第 3 章　MATLAB 符号运算

符号计算是指利用数学定理和恒等式，通过推理和演绎，分析化简表达式，将复杂表达式变为形式简单的恒等表达式。利用符号运算可以避免计算过程中产生误差。MATLAB 中的符号计算功能是由 Maple 独立引擎提供的，利用这个内置的 Maple 符号计算引擎，可以进行各种针对符号对象或解析式的数学运算，如微积分运算，代数、微分方程求解，线性代数和矩阵运算，以及 Laplace 变换、Fourier 变换和 Z 变换。

3.1　符号运算入门

符号运算与数值计算相同，都是科学研究中的重要内容。运用符号运算，可以轻松解决许多公式和关系式的推导问题。

3.1.1　符号对象的创建

MATLAB 提供了两个建立符号对象的函数：sym 和 syms，这两个函数的用法不同。

1．sym 函数

sym 函数用来创建单个符号量，调用格式为

```
符号量名=sym('符号字符串')
```

该函数可以建立一个符号量，符号字符串可以是常量、变量、函数或表达式。

【例 3-1】 利用 sym 函数创建符号变量，完成对方程组求解。

```
>> a=sym('a');
>> b=sym('b');
>> x=sym('x');
>> y=sym('y');
>> [x,y]=solve('a*x-b*y=1','a*x+b*y=3','x','y')
 x =
 2/a
 y =
 1/b
 >>
```

【例 3-2】 创建符号变量，求复数表达式 z=x+i*y 的共轭复数。

```
>> x=sym('x','real');
>> y=x+i*y;
>> x=sym('x','real');
>> y=sym('y','real');
>> z=x+i*y;
```

```
>> conj(z)
   ans =
x - y*i
```

2．syms 函数

syms 函数可以在一条语句中定义多个符号变量，调用格式为

```
syms  符号变量名 1 符号变量名 2 … 符号变量名 n
```

用这种格式定义符号变量时不要在变量名上加字符串分界符“‘’”，变量间用空格而不要用逗号分隔。在数学表达式中，一般习惯于使用排在字母表中前面的字母作为变量的系数，而用排在后面的字母表示变量。

例如，$f=ax^2+bx+c$，表达式中的 a、b、c 通常被认为是常数，用作变量的系数；而将 x 看作自变量。若在 MATLAB 中表示上述表达式，首先用 syms 函数定义 a、b、x 为符号对象。在进行导数运算时，由于没有指定符号变量，则系统采用数学习惯来确定表达式中的自变量，默认 a、b、c 为符号常数，x 为符号变量。即对函数 f 求导为 df/dx。

3.1.2 符号表达式的创建

含有符号对象的表达式被称为符号表达式，一个符号表达式应该由符号变量、函数、算术运算符组成，符号表达式的书写格式与数据表达式相同。

表 3-1 符号表达式和 MATLAB 表达式的对照

符号表达式	MATLAB 表达式
$y=\frac{1}{\sqrt{2x}}$	y=’1/sqrt(2*x)’
$\cos\left(x^2\right)-\sin\left(2x\right)$	‘cos(x^2)-sin(2*x)’
$\frac{e^{x^3}}{\sqrt{1-x}}$	‘exp(x^3)/sqrt(1-x)’
$\frac{1}{2x^n}$	‘1/(2*x^n)’

符号表达式有如下三种建立的办法。

（1）利用单引号来生成符号表达式。

（2）用 sym 函数建立符号表达式。

（3）使用已经定义的符号变量组成符号表达式。

将表达式中的自变量定义为符号变量后，赋值给符号函数名，即可生成符号函数。例如有一数学表达式，其用符号表达式生成符号函数 fxy 的过程为

```
fxy=(a*x^2+b*y^2)/c^2        %生成符号函数
syms a b c x y               %定义符号运算量
```

