25$,

$$X = \begin{bmatrix} 1 & a_{11} & a_{12} & a_{13} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & a_{25,1} & \cdots & a_{25,3} \end{bmatrix}, \quad Y = \begin{bmatrix} b_1 \\ \vdots \\ b_{25} \end{bmatrix}$$

用最小二乘法求 c_0, c_1, c_2, c_3 的估计值, 即应选取估计值 \hat{c}_j , 使当 $c_j = \hat{c}_j$, $j = 0, 1, 2, 3$ 时, 误差平方和

$$Q = \sum_{i=1}^{25} \varepsilon_i^2 = \sum_{i=1}^{25} (b_i - \hat{b}_i)^2 = \sum_{i=1}^n (b_i - c_0 - c_1 a_{i1} - c_2 a_{i2} - c_3 a_{i3})^2$$

达到最小。为此，令：

$$\frac{\partial Q}{\partial c_j} = 0, \quad j = 0, 1, 2, 3,$$

得到正规方程组，求解正规方程组得 c_0, c_1, c_2, c_3 的估计值，

$$[\hat{c}_0, \hat{c}_1, \hat{c}_2, \hat{c}_3] = (X^T X)^{-1} X^T Y.$$

利用 Matlab 程序，求得：

$$\hat{c}_0 = 0.8539, \quad \hat{c}_1 = 0.0178, \quad \hat{c}_2 = 2.0782, \quad \hat{c}_3 = 1.9396.$$

(2) 因变量 y 与自变量 x_1, x_2, x_3 之间是否存在线性关系是需要检验的，显然，如果所有的 $|\hat{c}_j|$ ($j=1, 2, 3$) 都很小， y 与 x_1, x_2, x_3 的线性关系就不明显，所以可令原假设为

$$H_0: c_j = 0, \quad j = 1, 2, 3. \quad (\text{式 } 15.41)$$

记 $m = 3$ ， $n = 25$ ， $Q = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (b_i - \hat{b}_i)^2$ ， $U = \sum_{i=1}^n (\hat{b}_i - \bar{b})^2$ ，这里

$$\hat{b}_i = \hat{c}_0 + \hat{c}_1 a_{i1} + \cdots + \hat{c}_m a_{im} \quad (i = 1, \cdots, n), \quad \bar{b} = \frac{1}{n} \sum_{i=1}^n b_i. \quad \text{当 } H_0 \text{ 成立时统计量}$$

$$F = \frac{U/m}{Q/(n-m-1)} \sim F(m, n-m-1), \text{ 在显著性水平 } \alpha \text{ 下, 若}$$

$$F_{1-\alpha/2}(m, n-m-1) < F < F_{\alpha/2}(m, n-m-1),$$

当 (15.41) 式的 H_0 被拒绝时， β_j 不全为零，但是不排除其中若干个等于零。所以应进一步作如下 $m+1$ 个检验

$$H_0^{(j)}: c_j = 0, \quad j = 0, 1, \cdots, m, \quad (9)$$

$$\text{当 } H_0^{(j)} \text{ 成立时 } t_j = \frac{\hat{\beta}_j / \sqrt{c_{jj}}}{\sqrt{Q/(n-m-1)}} \sim t(n-m-1),$$

这里 c_{jj} 是 $(X^T X)^{-1}$ 中的第 (j, j) 元素，对给定的 α ，若 $|t_j| < t_{\frac{\alpha}{2}}(n-m-1)$ ，接受 $H_0^{(j)}$ ；

否则拒绝。

利用 Matlab 程序，求得统计量

$$t_0 = 0.6223, \quad t_1 = 0.6090, \quad t_2 = 7.7407, \quad t_3 = 3.8062,$$

查表得上 $\alpha/2$ 分位数 $t_{0.025}(21) = 2.0796$ 。

对于 (9) 式的检验, 在显著性水平 $\alpha = 0.05$ 时, 接受 $H_0^{(j)}: c_j = 0$ ($j = 0, 1$), 拒绝 $H_0^{(j)}: c_j = 0$ ($j = 2, 3$), 即变量 x_1 对模型的影响是不显著的。建立线性模型时, 可以不使用 x_1 。

把全部原始数据, 包括 13 行后面的空行, 复制保存到纯文本文件 zhu.txt 中。

问题 (1) 和 (2) 的 Matlab 程序如下:

```
clc, clear
ab=textread('zhu.txt');
y=ab(:,[2,7]); %提取因变量 y 的观测值
Y=nonzeros(y) %去掉 y 后面的 0, 并变成列向量
x123=[ab([1:13],[3:5]); ab([1:12],[8:10])]; %提取 x1,x2,x3 的观测值
X=[ones(25,1),x123]; %构造多元线性回归分析的数据矩阵 X
[beta,betaint,r,rint,st]=regress(Y,X) %计算回归系数和统计量等, st 的第 2 个分量就是 F 统计量, 下面根据统计量的表达式重新计算的结果和这里是一样的。
q=sum(r.^2) %计算残差平方和
ybar=mean(Y) %计算 y 的观测值的平均值
yhat=X*beta; %计算 y 的估计值
u=sum((yhat-ybar).^2) %计算回归平方和
m=3; %变量的个数, 拟合参数的个数为 m+1
n=length(Y); %样本点的个数
F=u/m/(q/(n-m-1)) %计算 F 统计量的值, 自由度为样本点的个数减拟合参数的个数
fw1=finv(0.025,m,n-m-1) %计算上 alpha/2 分位数
fw2=finv(0.975,m,n-m-1) %计算上 1-alpha/2 分位数
c=diag(inv(X'*X)) %计算 c(j, j) 的值
t=beta./sqrt(c)/sqrt(q/(n-m-1)) %计算 t 统计量的值
tfw=finv(0.975,n-m-1) %计算 t 分布的上 alpha/2 分位数
save xydata Y x123 %把 Y 和 x123 保存到 mat 文件 xydata 中, 供问题 (3) 的二次模型使用
```

注: I) 在 **regress** 函数的第 5 个返回值中, 就包含 F 统计量的值, 不需单独计算。

II) **regress** 的返回值中不包括 t 统计量的值, 如果需要则要单独计算。由于假设检验和参数的区间估计是等价的, **regress** 的第 2 个返回值是**各参数的区间估计**, 如果某参数的区间估计包含 0 点, 则该参数对应的变量是不显著的。

III) 拟合时常用到的几个参数说明:

a) **SSE** 误差平方和
$$SSE = \sum_{i=1}^n w_i (y_i - \hat{y}_i)^2$$

b) **MSE** 均方差 $MSE = SSE / n = \frac{1}{n} \sum_{i=1}^n w_i (y_i - \hat{y}_i)^2$

c) **RMSE** 均方根，也叫回归系统的拟合标准差

$$RMSE = \sqrt{MSE} = \sqrt{SSE / n} = \sqrt{\frac{1}{n} \sum_{i=1}^n w_i (y_i - \hat{y}_i)^2}$$

d) **R-square** 确定系数（复相关系数），由 **SSR** 和 **SST** 确定，其中 SSR 是预测数据

与原始数据均值之差的平方和， $SSR = \sum_{i=1}^n w_i (\hat{y}_i - \bar{y}_i)^2$ ，SST 是原始数据和均值之差的平方和， $SST = \sum_{i=1}^n w_i (y_i - \bar{y}_i)^2$ ， $SST = SSE + SSR$ ， $R\text{-square} = \frac{SSR}{SST} = \frac{SST - SSE}{SST} = 1 - \frac{SSE}{SST}$

“确定系数”是通过数据的变化来表征一个拟合的好坏。由上面的表达式可以知道“确定系数”的正常取值范围为[0 1]，越接近 1，表明方程的变量对 y 的解释能力越强，这个模型对数据拟合的也较好

（3）我们使用 Matlab 的用户图形界面解法求二项式回归模型。根据剩余标准差（rmse）这个指标选取较好的模型是交叉二次模型，模型为

$$y = -17.0988 + 0.3611x_1 + 2.3563x_2 + 18.2730x_3 - 0.1412x_1x_2 - 0.4404x_1x_3 - 1.2754x_2x_3 + 0.0217x_1^2 + 0.5025x_2^2 + 0.3962x_3^2.$$

计算的 Matlab 程序如下

```
clc, clear
load xydata
rstool(x123,Y)
```

示例：对某地区生产同一产品的 8 个不同规模的企业进行生产费用调查，得到产量和生产费用的数据，对其进行回归分析。

```
X=[1 1 1 1 1 1 1 1;1.5 2 3 4.5 7.5 9.1,10.5,12];
Y=[5.6 6.6 7.2 7.8 10.1 10.8 13.5 16.5];
[b,bint,r,rint,stats]=regress(Y',X',0.05)
rcoplot(r,rint)
X=X';Y=Y';
scatter3(X(:,1),X(:,2),Y,'filled');
hold on
x2fit=min(X(:,2)):0.5:max(X(:,2));
x1fit=ones(1,size(x2fit,2));
[x1,x2]=meshgrid(x1fit,x2fit);
Yfit=b(1)*x1+b(2)*x2;
mesh(x1,x2,Yfit);
```

示例 2: 求 $y=a+b*x_1+c*x_2+d*x_2^2$ 中的系数

```

y=[196 63 252 84 126 14 49 49 266 49 105 98 77 14 56 245 133 133]';
x1=[66.290 40.964 72.996 45.010 57.204 26.852 38.122 35.840 75.796 37.408 54.376
46.186 46.130 30.366 39.060 79.380 52.766 55.916]';
x2=[7 5 10 6 4 5 4 6 9 5 2 7 4 3 5 1 8 6]';
stand=ones(18,1);
x=[stand,x1,x2,x2.^2];
[b,bint,r,rint,stats]=regress(y,x)

```

12.2 非线性回归

非线性回归是指因变量 y 对回归系数 $\beta_1, \beta_2, \dots, \beta_m$ （而不是自变量）是非线性的。

Matlab 统计工具箱中的命令 **nlfit**, **nlparci**, **nlpredci**, **nlintool**, 不仅给出拟合的回归系数及其置信区间, 而且可以给出预测值及其置信区间等。下面通过例题说明这些命令的用法。

例 54: 在研究化学动力学反应过程中, 建立了一个反应速度和反应物含量的数学模型, 形式为

$$y = \frac{\beta_4 x_2 - \frac{x_3}{\beta_5}}{1 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3},$$

其中 β_1, \dots, β_5 是未知的参数, x_1, x_2, x_3 是三种反应物（氢, n 戊烷, 异构戊烷）的含量, y 是反应速度。今测得一组数据如表4, 试由此确定参数 β_1, \dots, β_5 , 并给出其置信区间。 β_1, \dots, β_5 的参考值为 $[0.1, 0.05, 0.02, 1, 2]$ 。

表4 反应数据

| 序号 | 反应速度 y | 氢 x_1 | n 戊烷 x_2 | 异构戊烷 x_3 |
|----|----------|---------|--------------|------------|
| 1 | 8.55 | 470 | 300 | 10 |
| 2 | 3.79 | 285 | 80 | 10 |
| 3 | 4.82 | 470 | 300 | 120 |
| 4 | 0.02 | 470 | 80 | 120 |
| 5 | 2.75 | 470 | 80 | 10 |
| 6 | 14.39 | 100 | 190 | 10 |
| 7 | 2.54 | 100 | 80 | 65 |
| 8 | 4.35 | 470 | 190 | 65 |
| 9 | 13.00 | 100 | 300 | 54 |
| 10 | 8.50 | 100 | 300 | 120 |
| 11 | 0.05 | 100 | 80 | 120 |
| 12 | 11.32 | 285 | 300 | 10 |
| 13 | 3.13 | 285 | 190 | 120 |

解 首先，以回归系数和自变量为输入变量，将要拟合的模型写成匿名函数。然后，用 **nlinf** 计算回归系数，用 **nlparci** 计算回归系数的置信区间，用 **nlpredci** 计算预测值及其置信区间，编程如下 Matlab 程序

```

clc, clear
xy0=[8.55 470 300 10
3.79 285 80 10
4.82 470 300 120
0.02 470 80 120
2.75 470 80 10
14.39 100 190 10
2.54 100 80 65
4.35 470 190 65
13.00 100 300 54
8.50 100 300 120
0.05 100 80 120
11.32 285 300 10
3.13 285 190 120];
x=xy0(:,[2:4]);
y=xy0(:,1);
huaxue=@(beta,x)
(beta(4)*x(:,2)-x(:,3)/beta(5))./(1+beta(1)*x(:,1)+beta(2)*x(:,2)+beta(3)*x(:,3)); %用匿名函数定义要拟合的函数
beta0=[0.1,0.05,0.02,1,2]'; %回归系数的初值,可以任意取,这里是给定的
[beta,r,j]=nlinf(x,y,huaxue,beta0) %计算回归系数 beta; r,j 是下面命令用的信息
betaci=nlparci(beta,r,'jacobian',j) %计算回归系数的置信区间
[yhat,delta]=nlpredci(huaxue,x,beta,r,'jacobian',j) %计算 y 的预测值及置信区间半径,
用 nlintool 得到一个交互式画面, 左下方的 Export 可向工作空间传送数据, 如剩余