生成符号函数 fxy 后，即可用于微积分等符号计算。

【例 3-3】 符号函数 fxy=(a*x2+b*y2)/c2 ，分别求该函数对 x、y 的导数和对 x 的积分。

```
syms a b c x y               %定义符号变量
```

```
fxy=(a*x^2+b*y^2)/c^2;          %生成符号函数
diff(fxy,x)                     %符号函数 fxy 对 x 求导数
ans =2*a*x/c^2
diff(fxy, y)                    %符号函数 fxy 对 y 求导数
ans =2*b*y/c^2
int(fxy, x)                     %符号函数 fxy 对 x 求积分
ans =1/c^2* (1/3*a*x^3+b*y^2*x)
```

3.1.3 符号矩阵的相关操作

1．使用 sym 函数创建

sym 函数可以创建符号矩阵，用法如下。

```
A=sym('[ ]')
```

【例 3-4】 利用 sym 函数直接创建符号矩阵。

```
A = sym('[a , 2*b ; 3*a , 0]')
A =
    [ a, 2*b]
    [3*a, 0]
```

符号矩阵的每一行的两端都有方括号，这是与 MATLAB 数值矩阵的一个重要区别。

2．基于字符串创建

（1）用字符串直接创建矩阵。

（2）模仿 MATLAB 数值矩阵的创建方法。

需保证同一列中各元素字符串有相同的长度。

【例 3-5】 模仿数值矩阵的方式创建符号矩阵。

```
A =['[ a,2*b]'; '[3*a, 0]']
A =
    [ a, 2*b]
    [3*a, 0]
```

3．符号矩阵的修改

修改符号矩阵有两种方式，利用光标键移到指定位置直接修改或利用指令修改，本段主要介绍用指令修改。

指令修改运用了 subs 函数，使用形式如

```
A1=subs(A, 'new', 'old')
```

【例 3-6】 利用 subs 函数修改符号矩阵。

```
A =[ a, 2*b]
   [3*a, 0]
A = sym('[a , 2*b ; 3*a , 0]')
```

```
A1=sym(A,2,2,'4*b') %%等效于 A(2,2)='4*b';
A1 =[ a, 2*b]
    [3*a, 4*b]
A1=subs(A,'0','4*b')
A2=subs(A1, 'c', 'b')
A2 =[ a, 2*c]
    [3*a, 4*c]
```

3.1.4 符号运算中的运算符

MATLAB 中为符号运算提供了多种多样的运算符，如表 3-2 所示

表 3-2 符号运算中的运算符

符 号	符号用途说明
+	加
-	减
. *	点乘
*	矩阵相乘
^	矩阵求幂
.^	点幂
\	左除
/	右除
.\	点左除
./	点右除
kron	张量积
,	分隔符
;	（a）写在表达式后面时运算后不显示计算结果 （b）在创建矩阵的语句中指示一行元素的结束，如 m=[x y z;i j k]
:	创建向量的表达式分隔符，如 x=a:b:c a(:,j)表示 j 列的所有行元素；a(i,:)表示 i 行的所有列元素
[ ]	创建数组、向量、矩阵或字符串（字母型）
{ }	创建单元矩阵或结构
%	注释符，特别当编写自定义函数文件时，紧跟 function 后的注释语句， 在你使用 help 函数名时会显示出来
'	（a）定义字符串用 （b）向量或矩阵的共轭转置符
.'	一般转置符
...	表达式换行标记，表示表达式继续到下一行
=	赋值符号
==	等于关系运算符
< , >	小于，大于关系运算符
&	逻辑与
|	逻辑或
~	逻辑非
xor	逻辑异或

3.1.5 符号表达式中自变量的确定

MATLAB 中的符号可以表示符号变量和符号常量，findsym 可以帮助用户查找一个符号表达式中的符号变量，其调用方法如下。

- **findsym(expr)** 确定表达式 expr 中的所有符号为自变量。
- **findsym(expr,n)** 确定表达式 expr 中靠 x 最近的 n 个自变量。

【例 3-7】 利用 findsym 确定表达式中的自变量。

```
syms a x y z t
findsym(sin(pi*t))
findsym(x+i*y-j*z,1)
findsym(x+i*y-j*z,2)
findsym(x+i*y-j*z,3)
syms x a y z b;                %定义 5 个符号变量
s1=3*x+y;s2=a*y+b              %定义两个符号表达式
findsym(s1)
findsym(s2,2)
syms x y;
s=2*x+3*y;
findsym(s)
ans =
     x, y
syms a b x y;                  %定义符号变量
c=sym('3');                    %定义符号常量 c
findsym(a*x+b*y+c)
ans =                          %c 不在结果中出现
     a, b, x, y
```