```

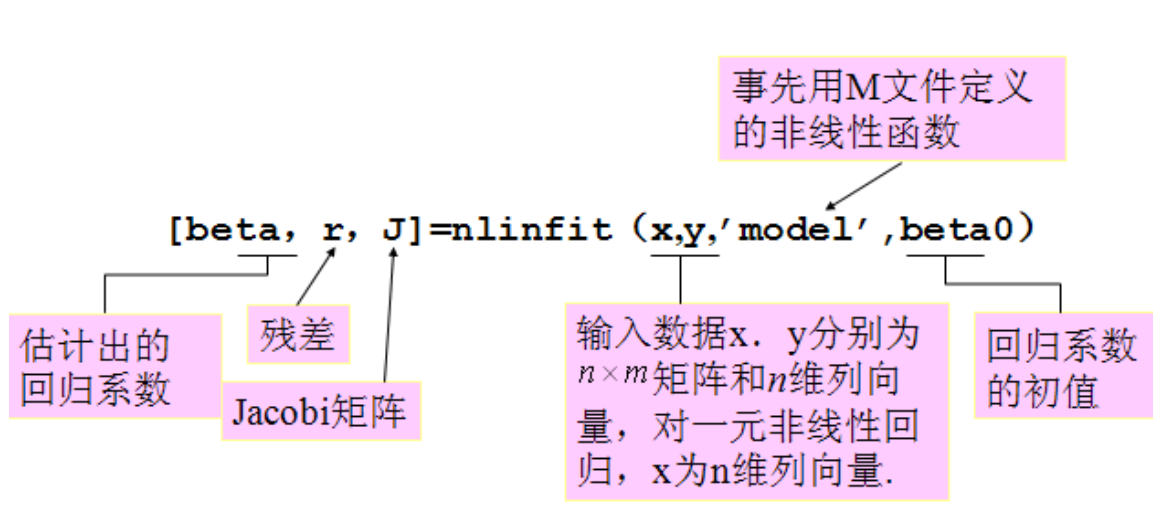
标准差等。使用命令

```
nlintool(x,y,huaxue,beta0)
```

(注意这里 `huaxue, beta0` 必须在工作空间中, 也就是说要把上面的程序运行一遍, 再运行 `nlintool`) 可看到画面, 并向工作空间传送有关数据, 例如剩余标准差 `rmse=0.1933`。

nlinfitt 的用法:

`[beta, r, J]=nlinfitt (x,y,'model',beta0)` `r` 返回残差, `J` 返回用于估计预测误差的 Jacobi 矩阵



使用函数 `[Y, DELTA]=nlpredci ('model',x, beta, r, J)`

求 `nlinfitt` 或 `lintool` 所得的回归函数在 `x` 处的预测值 `Y` 及预测值的显著性水平为 `1-alpha` 的置信区间 `Y+(-)DELTA`.

示例: 给出一个反应速率和反应物含量的数学模型, 形式为:

$$y = \frac{\beta_1 x_2 - \frac{x_3}{\beta_5}}{1 + \beta_2 x_1 + \beta_3 x_2 + \beta_4 x_3}$$

β_i : 未知参数, x_i 是3种反应物的含量, y 是反应速率, 测得一组数据 y 和 x_i , 由此确定 β_i , 给出参考值为(1,0.05,0.02,0.1,2)。

解:

首先编写 M 文件 `fun1.m`

```
function y=fun1(beta,x)
```

```
b1=beta(1);b2=beta(2);b3=beta(3);b4=beta(4);b5=beta(5);
```

```
x1=x(:,1);x2=x(:,2);x3=x(:,3);
```

```
y=(b1*x2-x3/b5)./(1+b2*x1+b3*x2+b4*x3);
```


end

然后编写调用文件:

```
clear all;clc;
x=[470 285 470 470 470 100 100 470 100 100 100 285 285;
   300 80 300 80 80 190 80 190 300 300 80 300 190;
   10 10 120 120 10 10 65 65 54 10 120 10 120]';
y=[8.55 3.79 4.82 0.02 2.75 14.39 2.54 4.35 13.00 8.5 0.05 11.32 3.13]';
beta0=[1 0.05 0.02 0.1 2]';
[beta,R,J]=nlinfit(x,y,@fun1,beta0);
beta
betacu=nlparci(beta,R,J)
[ypre,delta]=nlpredci(@fun1,x,beta,R,J);
plot(x(:,1),y,'o',x(:,1),ypre,'*');
```

12.3 GM(1,1)预测模型

灰色预测的主要特点是模型不使用原始数据序列,而是生成的数据序列,对原始数据做累加生成得到近似的指数规律在进行建模的方法。优点是不需要很多数据,一般只要 4 个数据,就可以解决历史数据少、序列的完整性及可靠性低的问题。缺点是只适用于中短期和指数增长的预测。

GM(1,1)表示模型是 1 阶微分方程,且只含 1 个变量的灰色模型。

定义 1 已知参考数据列 $x^{(0)} = (x^{(0)}(1), x^{(0)}(2), \dots, x^{(0)}(n))$, 1 次累加生成序列 (1—AGO)

$$\begin{aligned}x^{(1)} &= (x^{(1)}(1), x^{(1)}(2), \dots, x^{(1)}(n)) \\ &= (x^{(0)}(1), x^{(0)}(1) + x^{(0)}(2), \dots, x^{(0)}(1) + \dots + x^{(0)}(n)),\end{aligned}$$

其中 $x^{(1)}(k) = \sum_{i=1}^k x^{(0)}(i)$ ($k = 1, 2, \dots, n$)。 $x^{(1)}$ 的均值生成序列

$$z^{(1)} = (z^{(1)}(2), z^{(1)}(3), \dots, z^{(1)}(n)),$$

其中 $z^{(1)}(k) = 0.5x^{(1)}(k) + 0.5x^{(1)}(k-1)$, $k = 2, 3, \dots, n$ 。

建立灰微分方程

$$x^{(0)}(k) + az^{(1)}(k) = b, \quad k = 2, 3, \dots, n,$$

相应的白化微分方程为

$$\frac{dx^{(1)}}{dt} + ax^{(1)}(t) = b. \quad (10)$$

$$\text{记 } u = [a, b]^T, \quad Y = [x^{(0)}(2), x^{(0)}(3), \dots, x^{(0)}(n)]^T, \quad B = \begin{bmatrix} -z^{(1)}(2) & 1 \\ -z^{(1)}(3) & 1 \\ \vdots & \vdots \\ -z^{(1)}(n) & 1 \end{bmatrix}, \quad \text{则由最}$$

小二乘法, 求得使 $J(u) = (Y - Bu)^T(Y - Bu)$ 达到最小值的 u 的估计值

$$\hat{u} = [\hat{a}, \hat{b}]^T = (B^T B)^{-1} B^T Y. \text{ 于是求解方程 (10) 得}$$

$$\hat{x}^{(1)}(k+1) = \left(x^{(0)}(1) - \frac{\hat{b}}{\hat{a}} \right) e^{-\hat{a}k} + \frac{\hat{b}}{\hat{a}}, \quad k = 0, 1, \dots, n-1, \dots.$$

GM(1,1)模型预测步骤

1. 数据的检验与处理

首先, 为了保证建模方法的可行性, 需要对已知数据列作必要的检验处理。设参考数据为 $x^{(0)} = (x^{(0)}(1), x^{(0)}(2), \dots, x^{(0)}(n))$, 计算序列的级比

$$\lambda(k) = \frac{x^{(0)}(k-1)}{x^{(0)}(k)}, \quad k = 2, 3, \dots, n.$$

如果所有的级比 $\lambda(k)$ 都落在可容覆盖 $\Theta = (e^{-\frac{2}{n+1}}, e^{\frac{2}{n+2}})$ 内, 则序列 $x^{(0)}$ 可以作为模型 GM(1,1) 的数据进行灰色预测。否则, 需要对序列 $x^{(0)}$ 做必要的变换处理, 使其落入可容覆盖内。即取适当的常数 c , 作平移变换

$$y^{(0)}(k) = x^{(0)}(k) + c, \quad k = 1, 2, \dots, n,$$

使序列 $y^{(0)} = (y^{(0)}(1), y^{(0)}(2), \dots, y^{(0)}(n))$ 的级比

$$\lambda_y(k) = \frac{y^{(0)}(k-1)}{y^{(0)}(k)} \in \Theta, \quad k = 2, 3, \dots, n.$$

2. 建立模型

按 (10) 式建立 GM(1,1) 模型, 则可以得到预测值

$$\hat{x}^{(1)}(k+1) = \left(x^{(0)}(1) - \frac{\hat{b}}{\hat{a}} \right) e^{-\hat{a}k} + \frac{\hat{b}}{\hat{a}}, \quad k = 0, 1, \dots, n-1, \dots,$$

而且 $\hat{x}^{(0)}(k+1) = \hat{x}^{(1)}(k+1) - \hat{x}^{(1)}(k), \quad k = 1, 2, \dots, n-1, \dots$ 。

3. 检验预测值

(1) 残差检验

令残差为 $\varepsilon(k)$, 计算

$$\varepsilon(k) = \frac{x^{(0)}(k) - \hat{x}^{(0)}(k)}{x^{(0)}(k)}, \quad k = 1, 2, \dots, n,$$

这里 $\hat{x}^{(0)}(1) = x^{(0)}(1)$ ，如果 $\varepsilon(k) < 0.2$ ，则可认为达到一般要求；如果 $\varepsilon(k) < 0.1$ ，则认为达到较高的要求。

(2) 级比偏差值检验

首先由参考数据 $x^{(0)}(k-1)$ ， $x^{(0)}(k)$ 计算出级比 $\lambda(k)$ ，再用发展系数 a 求出相应的级比偏差：

$$\rho(k) = 1 - \left(\frac{1 - 0.5a}{1 + 0.5a} \right) \lambda(k),$$

如果 $\rho(k) < 0.2$ ，则可认为达到一般要求；如果 $\rho(k) < 0.1$ ，则认为达到较高的要求。

4. 预测预报

由 GM(1,1) 模型得到指定时区内的预测值，根据实际问题的需要，给出相应的预测预报。

例 55：北方某城市 1986~1992 年道路交通噪声平均声级数据见表 5。

表 5 城市交通噪声数据[dB(A)]

| 序号 | 年份 | L_{eq} | 序号 | 年份 | L_{eq} |
|----|------|----------|----|------|----------|
| 1 | 1986 | 71.1 | 5 | 1990 | 71.4 |
| 2 | 1987 | 72.4 | 6 | 1991 | 72.0 |
| 3 | 1988 | 72.4 | 7 | 1992 | 71.6 |
| 4 | 1989 | 72.1 | | | |

1. 级比检验

建立交通噪声平均声级数据时间序列如下

$$x^{(0)} = (x^{(0)}(1), x^{(0)}(2), \dots, x^{(0)}(7)) = (71.1, 72.4, 72.4, 72.1, 71.4, 72.0, 71.6).$$

(1) 求级比 $\lambda(k)$

$$\lambda(k) = \frac{x^{(0)}(k-1)}{x^{(0)}(k)},$$

$$\lambda = (\lambda(2), \lambda(3), \dots, \lambda(7)) = (0.982, 1, 1.0042, 1.0098, 0.9917, 1.0056).$$

(2) 级比判断

由于所有的 $\lambda(k) \in [0.982, 1.0098]$ ， $k = 2, \dots, 7$ ，故可以用 $x^{(0)}$ 作满意的 GM(1,1) 建模。

2. GM(1,1)建模

(1) 对原始数据 $x^{(0)}$ 作一次累加, 得到

$$x^{(1)} = (71.1, 143.5, 215.9, 288, 359.4, 431.4, 503).$$

(2) 构造数据矩阵 B 及数据向量 Y

$$B = \begin{bmatrix} -\frac{1}{2}(x^{(1)}(1) + x^{(1)}(2)) & 1 \\ -\frac{1}{2}(x^{(1)}(2) + x^{(1)}(3)) & 1 \\ \vdots & \vdots \\ -\frac{1}{2}(x^{(1)}(6) + x^{(1)}(7)) & 1 \end{bmatrix}, \quad Y = \begin{bmatrix} x^{(0)}(2) \\ x^{(0)}(3) \\ \vdots \\ x^{(0)}(7) \end{bmatrix}.$$

(3) 计算

$$\hat{u} = \begin{bmatrix} \hat{a} \\ \hat{b} \end{bmatrix} = (B^T B)^{-1} B^T Y = \begin{bmatrix} 0.0023 \\ 72.6573 \end{bmatrix}, \text{ 于是得到 } \hat{a} = 0.0023, \quad \hat{b} = 72.6573.$$

(4) 建立模型

$$\frac{dx^{(1)}}{dt} + \hat{a}x^{(1)} = \hat{b}, \text{ 求解得}$$

$$\hat{x}^{(1)}(k+1) = \left(x^{(0)}(1) - \frac{\hat{b}}{\hat{a}} \right) e^{-\hat{a}k} + \frac{\hat{b}}{\hat{a}} = -30929e^{-0.0023k} + 31000. \quad (11)$$

(5) 求生成序列预测值 $\hat{x}^{(1)}(k+1)$ 及模型还原值 $\hat{x}^{(0)}(k+1)$, 令 $k=1,2,3,4,5,6$, 由 () 式的时间响应函数可算得 $\hat{x}^{(1)}$, 其中取 $\hat{x}^{(1)}(1) = \hat{x}^{(0)}(1) = x^{(0)}(1) = 71.1$, 由 $\hat{x}^{(0)}(k+1) = \hat{x}^{(1)}(k+1) - \hat{x}^{(1)}(k)$, 取 $k=1,2,3,4,5,6$, 得

$$\hat{x}^{(0)} = (\hat{x}^{(0)}(1), \hat{x}^{(0)}(2), \dots, \hat{x}^{(0)}(7)) = (71.1, 72.4, 72.2, 72.1, 71.9, 71.7, 71.6).$$

3. 模型检验

模型的各种检验指标值的计算结果见表 6。

表 6 GM(1, 1)模型检验表

| 序号 | 年份 | 原始值 | 预测值 | 残差 | 相对误差 | 级比偏差 |
|----|------|------|---------|---------|-------|---------|
| 1 | 1986 | 71.1 | 71.1 | 0 | 0 | |
| 2 | 1987 | 72.4 | 72.4057 | -0.0057 | 0.01% | 0.0023 |
| 3 | 1988 | 72.4 | 72.2362 | 0.1638 | 0.23% | 0.0203 |
| 4 | 1989 | 72.1 | 72.0671 | 0.0329 | 0.05% | -0.0018 |
| 5 | 1990 | 71.4 | 71.8984 | -0.4984 | 0.7% | -0.0074 |
| 6 | 1991 | 72.0 | 71.7301 | 0.2699 | 0.37% | 0.0107 |
| 7 | 1992 | 71.6 | 71.5622 | 0.0378 | 0.05% | -0.0032 |

经验证，该模型的精度较高，可进行预测和预报。

计算的 Matlab 程序如下

```
clc,clear
x0=[71.1 72.4 72.4 72.1 71.4 72.0 71.6]'; %注意这里为列向量
n=length(x0);
lamda=x0(1:n-1)./x0(2:n) %计算级比
range=minmax(lamda') %计算级比的范围
x1=cumsum(x0); %累加运算
B=[-0.5*(x1(1:n-1)+x1(2:n)),ones(n-1,1)];
Y=x0(2:n);
u=B\Y %拟合参数 u(1)=a,u(2)=b
x=dsolve('Dx+a*x=b','x(0)=x0'); %求微分方程的符号解
x=subs(x,{'a','b','x0'},{u(1),u(2),x0(1)}); %代入估计参数值和初始值
yuce1=subs(x,'t',[0:n-1]); %求已知数据的预测值
y=vpa(x,6) %其中的6表示显示6位数字
yuce=[x0(1),diff(yuce1)] %差分运算，还原数据
epsilon=x0'-yuce %计算残差
delta=abs(epsilon./x0') %计算相对误差
rho=1-(1-0.5*u(1))/(1+0.5*u(1))*lamda' %计算级比偏差值，u(1)=a
```

说明：GM(1,1)模型适用于具有较强指数规律的序列，只能描述单调的变化过程，对于非单调的摆动发展序列或有饱和的S型序列，可以考虑建立 **GM(2,1)**、**DGM(2,1)** 或者灰色 **Verhulst** 模型求解。

12.4 预测方法及其适用的范围小结

1.插值与拟合方法：小样本内部预测；

应用案例：

- (1) CUMCM2001-A:血管的三维重建问题；
- (2) CUMCM2003-A,C:SARS 的传播问题；
- (3) CUMCM2004-C:饮酒驾车问题；
- (4) CUMCM2005-A:长江水质的评价与预测；
- (5) CUMCM2006-B:艾滋病疗法的评价与预测。

2.回归模型方法：大样本的内部预测；

应用案例：

- (1) CUMCM2004-A:奥运临时超市网点设计；
- (2) CUMCM2004-B:电力市场的输电阻塞管理；

(3) CUMCM2005-A:长江水质的评价与预测;

(4) CUMCM2006-B:艾滋病疗法的评价与预测;

(5) CUMCM2008-B:高教学费标准探讨问题。

3.灰度预测 $GM(1,n)$: 小样本的未来中短期预测;

应用案例:

(1) CUMCM2003-A:SARS 的传播问题;

(2) CUMCM2005-A:长江水质的评价与预测;

(3) CUMCM2006-B:艾滋病疗法的评价与预测;

(4) CUMCM2008-B:高教学费标准探讨问题;

(5) CUMCM2010-B:上海世博会影响力的评价。

4.时间序列方法: 大样本的随机因素或周期特征的 未来预测;

应用案例:

(1) CUMCM2003-A:SARS 的传播问题;

(2) CUMCM2005-A:长江水质的评价与预测;

(3) CUMCM2006-B:艾滋病疗法的评价与预测。

(4) CUMCM2010-B:上海世博会影响力的评价。

5.神经网络方法: 大样本的未来预测。

13、智能算法

13.1 人工神经网络

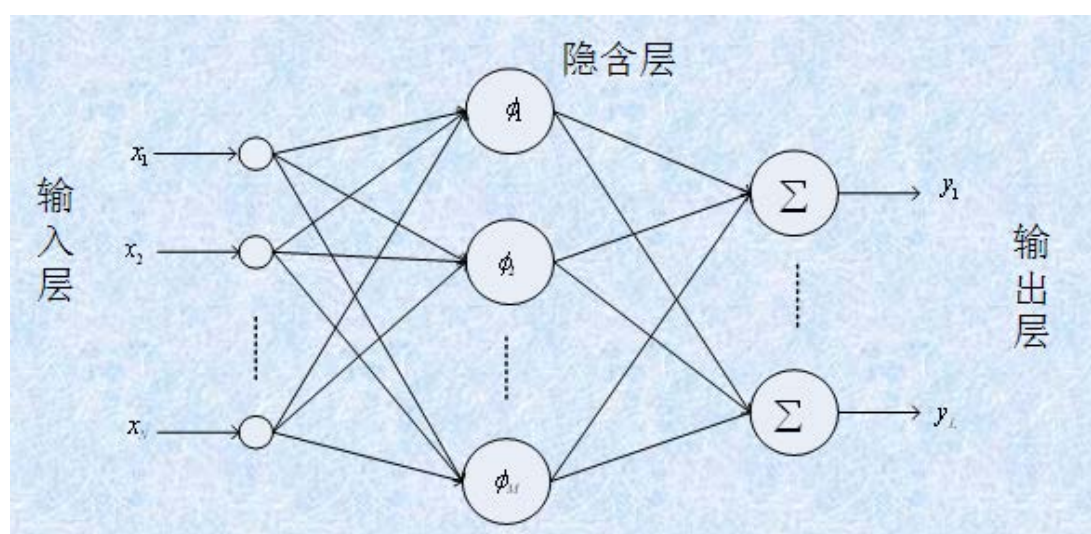
1、人工神经网络的原理

假如我们只知道一些输入和相应的输出，但是不清楚这些输入和输出之间的具体关系是什么，我们可以把输入和输出之间的未知过程看成是一个“网络”，通过不断的网络输入和相应的输出进行“训练”（学习），网络根据输入和对应输出不断调整连接网络的权值，直到满足我们的目标要求，这样就训练好了一个神经网络，当我们给定一个输入，网络就会计算出一个相应的输出。

2、网络结构

神经网络一般有一个输入层，多个隐层，和一个输出层。隐层并非越多越好。

如下图所示：



13.2 Matlab 神经网络工具箱

MATLAB 神经网络工具箱几乎涵盖了所有的神经网络的基本常用模型，如感知器、BP 网络和 RBFNN 等。它由 **nftool, nctool, nprtool, nntraintool** 和 **nnntool** 组成。主要应用于函数逼近和数据拟合、信息处理和预测、神经网络控制和故障诊断等领域。

在实际应用中，针对具体的问题，首先需要分析利用神经网络来解决问题的性质，然后依据问题的特点，提取训练和测试数据样本，确定网络模型，最后通过对网络进行训练、仿真等检验网络的性能是否满足要求。具体过程如下：

(1)确定信息表达的方式，主要包括数据样本已知；数据样本之间相互关系不明确；输入/输出模式为连续的或离散的；数据样本的预处理；将数据样本分成训练样本和测试样本。

(2)网络模型的确定。 确定选择何种神经网络以及网络层数。

(3)网络参数的选择，如输入输出神经元个数的确定，隐层神经元的个数等

(4)训练模式的确定，包括选择合理的训练算法、确定合适的训练步数、指定适当的训练目标误差等

(5)网络测试，选择合理的样本对网络进行测试。

简单来讲就是三个步骤：建立网络(newXX)—训练网络(trainXX)—仿真网络(sim)

13.3 BP 神经网络的 Matlab 相关函数

BP 算法的基本思想：学习过程由信号的正向传播与误差的反向传播两个过程组成。正向传播时，输入样本从输入层传入，经各隐层逐层处理后，传向输出层。若输出层的实际输出与期望输出（教师信号）不符，则转入误差的反向传播阶段。误差反传是将输出误差以某种形式通过隐层向输入层逐层反传，并将误差分摊给各层的所有单元，从而获得各层单元的误差信号，此误差信号作为修正各单元权值的依据。权值不断调整的过程就是神经网络的学习训练过程。

BP 神经网络的设计内容：

(1) 网络层数的确定。模式样本较少时，选用较少的隐层节点，一般采用两层 BP 网络；当模式样本较多时，减少网络规模，可以增加一个隐层。

(2) 输入层节点数的确定。输入层起到缓冲存储器的作用，其节点个数取决于输入矢量的维数。

(3) 输出层节点数的确定。取决于两个方面，输出数据类型和表示该类型所需的数据大小。当 BP 网络用语模式分类时，以二进制形式来表示不同模式的输出结果，则输出层的节点数可根据分类模式数来确定。若待分类模式的总数为 m ，则有两种方法确定输出层的节点数：

1) 节点数即待分类模式总数 m ，输出为对应的第 j 个分量为 1，其余为 0；

2) 节点数取经验值 $\log_2 m$ ，对应 m 种输出模式的二进制编码。

(4) 隐层节点数的确定。一般通过反复测试获取较好的节点数。对于模式识别/分

类的 BP 网络，根据经验公式 $n=\sqrt{n_1+n_0}+a$ ， n_1 为输入节点数， n_0 为输出节点数， a 取 1~10 之间的常数。

(5) 传输函数。一般采用 S 型函数 $f(x)=\frac{1}{1+e^{-x}}$

(6) 训练方法及其参数选择。Matlab 工具箱提供了多种训练函数可供选择。

Matlab 工具箱中与 BP 神经网络相关的函数：

(1) 创建一个 BP 网络的函数 newff,调用形式为：

net=newff();在对话框中创建一个 BP 网络

net=newff(PR,[S1 S2 ... SN],[TF1 TF2...TFN],BTF,BLF,PF);

PR:由每组输入(共 R 组)元素的最大值和最小值组成的 $R \times 2$ 维矩阵; Si: 第 i 层的长度, 共有 N 层;TFi:第 i 层的传递函数, 默认为 tansig; BTF:BP 网络的训练函数, 默认为 trainlm; BLF:权值和阈值的学习算法, 默认为 learnngdm;PF 网络的性能函数, 默认为 mse.