MATLAB 按离字母 x 最近原则确定默认变量。

3.2 符号表达式运算

符号表达式可以进行多种运算，如基本的四则运算，也可进行表达式求值、数值转换及变量替换。

3.2.1 提取分子和分母

如果表达式是一个有理分式（两个多项式之比），或可以展开为有理分式（包括哪些分母为 1 的分式），可以利用 numden 将分子或分母提取出来。

【例 3-8】 利用 numden 提取分子分母。

```
 >> m= ' x^2 '
m=      x^2
```

```
 >> [n, d]=numden(m)
n=      x^2
d=    1
 >> f= ' a*x^2/(b-x) '
f=      a*x^2/(b-x)
 >> [n, d]=numden(f)
n=      a*x^2
d=      b-x
 %前二个表达式得到期望结果。
 >> g= ' 3/2*x^2+2/3*x-3/5 '
g=      3/2*x^2+2/3*x-3/5
 >> [n, d]=numden(g)
n=      45*x^2+20*x-18
d=      30
 >> h= ' (x^2+3)/(2*x-1)+3*x/(x-1) '
h=      (x^2+3)/(2*x-1)+3*x/(x-1)
 >> [n, d]=numden(h)
n=      x^3+5*x^2-3
d=      (2*x-1) * (x-1)
```

%在提取各部分之前，这两个表达式 g 和 h 被有理化，并变换成具有分子和分母的一个简单表达式。

```
>> k=sym( ' [3/2, (2*x+1)/3; 4/x^2, 3*x+4] ' )  %   try a symbolic array
k=
  [   3/2, (2*x+1)/3]
  [4/x^2,     3*x+4]
>> [n, d]=numden(k)
n=
  [3,  2*x+1]
  [4,  3*x+4]
d=
  [  2, 3]
  [x^2, 1]
```

这个表达式 k 是符号数组，numden 返回两个新数组 n 和 d，其中，n 是分子数组，d 是分母数组。如果采用 s=numden(f)形式，numden 仅把分子返回到变量 s 中。

numden 也可以化简分数表达式如下。

```
>> syms x y;
>> f=x/y+y/x;
>> [n,d]=numden(f)
 n =
    x^2 + y^2
 d =
    x*y
```

3.2.2 数值转换

1．数据类型转换函数

利用数据类型转换函数可以将常数转换为数值，常用的数据类型转换函数如表 3-3

所示。

表 3-3　数据类型转换函数

函　数　名	作　　用
logical	数值转化为逻辑值
char	转换为字符串数组
int8	转换为 8 字节整型数
uint8	转换为 8 字节数
int16	转换为 16 字节整型数
uint16	转换为 16 字节数
int32	转换为 32 字节整型数
uint32	转换为 32 字节数
int64	转换为 64 字节整型数
uint64	转换为 64 字节数
single	转换为单精度浮点数
double	转换为 64 字节浮点数
cell	转换为细胞数组
struct	转换为结构体类型

【例 3-9】 利用转换函数转换符号常量。

```
>> a=3.8495;
>> f=sym('6*a+2^(2*a)');
>> m=eval(f)
m =
    230.8895
>> int8(m)
ans =
    127
>> logical(m)
ans =
     1
```

2．sym2poly

sym2ploy 可以将符号表达式转换为数值多项式的系数向量，且系数从高到低依次排列。

【例 3-10】 使用 sym2poly 函数显示数值多项式的系数向量。

```
f=sym(7+3*x+5*x^2);
sym2poly(f)
ans=
     5   3   7
```

3．poly2sym

poly2sym 与 sym2poly 相反，可以将数值多项式的系数向量转换为符号表达式。

```
a=[5 3 7];
```

```
poly2sym(a)
ans=
    5*x^2+3*x+7
```