新版本的创建函数为 feedforwardnet(), 调用格式为：

net=feedforwardnet(h,f)

其中 h 为行向量, 表示各隐层节点个数, f 为训练函数, 默认为'trainlm'。

(2) 传递函数 有 logsig,dlogsig(导函数),tansig(双曲正切),dtansgi,purelin,dpurelin 等

(3) 学习函数 有 learnngd(梯度下降权值/阈值学习函数), learnngdm(梯度下降动量学习函数)等

(4) 训练函数 有 train,trainbfg(BFGS 准牛顿算法),traingd,traingdm 等;

(5) 性能函数 有 mse,msereg,mae 等

(6) 显示函数 有 plotperf,plotes(误差曲面),plotep,errsurf 等

示例 1: 分类问题的 BP 神经网络实现

训练样本集: $p=\begin{bmatrix} 1 & -1 & -2 & -4 \\ 2 & 1 & 1 & 0 \end{bmatrix}$, $t=[0.2 \ 0.8 \ 0.8 \ 0.2]$, t 为目标向量。将 p 分为两类。

Matlab 程序如下：

%定义输入向量和目标向量

p=[1 2;-1 1;-2 1;-4 0]';

t=[0.2 0.8 0.8 0.2];

%创建 BP 网络并定义训练函数及参数

net=newff([-1 1;-1 1],[5 1],{'logsig','logsig'},'traingd');

net.trainParam.goal=0.001;

```

net.trainParam.epochs=5000;
% 训练神经网络
[net,tr]=train(net,p,t);
save BPnet1 net;% 保存网络

```

```

% 仿真网络
load BPnet1 net;% 加载网络
p1=[1 2;-1 1;-2 1;-4 0]';
a2=sim(net,p1);
a2=a2>0.5

```

注：在新版本的 Matlab 中，创建前馈 BP 神经网络的函数已经更新为

feedforwardnet(hiddenSizes,trainFcn)，相应的程序修改为：

```

% 定义输入向量和目标向量
p=[1 2;-1 1;-2 1;-4 0]';
t=[0.2 0.8 0.8 0.2];
% 创建 BP 网络并定义训练函数及参数
feedforwardnet(5,'traingd');
% 训练神经网络
[net,tr]=train(net,p,t);
save BPnet1 net;% 保存网络

```

示例 2：BP 网络用于曲线拟合

直接给出 Matlab 程序：

```

% 训练
p=-1:0.1:0.9;
t=[-0.832 -0.423 -0.024 0.344 1.282 3.456 4.02 3.232 2.102 1.504 ...
    0.248 1.242 2.344 3.262 2.052 1.684 1.022 2.224 3.022 1.984];
net=feedforwardnet(15);
net.trainParam.epochs=2500;
net.trainParam.goal=0.001;
net.trainParam.show=10;
net.trainParam.lr=0.005;
net=train(net,p,t);
save BPnet2 net;
% 仿真
p=-1:0.1:0.9;
t=[-0.832 -0.423 -0.024 0.344 1.282 3.456 4.02 3.232 2.102 1.504 ...
    0.248 1.242 2.344 3.262 2.052 1.684 1.022 2.224 3.022 1.984];
hold on
plot(p,t,'r*');
load BPnet2 net;
r=sim(net,p);
plot(p,r);
perf=perform(net,t,r)

```

hold off;

13.4 RBF 神经网络的 Matlab 相关函数

RBF 神经网络的主要特点有：（1）RBF 神经网络只有一个隐层，结构简单；（2）RBF 神经网络的隐层神经元和输出层神经元的模型不同，在网络中起到的作用也是不同的；（3）RBF 神经网络的隐层是非线性的，输出层是线性的；（4）RBF 神经网络的基函数计算的是输入向量和中心的欧式距离，以此作为自变量；（5）RBF 神经网络使用局部指数衰减的非线性函数对非线性输入输出映射进行局部逼近。它的优点有：（1）具有全局的非线性逼近能力；（2）相比其他神经网络具有更简单的结构¹；（3）因为神经元的局部调整而具有更快的学习速率。它在非线性函数逼近、时间序列分析、模式识别、信息处理、数据分类、图像处理、系统建模等方面具有广泛的应用。

Matlab 工具箱中与 RBF 神经网络相关的函数：

newrb: 新建一个 RBF 神经网络，逐个增加隐层神经元的个数，使误差满足要求。

`net = newrb(P,T,goal,spread,MN,DF)`

P: R-by-Q matrix of Q input vectors

T: S-by-Q matrix of Q target class vectors

goal: Mean squared error goal (default = 0.0)

spread: Spread of radial basis functions (default = 1.0)

MN: Maximum number of neurons (default is Q)

DF: Number of neurons to add between displays (default = 25)

newrbe:快速创建一个 RBFNN，径向基神经元的数目等于输入样本数，如果输入像两个数很多时会导致网络规模很大，所以更常用的是 newrb 函数。

`net = newrbe(P,T,spread)`

P: RxQ matrix of Q R-element input vectors

T: SxQ matrix of Q S-element target class vectors

spread: Spread of radial basis functions (default = 1.0)

例 3: RBFNN 用于曲线拟合（续例 2）

Matlab 程序如下：

%创建网络

`p=-1:0.1:0.9;`

`t=[-0.832 -0.423 -0.024 0.344 1.282 3.456 4.02 3.232 2.102 1.504 ...`

```

0.248 1.242 2.344 3.262 2.052 1.684 1.022 2.224 3.022 1.984];
net=newrb(p,t,0.1,0.2,20,5);
save RBFnet1 net;
% 仿真网络
p=-1:0.1:0.9;
t=[-0.832 -0.423 -0.024 0.344 1.282 3.456 4.02 3.232 2.102 1.504 ...
    0.248 1.242 2.344 3.262 2.052 1.684 1.022 2.224 3.022 1.984];
hold on
plot(p,t,'r*');
load RBFnet1 net;
i=-1:0.05:0.9;
r=sim(net,i);
plot(i,r);
hold off;

```

13.5 遗传算法

1、遗传算法原理

遗传算法是一种基于自然群体遗传演化机制的高效探索算法，它摒弃了传统的搜索方式，模拟自然界生物进化过程，采用人工进化的方式对目标空间进行随机化搜索，它将问题域中的可能解看作是群体的一个个体或染色体，并将每一个个体编码成符号串形式，模拟达尔文的遗传选择和自然淘汰的生物进化过程，对群体反复进行基于遗传学的操作(选择，交叉、变异)，根据预定的目标适应度函数对每个个体进行评价，依据适者生存，优胜劣汰的进化规则，不断得到更优的群体，同时以全局并行搜索方式来搜索优化群体中的最优个体，求得满足要求的最优解。

遗传算法在生产调度、自动控制、机器人学、图像处理等方面都有广泛的应用。

2、遗传算法的步骤

遗传算法包括三个基本的操作：选择、交叉和变异，分别有不同的操作方法：

1、选择

选择的含义为确定重组或交叉个体，以及被选个体将产生多少个子代个体。选择的标准一般是按照适应度来进行，适应度有两种方法计算：1) 按比例计算；2) 基于排序的适应度计算。适应度计算之后是实际的选择，选择方法有：

- 1) 轮盘赌选择方法；
- 2) 随机遍历抽样；
- 3) 局部选择；

4) 截断选择;

5) 锦标赛选择。

2、交叉

交叉是结合来自父代交配种群中的信息产生新的个体, 依据个体编码表示方法的不同有以下算法:

1) 实值重组: 离散重组; 中间重组; 线性重组等;

2) 二进制交叉: 单点交叉; 多点交叉; 均匀交叉等

3、变异

交叉之后子代经历的变异, 实际上是子代基因按小概率扰动产生的变化。依据个体编码方式的不同, 有两种方法: 实值变异和二进制变异。

3、Matlab 遗传算法工具箱

例 4:
$$f(X) = [0.01 + \sum_{i=1}^5 \frac{1}{i + (x_i - 1)^2}]^{-1}, -10 \leq x \leq 10, i = 1, 2, \dots, 5$$

解: 首先建立适应函数:

```
function y=FitFun(x)
y=0.01;
for i=1:5
    y=y+1/(i+(x(i)-1)^2);
end
y=1/y;
```

然后打开 optimtool, 选择其中的 ga 求解器, 在 Fitness function 一栏中输入 @FitFun,

Variable number 输入 5, 然后 start 即可。

例 5:
$$\min f(s, t) = s^2 + 2t^2 - 4s - 8t + 15, s.t. \begin{cases} 9 - s^2 - t^2 \geq 0 \\ s \geq 0, t \geq 0 \end{cases}$$

解: 首先分别建立适应函数和非线性约束函数的 m 文件

```
function y=FitFun1(x)
y=x(1)^2+2*x(2)^2-4*x(1)-8*x(2)+15;

function [c,ceq]=NonCon1(x)
c=x(1)^2+x(2)^2-9;
ceq=[];
```

然后分别在对应位置输入 @FitFun1 和 @NonCon1, 点击 start。

14、Simulink 初步

SIMULINK 是 MATLAB 软件的扩展,它是实现动态系统建模和仿真的一个软件包,它与 MATLAB 语言的主要区别在于,其与用户交互接口是基于 Windows 的模型化图形输入,其结果是使得用户可以把更多的精力投入到系统模型的构建,而非语言的编程上。

所谓**模型化图形输入**是指 SIMULINK 提供了一些按功能分类的基本的系统模块,用户只需要知道这些模块的输入输出及模块的功能,而不必考察模块内部是如何实现的,通过对这些基本模块的调用,再将它们连接起来就可以构成所需要的系统模型(以.mdl 文件进行存取),进而进行仿真与分析。

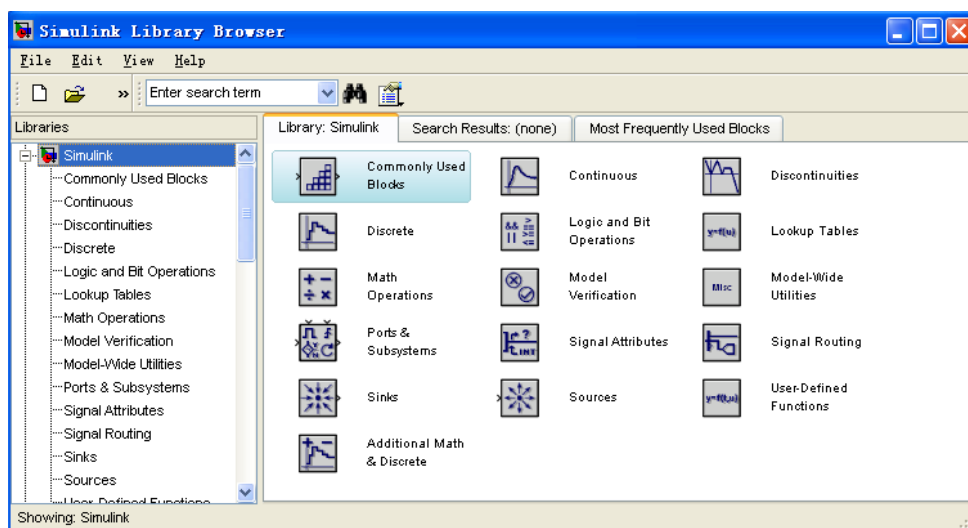
14.1 什么是 Simulink

Simulink 是 Matlab 提供的实现动态系统建模和仿真的一个软件包。使用户把精力从编程转向模型的构造。

使用 Simulink 构造好模型后,用户可以进行仿真,等待结果,或者改变参数,再运行。至于像各个模块在运行时如何执行,时间是如何采样,时间是如何驱动等细节性问题根本不用操心, Simulink 都已经替我们做好了。

可以在命令行输入 Simulink 命令即可打开 Simulink 模块库浏览器。

通过文件/新建/Model 命令可以新建一个名为 untitled.mdl 的模型编辑窗口,同时可以打开模块库浏览器。



14.2 Simulink 仿真模型的组成

一个典型的 Simulink 仿真模型由以下三种类型的模块组成：

(1) 信号源模块。

信号源为系统的输入，包括常数信号源、函数信号发生器以及用户自定义的各种信号。

(2) 被模拟的系统模块。

系统模块作为仿真的中心模块，是 Simulink 仿真建模所要解决的主要问题。

(3) 输出显示模块。

系统的输出由显示模块接收，输出显示的形式包括图形显示、示波器显示和输出到文件或 Matlab 工作空间三种，输出模块主要在 Sinks 模块库中。

Simulink 的三个特性：

(1) 和实际示波器输出相似的图形化显示结果的功能。

(2) 层次性。顶层系统和子系统的概念。

(3) Simulink 为用户提供了一种封装子系统的功能，用户自定义系统的图标和设置参数对话框。

1、模块的编辑

主要包括添加模块、选取模块、复制和删除模块、模块外形的调整（包括改变大小、方向、颜色和给模块添加阴影）和模块名的处理（包括隐藏模块名和修改模块名及相关字体）。

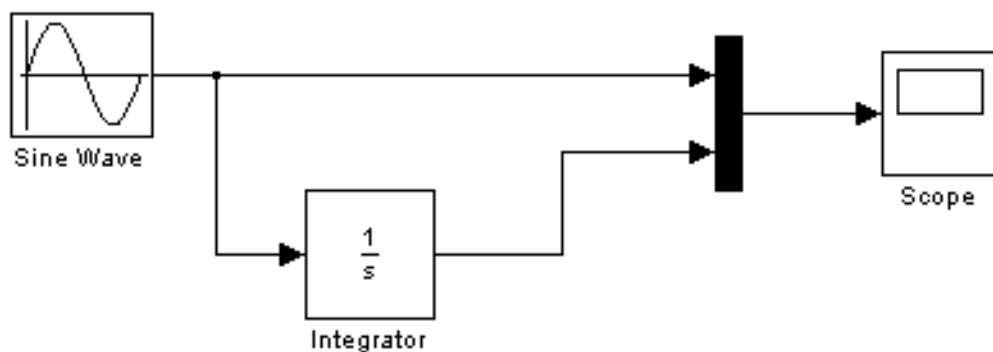
2、模块的连接

主要包括连接两个模块，模块间连线的调整，标注连线，连线分支处理和删除连线等操作。

注意：

(1) 可以通过按住 shift 键绘制斜线。

(2) 用户界面里不能有孤立的模块存在，即不能有和别的模块没有任何连接的模块。如下图所示：

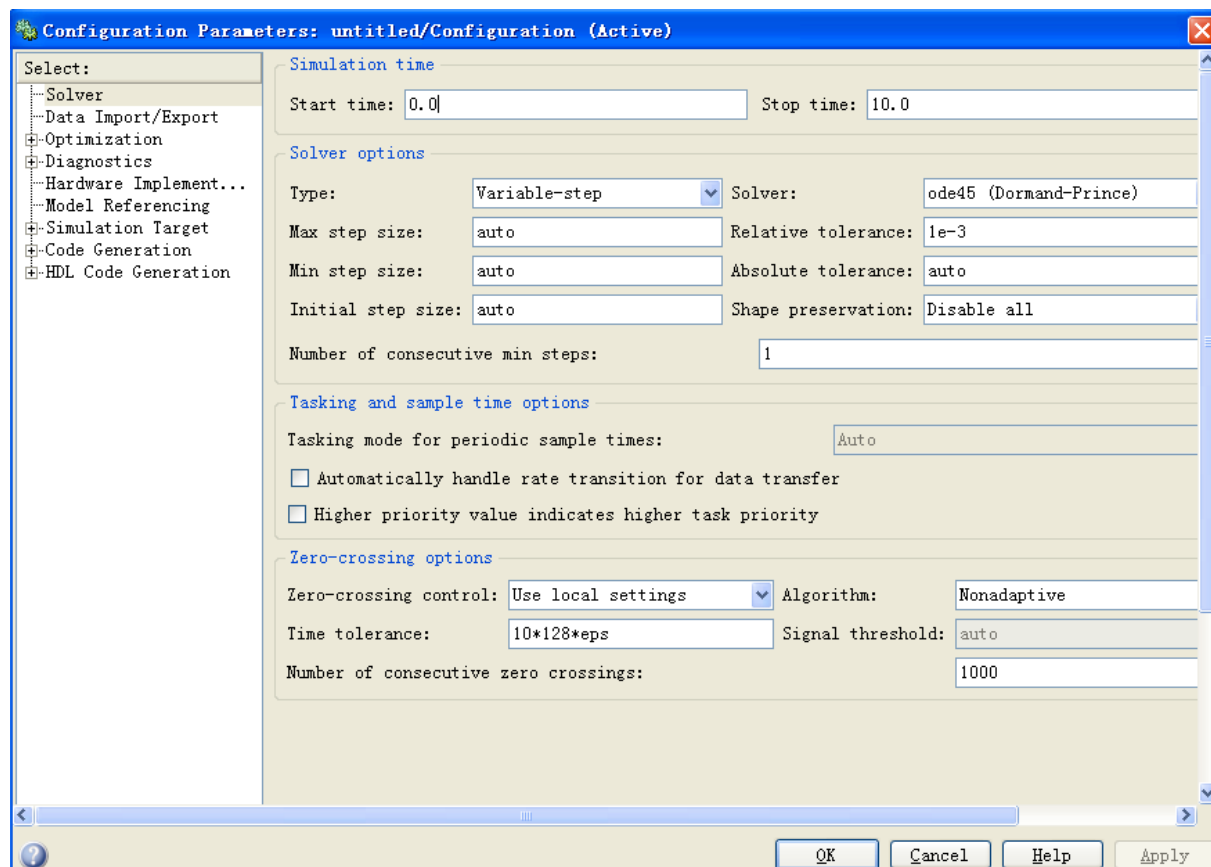


3、模块的参数设置

通过双击添加的模块可以打开模块的参数设置对话框。

4、仿真参数设置

设置仿真参数和选择解法器，选择 **Simulation** 菜单下的 **Parameters** 命令，就会弹出一个仿真参数对话框，它主要用三个页面来管理仿真的参数。

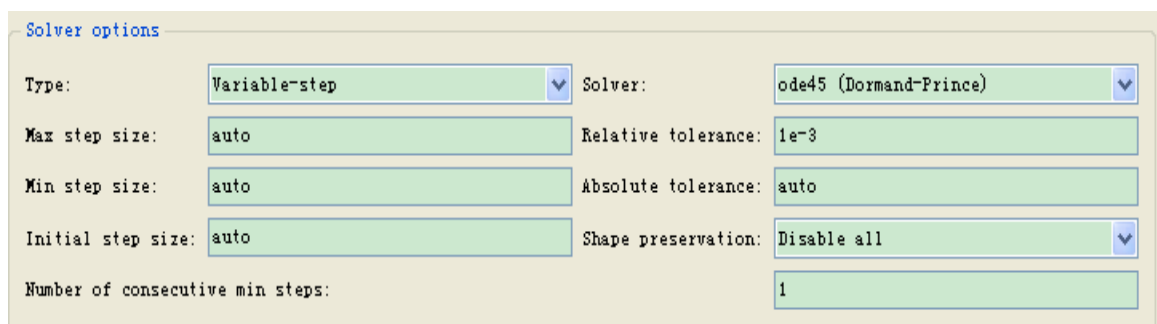


(1) Solver 页

此页可以进行的设置有：选择仿真开始和结束的时间；选择解法器，并设定它的参数；选择输出项。

1) 仿真时间：注意这里的时间概念与真实的时间并不一样，只是计算机仿真中对时间的一种表示，比如 10 秒的仿真时间，如果采样步长定为 0.1，则需要执行 100 步，若把步长减小，则采样点数增加，那么实际的执行时间就会增加。一般仿真开始时间设为 0，而结束时间视不同的因素而选择。总的说来，执行一次仿真要耗费的时间依赖于很多因素，包括模型的复杂程度、解法器及其步长的选择、计算机时钟的速度等等。

2) 仿真步长模式：用户在 Type 后面的第一个下拉选项框中指定仿真的步长选取方式，可供选择的有 Variable-step（变步长）和 Fixed-step（固定步长）方式。变步长模式可以在仿真的过程中改变步长，提供误差控制和过零检测。固定步长模式在仿真过程中提供固定的步长，不提供误差控制和过零检测。用户还可以在第二个下拉选项框中选择对应模式下仿真所采用的算法。



| Solver options | | | |
|----------------------------------|---------------|---------------------|------------------------|
| Type: | Variable-step | Solver: | ode45 (Dormand-Prince) |
| Max step size: | auto | Relative tolerance: | 1e-3 |
| Min step size: | auto | Absolute tolerance: | auto |
| Initial step size: | auto | Shape preservation: | Disable all |
| Number of consecutive min steps: | | 1 | |

变步长模式解法器有：ode45，ode23，ode113，ode15s，ode23s，ode23t，ode23tb 和 discrete。

ode45：缺省值，四/五阶龙格-库塔法，适用于大多数连续或离散系统，但不适用于刚性（stiff）系统。它是单步解法器，也就是，在计算 $y(t_n)$ 时，它仅需要最近处理时刻的结果 $y(t_{n-1})$ 。一般来说，面对一个仿真问题最好是首先试试 ode45。

ode23：二/三阶龙格-库塔法，它在误差限要求不高和求解的问题不太难的情况下，可能会比 ode45 更有效。也是一个单步解法器。

ode113：是一种阶数可变的解法器，它在误差容许要求严格的情况下通常比 ode45 有效。ode113 是一种多步解法器，也就是在计算当前时刻输出时，它需要以前多个时刻的解。

ode15s：是一种基于数字微分公式的解法器（NDFs）。也是一种多步解法器。适用于刚性系统，当用户估计要解决的问题是比较困难的，或者不能使用 ode45，或者即使使用效果也不好，就可以用 ode15s。

ode23s：它是一种单步解法器，专门应用于刚性系统，在弱误差允许下的效果好于

ode15s。它能解决某些 ode15s 所不能有效解决的 stiff 问题。

ode23t: 是梯形规则的一种自由插值实现。这种解法器适用于求解适度 stiff 的问题而用户又需要一个无数字振荡的解法器的情况。

ode23tb: 是 TR-BDF2 的一种实现, TR-BDF2 是具有两个阶段的隐式龙格-库塔公式。

discretet: 当 Simulink 检查到模型没有连续状态时使用它。

固定步长模式解法器有: ode5, ode4, ode3, ode2, ode1 和 discrete。

ode5: 缺省值, 是 ode45 的固定步长版本, 适用于大多数连续或离散系统, 不适用于刚性系统。

ode4: 四阶龙格-库塔法, 具有一定的计算精度。

ode3: 固定步长的二/三阶龙格-库塔法。

ode2: 改进的欧拉法。

ode1: 欧拉法。

discrete: 是一个实现积分的固定步长解法器, 它适合于离散无连续状态的系统。

3) 步长参数

步长参数: 对于变步长模式, 用户可以设置最大的和推荐的初始步长参数, 缺省情况下, 步长自动地确定, 它由值 auto 表示。

Maximum step size (最大步长参数): 它决定了解法器能够使用的最大时间步长, 它的缺省值为“仿真时间/50”, 即整个仿真过程中至少取 50 个取样点, 但这样的取法对于仿真时间较长的系统则可能带来取样点过于稀疏, 而使仿真结果失真。一般建议对于仿真时间不超过 15s 的采用默认值即可, 对于超过 15s 的每秒至少保证 5 个采样点, 对于超过 100s 的, 每秒至少保证 3 个采样点。

Initial step size (初始步长参数): 一般建议使用“auto”默认值即可。

4) 仿真精度的定义 (对于变步长模式)

Relative tolerance (相对误差): 它是指误差相对于状态的值, 是一个百分比, 缺省值为 1e-3, 表示状态的计算值要精确到 0.1%。

Absolute tolerance (绝对误差): 表示误差值的门限, 或者说是在状态值为零的情况下, 可以接受的误差。如果它被设成了 auto, 那么 simulink 为每一个状态设置初始绝对误差为 1e-6。

5) Mode (固定步长模式选择)

Multitasking: 选择这种模式时，当 simulink 检测到模块间非法的采样速率转换，它会给出错误提示。所谓的非法采样速率转换指两个工作在不同采样速率的模块之间的直接连接。在实时多任务系统中，如果任务之间存在非法采样速率转换，那么就有可能出现一个模块的输出在另一个模块需要时却无法利用的情况。通过检查这种转换，Multitasking 将有助于用户建立一个符合现实的多任务系统的有效模型。

使用速率转换模块可以减少模型中的非法速率转换。Simulink 提供了两个这样的模块：unit delay 模块和 zero-order hold 模块。对于从慢速率到快速率的非法转换，可以在慢输出端口和快输入端口插入一个单位延时 unit delay 模块。而对于快速率到慢速率的转换，则可以插入一个零阶采样保持器 zero-order hold。

Singletasking: 这种模式不检查模块间的速率转换，它在建立单任务系统模型时非常有用，在这种系统就不存在任务同步问题。

Auto: 这种模式，simulink 会根据模型中模块的采样速率是否一致，自动决定切换到 multitasking 和 singletasking。

6) 输出选项

Refine output: 这个选项可以理解成精细输出，其意义是在仿真输出太稀疏时，simulink 会产生额外的精细输出，这一点就像插值处理一样。用户可以在 refine factor 设置仿真时间步间插入的输出点数。

产生更光滑的输出曲线，改变精细因子比减小仿真步长更有效。精细输出只能在变步长模式中才能使用，并且在 ode45 效果最好。

Produce additional output: 它允许用户直接指定产生输出的时间点。一旦选择了该项，则在它的右边出现一个 output times 编辑框，在这里用户指定额外的仿真输出点，它既可以是一个时间向量，也可以是表达式。与精细因子相比，这个选项会改变仿真的步长。

Produce specified output only: 它的意思是让 simulink 只在指定的时间点上产生输出。为此解法器要调整仿真步长以使之和指定的时间点重合。这个选项在比较不同的仿真时可以确保它们在相同的时间输出。