4．eval

MATLAB 中的 eval 函数可以计算符号表达式的具体值。

【例 3-11】 计算符号表达式 k*2+2^m 的值。

```
>> f=sym('k*2+2^m');
>> k=sym('5');
>> m=sym('7');
>> r=eval(f)
 r =
    138
```

3.2.3 变量替换

MATLAB 中提供了 subs 函数用于实现变量的替换，在处理复杂函数方程式时会使计算更简便。

- **Subs(S,old,new)** 用 new 替换 S 中的 old 变量，old 必须是 S 中的符号变量。
- **Subs(S,new)** 用 new 替换 S 中的自变量。

【例 3-12】 subs 函数用于实现变量的替换。

```
>> syms a m n w;
>> f=2+3^a;
>> subs(f,'a','m^2+5*n+w')
ans =
     3^(m^2 + 5*n + w) + 2
```

3.2.4 化简与格式化

MATLAB 提供了多种函数来实现对符号运算表达式进行化简，如 factor（因式分解），collect（合并同类项）、horner（将多项式分解为嵌套形式）、expand（展开表达式为多项式、指数函数、对数函数、三角函数）、simplify（化简一个表达式）、simple（将表达式化到最简形式）。

1．因式分解 factor

- **factor(x)** 若 x 可分解时，返回分解后的表达式，否则，返回原 x。

【例 3-13】 利用 factor 分解表达式 x^2+4*x+5。

```
>> f=sym('x^2+4*x+5');
>> factor(f)
ans =
      x^2 + 4*x + 5
      syms a b x y;
      A=a^3-b^3;
      factor(A)
```

```
ans =
    (a-b) * (a^2+a*b+b^2)
>>syms x; f=x^6+1;
>>factor(f)
  % factor 也可用于正整数的分解
 >>s=factor(100)
 >>factor(sym('123456789012345678901'))
```

大整数的分解要转化成符号常量，否则，表达精度不够会出错。

2．合并同类项 collect

❑ **collect(S)** 将 S 中相同次幂的项合并，S 可以是表达式也可以是符号矩阵。

❑ **collect(S,v)** 将 S 中 v 的相同幂次的项进行合并。

【例 3-14】 使用 collect 函数实现合并同类项。

（1）使用 collect 函数的第一种形式合并同类项。

```
>> f=sym('(x^2+2*x) * (x+2)');
>> collect(f)
 ans =
      x^3 + 4*x^2 + 4*x
```

（2）使用 collect 函数的第二种形式合并同类项。

```
>> syms x y;
>> f=(y^3+2*x) * (x+5);
>> collect (f,x)
 ans =
      2*x^2 + (y^3 + 10) *x + 5*y^3
>> syms x y;
>> f= x^2*y + y*x - x^2 + 2*x ;
>> collect(f)
>> collect(f,y)
 ans =
      (y - 1) *x^2 + (y + 2) *x
 ans =
      (x^2 + x) *y + 2*x - x^2
```

3．多项式分解 horner

用法：horner(S)，S 是符号多项式矩阵，horner 函数可以将每个多项式转换成嵌套形式。

【例 3-15】 分解多项式 f=5*x^4+3*x^2-x。

```
>> syms x
>> f=5*x^4+3*x^2-x;
>> horner(f)
ans =

      x* (x* (5*x^2 + 3) - 1)
>> syms x;
>> f=x^4+2*x^3+4*x^2+x+1;
>> g=horner(f)
```

```
    g =
       x* (x* (x* (x + 2) + 4) + 1) + 1
```

4．展开表达式 expand

用法：expand(S)，若 S 是多项式，则展开为相应的形式；若 S 是三角函数、指数函数和对数函数，则根据要求展开成相应形式。

【例 3-16】 用 expand 函数展开多项式。

```
>> syms x,y;
>> f=(5*x+4*y+3)^2;
>> expand(f)
    ans =
        25*x^2 + 40*x*y + 30*x + 16*y^2 + 24*y + 9
    s=(-7*x^2-8*y^2) * (-x^2+3*y^2)
    expand(s)                    %对 s 展开
    ans =
        7*x^4-13*x^2*y^2-24*y^4
%多项式展开
>> syms x;
>> f=(x+1)^6;
>> expand(f)
    ans =
       x^6 + 6*x^5 + 15*x^4 + 20*x^3 + 15*x^2 + 6*x + 1
%三角函数展开
>> syms x y; f=sin(x+y);
   expand(f)
   ans =
       cos(x)*sin(y) + cos(y)*sin(x)
```

5．化简表达式 simplify

用法：simplify(S)，表达式 S 可以是多项式也可以是符号表达式矩阵。

【例 3-17】 用 simplify 函数化简多项式。

（1）化简 sin(x)^2+cos(x)^2+2∗Sin(x) ∗cos(x)。

```
>> f=sym('sin(x)^2+cos(x)^2+2*Sin(x)*cos(x)');
>> simplify(f)
    ans =
         2*Sin(x)*cos(x) + 1
```

（2）化简 log(2∗x/y)和(-a^2+1)/(1-a)。

```
>>syms x y a
>>s=log(2*x/y);
>>simplify(s)
   ans =
         log(2)+log(x/y)
>>s=(-a^2+1)/(1-a)
>>simplify(s)
   ans =
         a+1
>>syms x;
```

```
>>f=sin(x)^2 + cos(x)^2 ;
>>simplify(f)
 ans =
      1
>> syms c alpha beta;
>> f=exp(c*log(sqrt(alpha+beta)));
>> simplify(f)
 ans =
    (alpha + beta)^(c/2)
```

6．最简转化 simple

有如下两种用法：

- **[r,how]=simple(S)** 返回值 r 是最简形式的符号表达式，how 是描述简化过程的字符串。
- **r=simple(S)** simple 函数将显示表达式 S 所有的简化形式，并返回其中最短的一个。

【例 3-18】 用 simple 函数化简多项式。

（1）化简 $f(x)=\sqrt[3]{\frac{1}{x^3}+\frac{6}{x^2}+\frac{12}{x}+8}$ 。

```
>> syms x;
>> f=(1/x^3+6/x^2+12/x+8)^(1/3);
>> y1=simplify(f)
   y1 =
      ((2*x + 1)^3/x^3)^(1/3)
 %多次使用 simple 可以达到最简表达。
>> g1=simple(f)
   g1 =
       ((2*x + 1)^3/x^3)^(1/3)
 >> g2=simple(g1)
    g2 =
       ((2*x + 1)^3/x^3)^(1/3)
```

（2）化简 y=(2+x)/x。

```
>> s=sym('y=(2+x)/x');
>> [r,how]=simple(s)
    r =
       x + 2 = x*y
    how =
       simplify(100)
```

（3）化简 2*sin(x) *cos(x)。

```
>> y=sym('2*sin(x) *cos(x)');
>> simple(y)
    simplify:
    sin(2*x)
    radsimp:
    2*cos(x) *sin(x)
    simplify(100):
```

```
    sin(2*x)
    combine(sincos):
    sin(2*x)
    combine(sinhcosh):
    2*cos(x)*sin(x)
    combine(ln):
    2*cos(x)*sin(x)
    factor:
    2*cos(x)*sin(x)
    expand:
    2*cos(x)*sin(x)
    combine:
    2*cos(x)*sin(x)
    rewrite(exp):
    2*((1/exp(x*i))/2 + exp(x*i)/2)*(((1/exp(x*i))*i)/2 - (exp(x*i)*i)/2)
    rewrite(sincos):
    2*cos(x)*sin(x)
    rewrite(sinhcosh):
    2*cosh(-x*i)*sinh(-x*i)*i
    rewrite(tan):
    -(4*tan(x/2)*(tan(x/2)^2 - 1))/(tan(x/2)^2 + 1)^2
    mwcos2sin:
    -2*sin(x)*(2*sin(x/2)^2 - 1)
    collect(x):
    2*cos(x)*sin(x)
    ans =
         sin(2*x)
```

3.2.5 数值表达式和符号表达式的互相转换

利用函数 sym 可以将数值表达式变换为其符号表达式。

```
>>sym(1.5)
   ans =
        3/2
>> sym(4.45)
   ans =
       89/20
```