(2) Data Import/Export 页

Load from workspace: 选中前面的复选框即可从 MATLAB 工作空间获取时间和输入变量，一般时间变量定义为 t，输入变量定义为 u。Initial state 用来定义从 MATLAB 工作空间获得的状态初始值的变量名。

Save to workspace: 用来设置存往 MATLAB 工作空间的变量类型和变量名，选中变量类型前的复选框使相应的变量有效。一般存往工作空间的变量包括输出时间向量（Time）、状态向量（States）和输出变量（Output）。Final state 用来定义将系统稳态值存往工作空间所使用的变量名。

Save option: 用来设置存往工作空间的有关选项。Limit rows to last 用来设定 SIMULINK 仿真结果最终可存往 MATLAB 工作空间的变量的规模，对于向量而言即其维数，对于矩阵而言即其秩；Decimation 设定了一个亚采样因子，它的缺省值为 1，也就是对每一个仿真时间点产生值都保存，而若为 2，则是每隔一个仿真时刻才保存一个值。Format 用来说明返回数据的格式，包括矩阵 matrix、结构 struct 及带时间的结构 struct with time。

（3）Diagnostics 页

此页分成两个部分：仿真选项和配置选项。

配置选项下的列表框主要列举了一些常见的事件类型，以及当 SIMULINK 检查到这些事件时给予的处理。

仿真选项 options 主要包括是否进行一致性检验、是否禁用过零检测、是否禁止复用缓存、是否进行不同版本的 SIMULINK 的检验等几项。

5、启动仿真

设置仿真参数和选择解法器之后，就可以启动仿真而运行。

选择 Simulink 菜单下的 start 选项来启动仿真，如果模型中有些参数没有定义，则会出现错误信息提示框。如果一切设置无误，则开始仿真运行，结束时系统会发出一鸣叫声。

除了直接在 SIMULINK 环境下启动仿真外，还可以在 MATLAB 命令窗口中通过函数进行，格式如下：

$[t,x,y]=\text{sim}(\text{'模型文件名'},[to\ tf],\text{simset}(\text{'参数 1'},\text{参数值 1},\text{'参数 2'},\text{参数值 2},\dots))$

其中 to 为仿真起始时间，tf 为仿真终止时间。 $[t,x,y]$ 为返回值，t 为返回的时间向量值，x 为返回的状态值，y 为返回的输出向量值。simset 定义了各种仿真参数。

14.3 Simulink 主要模块介绍

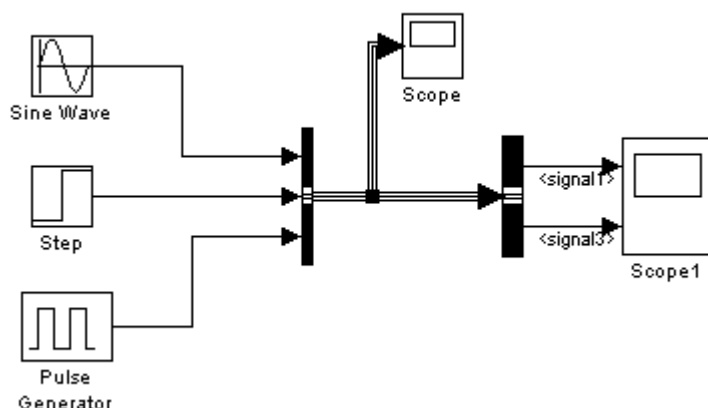
1、常用模块库

常用模块库是为了加速建模速度、节省时间建模过程中寻找模块的时间而将最常用

的基本模块集中放在一起形成的。主要包括以下部分：

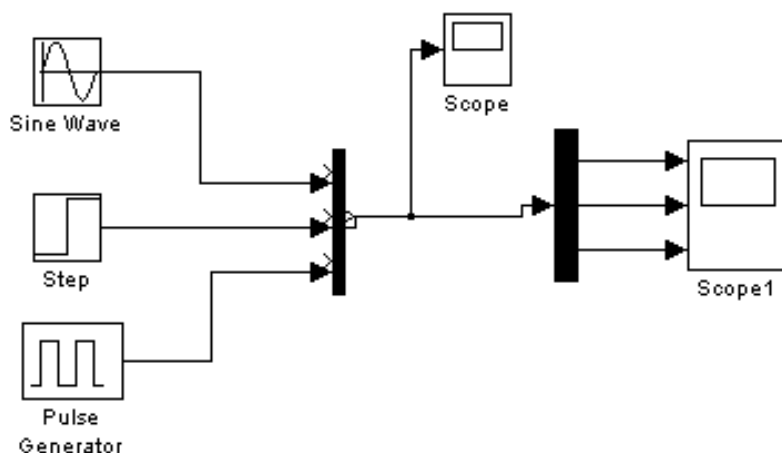
（1）总线信号生成与总线信号选择模块

Bus Creator 模块用于将多个信号合成为一个总线信号，常用于子系统接口信号传递；Bus Selector 模块用来选择总线信号中的一个或多个。



（2）信号合成与信号分离模块

Mux 模块和 Demux 模块的功能对所有的信号进行合成与分离，这与总线选择模块可任意选择总线上的信号进行输出不同。



（3）数据类型转换模块

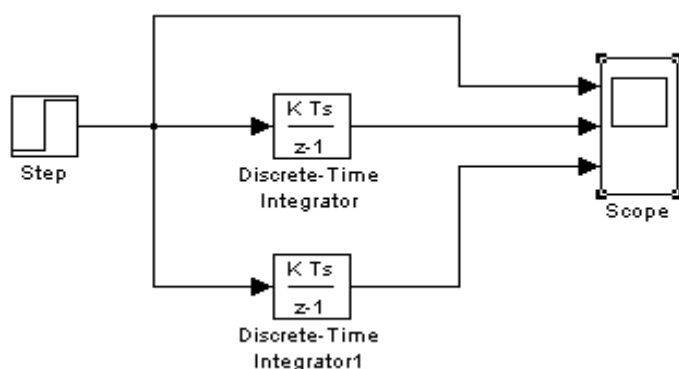
Data Type Conversion 可将输入数据转换为指定输出类型，具体的选择有 Inherit(与输入保持一致)、Double、Single、Int8、UInt8 等。输入输出数据上可以选择 Real World Value（实数值相等）或者 Stored Integer(存储整数相等)，还可选择取整方向。

（4）积分模块

Integer 模块为连续时间积分单元，可以在参数设置对话框中的 Initial condition 文本框中设置积分器初始值，在 Limit output 文本框中设置积分器输出最大和最小值。

(5) 离散时间积分模块

Discrete Time Integrator 模块可完成离散系统积分作用。其参数设置中的 GainValue 设置积分增益值，来改变积分速度，Sample Time 中设置离散积分的采样时间，如果为 -1 表示与输入信号采样时间保持一致；Limit Output 设置离散积分输出上下限，即设置积分饱和值。



(6) 乘法与加法模块

Product 模块可以用来求输入信号的乘积，可以修改输入信号的端口数完成多个信号之间的乘积。

Sum 模块用来求输入信号的加法、减法操作，可以选择加减法符号的位置和该模块的外观为方形或者圆形。

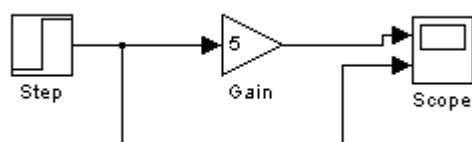
(7) 关系操作符以及逻辑操作模块

Relational Operator 模块用来比较两个输入信号的大小关系；

Logic Operator 模块用来求取两输入变量的逻辑操作关系。

(8) 增益与饱和模块

Gain 模块可用来设置信号放大倍数，在动态仿真中使用频繁。



(9) 输入/输出接口与子系统模块

In1 模块在建立子系统时作为输入信号的接口，Out1 作为输出信号的接口，

Subsystem 模块可用来将复杂系统的全部或局部生成为一个子系统，用来简化 Simulink 模型的结构。

(10) 终端模块

Terminator 模块可用来连接没有与其他模块相连的输出端口，在 Simulink 模型中，如果有输出端口没有连接，运行仿真时会在 Matlab 窗口中显示警告信息，使用终端模块可以避免这类警告信息的出现。

2、连续系统模块库

提供了连续系统 Simulink 建模和方针的基本模块，如 Derivative 模块，Integrator 模块，State-Space 模块，Transfer Fun 模块，Transport Delay 模块，Variable Transport Delay 以及 zero-Pole 等模块。

3、非连续系统模块库

Discontinites 模块库也称为非线性模块库，包含一些常用的非线性运算模块。

4、离散系统模块库

主要包括用于建立离散采样系统的模块

5、逻辑与位操作模块库

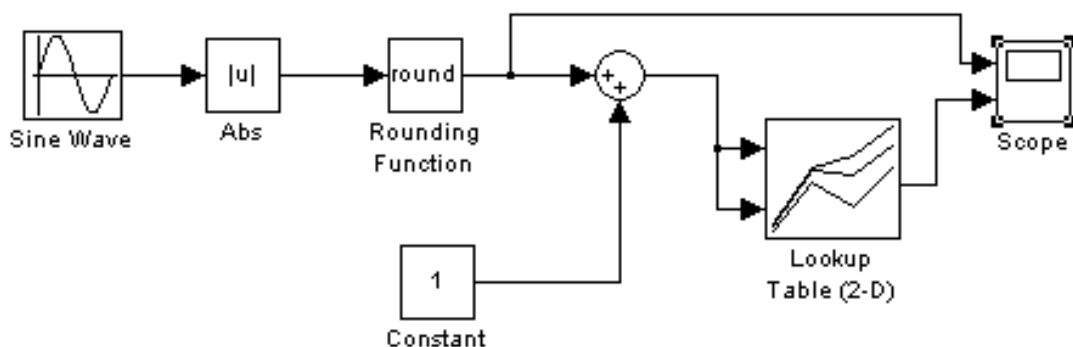
提供了建立逻辑系统及数字系统 Simulink 仿真建模的基本模块。

6、数学操作模块库

提供了与数学运算相关的仿真模块。

7、表格查询模块

Lookup Tables 用来建立一维、二维或者多维表格查询的 Simulink 仿真模型。



8、端口与子系统模块

Ports&Subsystems 模块用于建立各类子系统模型。

9、信号属性操作模块库

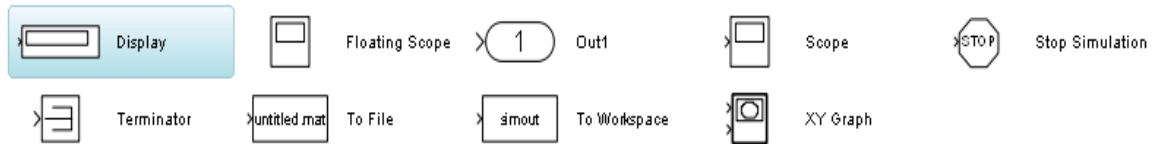
Singal Attribute 模块库可用于信号系统的 Simulink 建模与仿真。

10、信号路由模块库

Signal Routing 模块库用于控制信号的传递路径。

11、接收模块库

Sinks 主要包括 Model & Subsystem Outputs, Data Viewers 和 Simulation Control 三部分



To File 模块：将数据输出到文件

To Workspace 模块：该模块将与其相连接的数据输出到工作空间，可以设置输出到工作空间的变量名称和数据存储格式，通常选择 Array 格式。

XY Graph 模块，以第一个输入端口为 X 轴，第二个为 Y 轴坐标绘制图形。

Floating Scope：没有固定的输入端口，但需要首先设置信号。

Display：以数字形式显示当前输入的变量数值。

12、信号源模块库

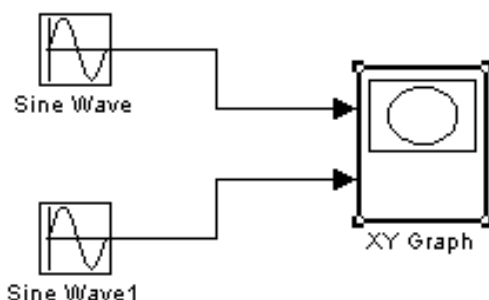
包括各种信号生成模块

13、用户自定义功能模块

14.4 仿真示例

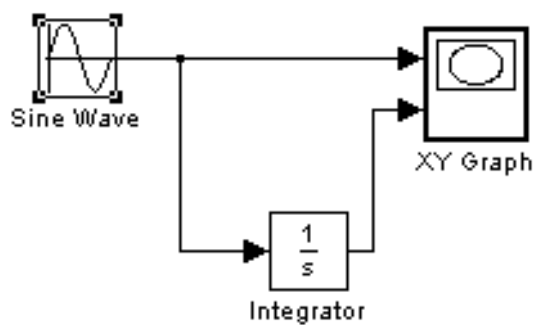
例 1：动态画圆

$$(1) \quad x = \cos t, y = \sin t$$



说明：Sine Wave 中 Phase(相位)为 $\pi/2$ ，实际为 $\cos(t)$ ；Sine Wave1 中 Phase 为 0。

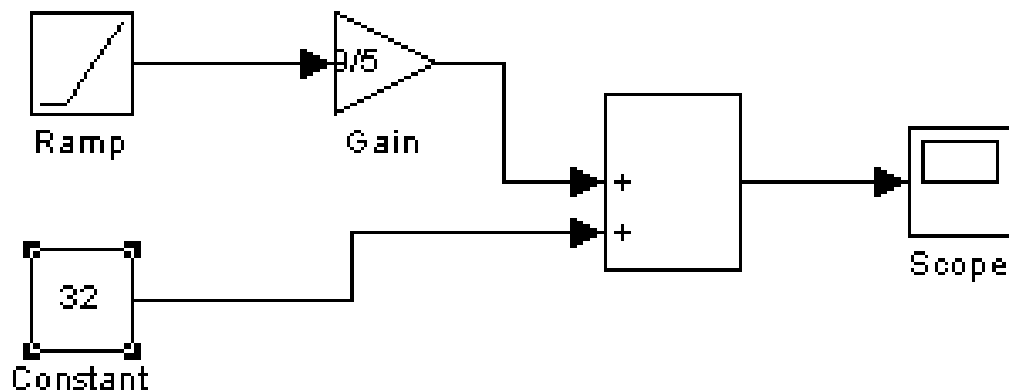
$$(2) \quad x = \cos t, y = \int_0^t x(t)dt$$



说明: Sine Wave 中 Phase(相位)为 $\pi/2$, 实际为 $\cos(t)$; Integrator 中 Initial condition(初始值)为 0.