函数 numeric 或 eval 可以将符号表达式变换成数值表达式：

```
phi='(1+sqrt(5))/2'
phi =
     (1+sqrt(5))/2
     numeric(phi)
ans =
     1.6180
>> p='(1+2^3)/2'
   p =
      (1+2^3)/2
```

```
>> eval(p)
   ans =
        4.5000
```

3.2.6 反函数

在 MATLAB 中，可以使用 finverse 计算反函数。

- **g=finverse(f)** 返回符号函数 f 的反函数 g。其中，f 是一个符号函数表达式，其变量为 x。求得的反函数 g 是一个满足 g(f(x))=x 的符号函数。

```
>> syms x;
>> f=sym(2/sin(x));
>> finverse(f)
   ans =
        asin(2/x)
```

- **g=finverse(f,v)** 返回自变量 v 的符号函数 f 的反函数。求得的反函数 g 是一个满足 g(f(v))=v 的符号函数。当 f 包含不止一个符号变量时，往往调用这个格式。

当 finverse 求得的解不唯一时，matlab 会给出警告。

```
>> syms x;
>> f=sym(x^2+1);
>> finverse(f)
   ans =
        (-1+x)^(1/2)
```

3.2.7 表达式替换函数

（1）subs(s) 用赋值语句中给定值替换表达式中所有同名变量。

（2）subs (s,old,new) 用符号或数值变量 new 替换原来 s 中的符号变量 old。

常用格式如下。

- **subs(f)** 求符号表达式 f 的值。
- **subs(f,a)** 用 a 替换 f 中的默认变量 x，并求值。
- **subs(f,x,a)** 用 a 替换 f 中的指定变量 x，并求值。

【例 3-19】 表达式替换函数演示。

（1）将表达式 x^2+y^2 中 x 取值为 2。

```
syms x y;
f=x^2+y^2;
subs(f,x,2)
```

（2）同时对两个或多个变量取值求解。

```
syms x y ;
f=x^2+y^2;
subs(f,[x,y],[1,2])    %同时替换两个变量并求值
```

```
subs(f,[x,y],[a+b,a-b])  %方括号换成大花括号也可以
```

3.3 符号运算精度

MATLAB 提供了三种计算精度：浮点运算的数值算法，精确运算的符号算法和可控精度的算法。

1．浮点运算的数值算法

浮点运算的数值算法是运算速度最快的运算方法，由于在计算机中以二进制进行存储，计算时取近似值，不可避免地会产生误差。

【例 3-20】 浮点运算的数值算法样例。

```
>> sym a;
>> a=2/3+4/7
   a =
      1.2381
```

2．精确运算的符号算法

精确运算速度较慢，但精确。

【例 3-21】 精确运算的符号算法样例。

```
>> a=sym(2/3+4/7)
   a =
      26/21
```

3．可控精度的算法

可控精度的算法通过规定有效数字位数控制精度，位数不同精度也不同。

- **digits(n)** 规定参加运算有效数字的位数，MATLAB 默认值为 32。
- **vpa(s)** 在 digits(n)控制下计算指定精度的 s，如果 n 未指定则默认 32。

【例 3-22】 可控精度的算法样例。

```
>> syms a b c
>> a=1/3+5/7;
>> b=pi;
>> c=3.7878882;
>> d=sym(4/9);
>> f1=vpa(a+b)
    f1 =
        4.1892
>> f2=vpa(a+c)
    f2 =
        4.8355
 >> f3=vpa(a+d)
     f3 =
        1.4921
>> digits(20)
>> f4=vpa(a+b)
```

```
    f4 =
        4.1892117012088405659
>> f5=vpa(a+c)
    f5 =
        4.8355072476190477104
>> f6=vpa(a+d)
    f6 =
        1.4920634920634920635
```