XY Graph 中, x 的范围为 $-1.5 \sim 1.5$, y 的范围为 $-1.2 \sim 1.2$.

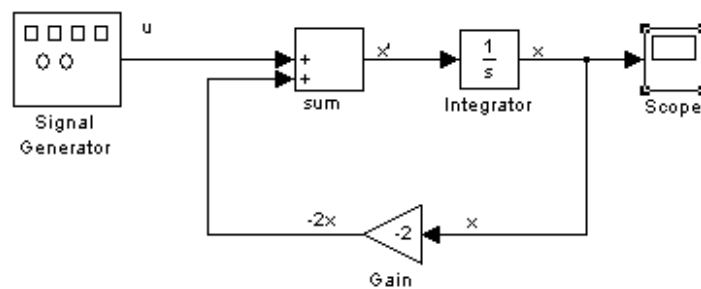
例 2: 摄氏温度转化成华氏温度



其中的 Ramp 模块用来产生摄氏温度

例 3: 简单连续系统的建模 $x'(t) = -2x(t) + u(t)$

其中 $u(t)$ 是幅度为 1, 频率为 1rad/s 的方波信号, 积分模块将 $x(t)$ 的微分信号积分来获得 $x(t)$ 。需要用到的模块有 Gain 模块, Sum 模块, Signal Generator 模块 (Square 方波), Integrator 模块, Scope 模块。模型如下所示:



注: x 是 Integrator 模块的输出, 它同样计算 x' 的模块的输出, 这个关系是通过模型

中的环路实现的。

例 4：二阶微分方程系统建模

$$m\ddot{x} + c\dot{x} + kx = 0, x(0) = x_0 = 1, \dot{x}(0) = \dot{x}_0 = 0$$

要求：采用 Simulink 对系统进行仿真，已知参数 $m=1$ ， $c=1$ ， $k=1$ ，解析解为：

$$X(t) = e^{-\zeta\omega_0 t} \left[x_0 \cos(\omega_d t) + \frac{\dot{x}_0 + \zeta\omega_0 x_0}{\omega_d} \sin(\omega_d t) \right]$$

其中： $\omega_0 = \sqrt{\frac{k}{m}}$ 系统固有频率

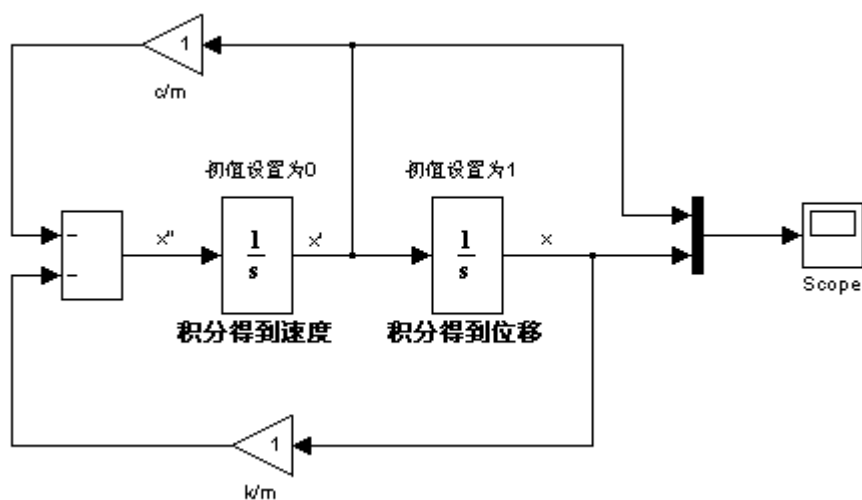
$\omega_d = \omega_0 \sqrt{1 - \zeta^2}$ 阻尼固有频率

$\zeta = \frac{c}{2\sqrt{km}}$ 相对阻尼系数

原微分方程可以写为：

$$\ddot{x} + \frac{c}{m} \dot{x} + \frac{k}{m} x = 0$$

系统模型如下图所示：

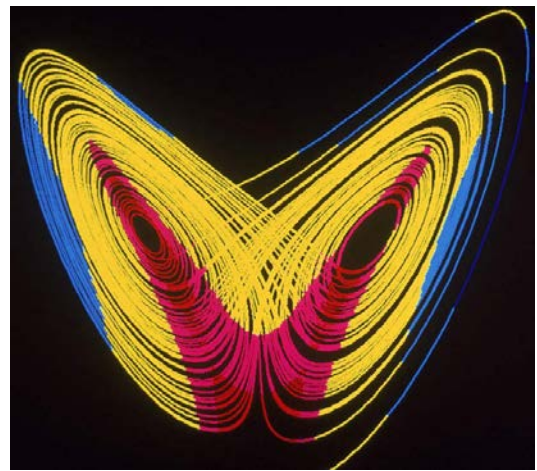
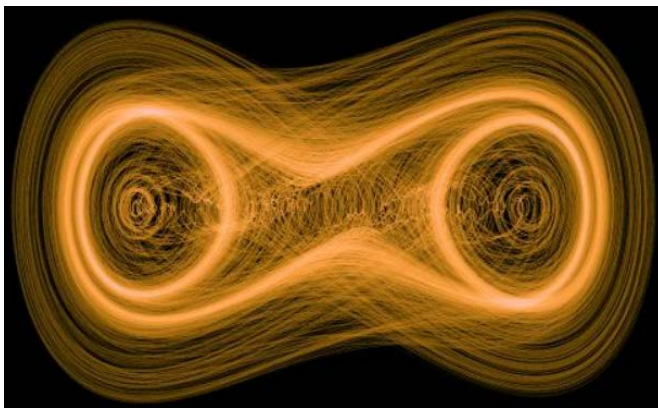


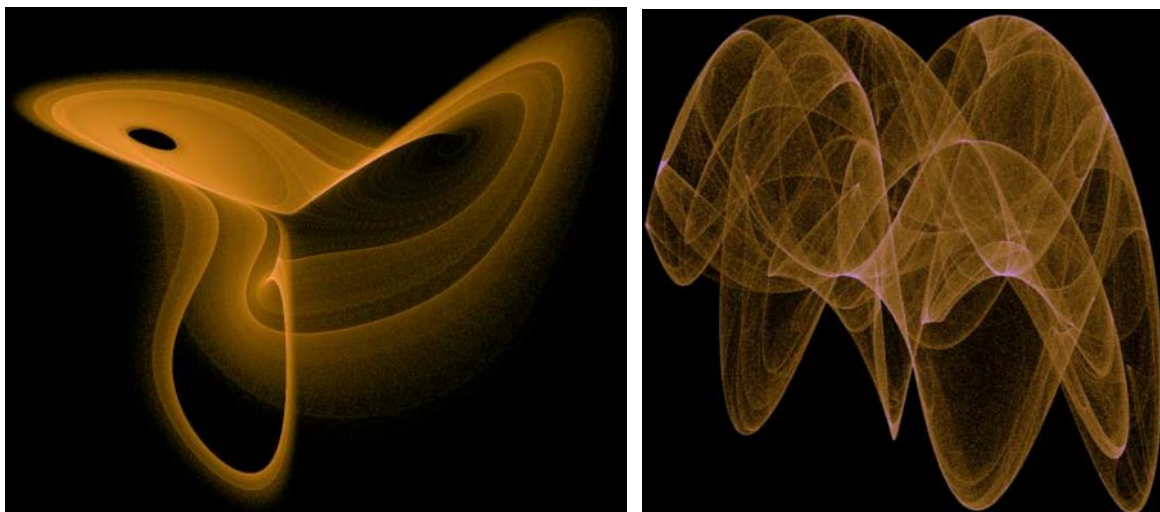
更多复杂的例子可以通过 `start/simulink/demos` 命令查看。

15、混沌与分形

15.1 混沌

混沌理论（Chaos theory）是一种兼具质性思考与量化分析的方法，用以探讨动态系统中（如：人口移动、化学反应、气象变化、社会行为等）无法用单一的数据关系，而必须用整体、连续的数据关系才能加以解释及预测之行为。混沌理论是对确定性非线性动力系统的不稳定非周期性行为的定性研究（Kellert, 1993）。混沌是非线性系统所独有且广泛存在的一种非周期运动形式，其覆盖面涉及到自然科学和社会科学的几乎每一个分支。近二三十年来，近似方法、非线性微分方程的数值积分法，特别是计算机技术的飞速发展，为人们对混沌的深入研究提供了可能，混沌理论研究取得的可喜成果也使人们能够更加全面透彻地认识、理解和应用混沌。1963年美国气象学家爱德华·诺顿·洛伦兹提出混沌理论，非线性系统具有的**多样性和多尺度性**。混沌理论解释了决定系统可能产生随机结果。混沌理论认为在混沌系统中，**初始条件十分微小的变化，经过不断放大**，对其未来状态会造成极其巨大的差别。理论的最大的贡献是用简单的模型获得明确的非周期结果。在气象、航空及航天等领域的研究里有重大的作用。





混沌系统的三个基本特征：

1) 内在随机性：从确定性非线性系统的演化过程看他们在混沌区的行为都表现出随机不确定性。然而这种不确定性不是来源于外部环境的随机因素对系统运动的影响，而是系统自发产生的。（不确定性）

2) 初值敏感性：对于没有内在随机性的系统，只要两个初始值足够接近从他们出发的两条轨线在整个系统演化过程中都将保持足够接近，但是对具有内在随机性的混沌系统而言，从两个非常接近的初值出发的两个轨线在经过长时间演化之后，可能变得相距“足够远”，表现出对初值的极端敏感，即所谓“失之毫厘，谬之千里。”（不可预测）

3) 非规则的有序：混沌不是纯粹的无序，而是不具备周期性和其他明显对称特征的有序态，确定性的非线性系统的控制参量按一定方向不断变化，当达到某种极限状态时，就会出现混沌这种非周期运动体制。但是非周期运动不是无序运动，而是另一种类型的有序运动。混沌区的系统行为往往体现出无穷嵌套自相似结构，这种不同层次上的结构相似性是标度变换下的不变性，这种不变性体现出混沌运动的规律。（不可重复）

1、奇异吸引子（Strange Attractor）

$$\begin{cases} \frac{dx}{dt} = -10x + 10y, \\ \frac{dy}{dt} = \mu x - y - xz, \\ \frac{dz}{dt} = -\frac{8}{3}z + xy. \end{cases}$$

奇异吸引子,又被称为奇怪吸引子或混沌吸引子,它是指事物在非线性(混沌)运动过程中,对其运行轨迹(秩序)具有关键性决定作用的范式,是混沌运动过程中内在的根本性

结构。奇异吸引子具有分形结构和自相似性特征。

奇异吸引子具有不同属性的内外两种运动模式:在奇异吸引子外的一切运动都趋向于吸引子,即事物表面看似杂乱无章的运动在按照某种既定的规律或秩序在进行,属于事物发展过程中关键性的稳定因素。但是在奇异吸引子内部随时都在产生不稳定的变化,属于可能改变事物运动轨迹的不稳定的因素。

2、倍周期分叉 (Period doubling Bifurcation)

倍周期分叉(period doubling bifurcation), 又称倍周期分岔、周期倍化分叉、叉形分岔, 是混沌理论中一个非常重要的概念, 是研究混沌理论必须理解的概念。

倍周期分叉过程是一条通向混沌的典型道路, 即可以认为是从周期窗口中进入混沌的一种方式。通过倍周期分叉到达混沌现象的过程中, 会依次经过周期 1, 周期 2, …… 周期 4, 混沌单吸引子和混沌双吸引子。

例 1:

```
function y=diedai(f,a,x1)
N=32;
y=zeros(N,1);
    for i=1:1e4
        x2=f(a,x1);
        x1=x2;
        y(mod(i,N)+1)=x2;
    end
end
f=@(a,x)a*sin(pi*x);
hold on;
for x0=-1:0.05:1
    for a=0:0.01:1
        y=diedai(f,a,x0);
        for count=1:32
            plot(a,y(count),'k.');
```

```
        hold on;
```

```
    end
```

```

        end
    end
    例 2:
    axis([2.7,4,0,1]);
    grid
    hold on
    for r = 2.7:0.005:3.9;
        x = 0.1;
        for i =2:200
             $x(i) = r * x(i-1) * (1-x(i-1));$ 
        end
        pause(0.01)
        for i=151:200
            plot(r,x(i),'k.');
```

```

        end
    end
end
```

3、李指数（Lyapunov 指数）

4、庞加莱截面（Poincare surface）

15.2 分形

分形（fractal）这个术语是美籍法国数学家 Mandelbrot 于 1975 年创造的。Fractal 出自拉丁语 fractus（碎片，支离破碎）、英文 fractured（断裂）和 fractional（碎片，分数），说明分形是用来描述和处理粗糙、不规则对象的。Mandelbrot 是想用此词来描述自然界中传统欧几里得几何学所不能描述的一大类复杂无规则的几何对象，如蜿蜒曲折的海岸线、起伏不定的山脉、令人眼花缭乱的漫天繁星等。它们的共同特点是极不规则

或极不光滑。



1975 年，Mandelbrot 出版了他的专著《分形对象：形、机遇与维数》，标志着分形理论的正式诞生。1982 年，随着他的名著 *The Fractal Geometry of Nature* 出版，分形这个概念被广泛传播，成为当时全球科学家们议论最为热烈、最感兴趣的热门话题之一。

目前还没有一个让各方都满意的分形定义，但在数学上，大家都认为分形有以下几个特点：

- (1) 具有无限精细的结构。
- (2) 有某种自相似的形式，可能是近似的或是统计的。
- (3) 一般它的分数维大于它的拓扑维数。
- (4) 可以由非常简单的方法定义，并由递归、迭代产生等。

上面的 (1) 和 (2) 两项说明分形在结构上的内在规律性。自相似是分形的灵魂，它使得分形的任何一个片段都包含了整个分形的信息。第 (3) 项说明了分形的复杂性。第 (4) 项则说明了分形的生成机制。分形是一种无限多层次自相似的、支离破碎的、奇异的图形。

Julia 集合与 Mandelbrot 集合是研究复平面上的迭代，考虑的是复平面上的一个二次映射 $P_c(z) = z^2 + c$, $z, c = a + bi \in \mathbb{C}$,

(17)

的迭代行为，即给定 z_0 和参数 c ，依次计算 $z_n = z_{n-1}^2 + c$, $n = 0, 1, \dots, N$ 。

二维实平面上的迭代为

$$\begin{cases} x_{n+1} = x_n^2 - y_n^2 + a, \\ y_{n+1} = 2x_n y_n + b. \end{cases} \quad n = 0, 1, \dots, N. \quad (18)$$

Julia 集是给定 c ，收敛的迭代中初始值 z_0 的集合。

Mandelbrot 集是给定初始值 $z_0 = 0$ ，收敛的迭代中参数 c 的集合。

算法 1 生成 Julia 集

- (1) 设定参数 $c = a + bi$ ，一个最大的迭代步数 N 以及一个界限值 $R \geq \sqrt{a^2 + b^2}$ ；
 (2) 对于平面上以 R 为半径的圆盘内的每一点进行迭代，使用式 (1) 迭代，如果对于所有的 $n \leq N$ ，都有 $|z_n| \leq R$ ，那么，绘制出相应的起始点，否则不绘制。

绘制的 Matlab 程序如下

```
clc, clear
a=-0.11; b=0.65; r=2; N=60; hold on
for x0=-r:0.02:r
    for y0=-r:0.02:r
        x(1)=x0; y(1)=y0;
        if x0^2+y0^2<=r^2
            for n=1:N
                x(n+1)=x(n)^2-y(n)^2+a;
                y(n+1)=2*x(n)*y(n)+b;
            end
            if x(end)^2+y(end)^2<r^2
                plot(x0,y0,'.')
            end
        end
    end
end
```

实践发现还有很多参数能够绘制出漂亮的分形图，例如参数 $(-0.19, 0.6557)$ 等。

□

复数运算的 Matlab 程序如下

```
clc, clear
c=-0.11+0.65*i; r=2; N=60; hold on
for x0=-r:0.02:r
    for y0=-r:0.02:r
        z0=x0+y0*i; z=z0;
        if abs(z0)<r
            for n=1:N
```



```

        z=z^2+c;
    end
    if abs(z)<r
        plot(z0, '.')
    end
end
end
end
end
end
end
end

```

算法 2 绘制 Mandelbrot 集合的点的分布图

(1) 设定一个最大的迭代步数 N 和一个界限值 R ，不妨设实数 $R=2$;

(2) 对于参数平面上每一点 c ，从 $z_0=0$ 开始迭代，如果对于所有的 $n \leq N$ ，都有 $|z_n| \leq R$ ，那么，绘制出相应的参数 c ，否则不绘制。

绘制的Matlab程序如下

```

clc, clear
r=2; N=20; hold on
for a=-r:0.02:r
    for b=-r:0.02:r
        x(1)=0; y(1)=0;
        for n=1:N
            x(n+1)=x(n)^2-y(n)^2+a; y(n+1)=2*x(n)*y(n)+b;
        end
        if x(end)^2+y(end)^2<r^2
            plot(a,b, '.')
        end
    end
end
end
end
end
end
end
end
end

```

例 1 用迭代算法生成 Sierpinski 三角形。

迭代算法的 Matlab 程序如下

```

clc, clear, N=20000; %画 N 个点

```

```

A=0; B=100; C=50+i*sqrt(3)*50; %用复数表示的三角形三个点 A，B，C 的坐标

```

```

P=10+20i; %任取三角形内的一点
TP=[]; %所有生成点的初始化
gailv=randperm(N); %产生 1 到 N 的随机全排列
for k=gailv
    if k<N/3+1;
        P=(P+A)/2; %生成新点为点 P 和 A 的中点
    elseif k<2*N/3+1
        P=(P+B)/2; %生成新点为点 P 和 B 的中点
    else
        P=(P+C)/2; %生成新点为点 P 和 C 的中点
    end
    TP=[TP,P]; %TP 中加入新点
end
plot(TP','',markersize',5) %画所有生成的新点

```

例 2 用递归算法生成 Sierpinski 三角形。

递归算法中点的相对位置见图 4 (a)，使用 Matlab 画出的图形效果见图 4 (b)。

递归算法的 Matlab 函数如下

```

function mysierpinski(x,y,L,n)
% x,y 为三角形中心点坐标，L 为三角形边长，n 为递归深度
hold on
if n==1
    x1=x-L/2; y1=y-L*tan(pi/6)/2; %计算三角形顶点的坐标
    x2=x+L/2; y2=y-L*tan(pi/6)/2;
    x3=x; y3=y+L*tan(pi/6)/2;
    plot([x1;x2],[y1;y2]); plot([x2;x3],[y2;y3]); plot([x3;x1],[y3;y1]) %画三角形的边
else
    x01=x-L/4; y01=y-L*tan(pi/6)/4; %计算小三角形中心的坐标
    x02=x+L/4; y02=y-L*tan(pi/6)/4;
    x03=x; y03=y+L*tan(pi/6)/4;
    mysierpinski(x01,y01,L/2,n-1) %递归调用

```

```
mysierpinski(x02,y02,L/2,n-1)
mysierpinski(x03,y03,L/2,n-1)
end
```

例 3 Koch 雪花曲线

```
p=[0 0;10 0]; %P 为初始两个点的坐标,第一列为 x 坐标,第二列为 y 坐标
n=2;%n 为结点数
A=[cos(pi/3) -sin(pi/3);sin(pi/3) cos(pi/3)]; %旋转矩阵
for k=1:4
d=diff(p)/3;%diff 计算相邻两个点的坐标之差,得到相邻两点确定的向量
%则 d 就计算出每个向量长度的三分之一,与题中将线段三等分对应
m=4*n-3;%迭代公式
q=p(1:n-1,:);%以原点为起点,前 n-1 个点的坐标为终点形成向量
p(5:4:m,:)=p(2:n,:);%迭代后处于 4k+1 位置上的点的坐标为迭代前的相应坐标
p(2:4:m,:)=q+d;%用向量方法计算迭代后处于 4k+2 位置上的点的坐标
p(3:4:m,:)=q+d+d*A';%用向量方法计算迭代后处于 4k+3 位置上的点的坐标
p(4:4:m,:)=q+2*d;%用向量方法计算迭代后处于 4k 位置上的点的坐标
n=m;%迭代后新的结点数目
end
plot(p(:,1),p(:,2))%绘出每相邻两个点的连线
axis([0 10 0 10])
```

例 4:

```
p=[0 0;10 0]; %P 为初始两个点的坐标,第一列为 x 坐标,第二列为 y 坐标
n=2; %n 为结点数
A=[0 -1;1 0]; %旋转矩阵
for k=1:4
d=diff(p)/3; %diff 计算相邻两个点的坐标之差,得到相邻两点确定的向量
%则 d 就计算出每个向量长度的三分之一,与题中将线段三等分对应
m=5*n-4; %迭代公式
q=p(1:n-1,:); %以原点为起点,前 n-1 个点的坐标为终点形成向量
p(6:5:m,:)=p(2:n,:); %迭代后处于 5k+1 位置上的点的坐标为迭代前的相应坐标
```

```

p(2:5:m,:)=q+d;          %用向量方法计算迭代后处于  $5k+2$  位置上的点的坐标
p(3:5:m,:)=q+d+d*A';    %用向量方法计算迭代后处于  $5k+3$  位置上的点的坐标
p(4:5:m,:)=q+2*d+d*A';  %用向量方法计算迭代后处于  $5k+4$  位置上的点的坐标
p(5:5:m,:)=q+2*d;        %用向量方法计算迭代后处于  $5k$  位置上的点的坐标
n=m;                      %迭代后新的结点数目
end
plot(p(:,1),p(:,2))      %绘出每相邻两个点的连线
axis([0 10 0 10])

```

例 5:

$p=[0\ 10;10\ 0;0\ -10;-10\ 0;0\ 10];$ %P 为四边形四个顶点的坐标,其中第五个点与第一个点重合,以便于绘图

%第一列为 x 坐标,第二列为 y 坐标

$n=5;$ %n 为结点数

$A=[\cos(-\pi/3)\ -\sin(-\pi/3);\sin(-\pi/3)\ \cos(-\pi/3)];$ %旋转矩阵,顺时针旋转 60 度

for k=1:5

$d=\text{diff}(p)/3;$

$m=4*n-3;$ %迭代公式

$q=p(1:n-1,:);$

$p(5:4:m,:)=p(2:n,:);$

$p(2:4:m,:)=q+d;$

$p(3:4:m,:)=q+2*d+d*A';$

$p(4:4:m,:)=q+2*d;$

$n=m;$

end

plot(p(:,1),p(:,2))

axis([-10 10 -10 10])

参考文献

- [1] 司守奎, 孙玺菁; 数学建模算法与应用, 北京: 国防工业出版社, 2011 年 8 月.
- [2] 葛哲学, 孙志强; 神经网络理论与 Matlab R2007, 北京: 电子工业出版社, 2007.9.
- [3] 周开利, 康耀红; 神经网络模型及其 Matlab 仿真程序设计, 北京: 清华大学出版社, 2005.7.
- [4] 王文波; 数学建模及其基础知识详解, 武汉: 武汉大学出版社, 2006.5.
- [5] 汪晓银, 邹庭荣等, 数学软件与数学实验 (第二版), 北京: 科学出版社, 2012.12.
- [6] S.J.Chapman, Matlab 编程, 北京: 科学出版社, 2006.9
- [7] 张德丰,《Matlab/Simulink 建模与仿真实例精讲》北京: 机械工业出版社, 2010.1
- [8] 薛定宇, 陈阳泉;《高等应用数学问题的 Matlab 求解》(第三版), 北京: 清华大学出版社, 2013.10
- [9] 部分内容摘自网络