3.4 符号矩阵的计算

在进行符号矩阵的计算时，很多方面在形式上和数值计算相同，不必再去重新学一套关于符号运算的新规则。这里介绍的符号矩阵运算在形式上与数值计算中的运算相似，容易掌握。

3.4.1 基本代数运算

在 MATLAB 中，符号对象的代数运算和双精度运算从形式上看是相同的，由于 MATLAB 中采用了符号的重载，用于双精度运算的运算符同样可以用于符号对象。

【例 3-23】 符号矩阵的加减运算输入指令如下。

```
>> syms a b c d;                %定义基本的符号变量
>> A=sym('[a b;c d]');          %定义符号矩阵
>> B=sym('[a 2*b;c+b d-2]');    %定义符号矩阵
>> A+B;                         %计算符号矩阵的加法
>> A-B;                         %计算符号矩阵的减法
>> A*B;                         %计算符号矩阵的乘法
>> A/B;                         %计算符号矩阵的除法
>> syms a b c d;                %定义基本的符号变量
```

输出结果如下：

```
ans =
      [   2*a,   3*b]
      [ 2*c+b, 2*d-2]
ans =
      [  0, -b]
      [ -b,  2]
ans =
      [a^2+b* (c+b), 2*a*b+b* (d-2)]
      [c*a+d* (c+b), 2*c*b+d* (d-2)]
ans =
      [ (a*d-2*a-c*b-b^2)/(-2*c*b-2*b^2+a*d-2*a),
      -a*b/(-2*c*b-2*b^2+a*d-2*a)]
      [ -(2*c+d*b)/(-2*c*b-2*b^2+a*d-2*a),
      (a*d-2*c*b)/(-2*c*b-2*b^2+a*d-2*a)]
```

【例 3-24】 计算符号矩阵的二次方、三次方运行输入指令如下。

```
>> A=sym('[7 4 2;1 5 6;3 0 8]');   %定义符号矩阵
>> A^2                             %计算符号矩阵的二次方
>>A^3                             %计算符号矩阵的三次方
```

输出结果如下。

```
A =
     [ 7, 4, 2]
     [ 1, 5, 6]
     [ 3, 0, 8]
ans =
     [ 59, 48, 54]
     [ 30, 29, 80]
     [ 45, 12, 70]
 ans =
     [ 623, 476, 838]
     [ 479, 265, 874]
     [ 537, 240, 722]
```

【例 3-25】 计算符号矩阵的 4 次幂，输入指令如下所示。

```
>> A=sym('[1 2 3;4 5 6;7 8 9]');          %定义符号矩阵
>>A^4
```

输出结果如下：

```
A =
   [ 1, 2, 3]
   [ 4, 5, 6]
   [ 7, 8, 9]
ans =
   [  7560,  9288, 11016]
   [ 17118, 21033, 24948]
   [ 26676, 32778, 38880]
```

【例 3-26】 计算符号矩阵的指数，输入指令如下所示。

```
>> A=sym('[1 2 3;4 5 6;7 8 9]');          %定义符号矩阵
>>B=exp(A)
```

输出结果如下：

```
A =
    [ 1, 2, 3]
     [ 4, 5, 6]
     [ 7, 8, 9]
B =
     [ exp(1), exp(2), exp(3)]
     [ exp(4), exp(5), exp(6)]
     [ exp(7), exp(8), exp(9)]
```

3.4.2 线性代数运算

符号对象的线性代数运算和双精度的线性代数运算相同，有以下几种函数指令，如表3-4 所示。

表 3-4 符号矩阵线性运算函数

函 数 名 称	功 能 介 绍
inv	矩阵求逆
det	计算行列式的值
diag	对角矩阵
triu	抽取矩阵的上三角部分
tril	抽取矩阵的下三角部分
rank	计算矩阵的秩
rref	返回矩阵的所见行阶梯矩阵
null	零空间的正交基
colspace	返回矩阵列空间的基
transpose	返回矩阵的转置
eig	特征值分解
jordan	约当标准型变换
svd	奇异值分解

函数的具体用法如下。

1．inv 函数

MATLAB 提供了 inv 函数指令用于计算符号矩阵的逆，其具体用法如下。

❑ **inv(A)** 计算符号矩阵的逆。

【例 3-27】 生成数值希尔伯特矩阵，计算其逆矩阵输入指令如下。

```
>> A=hilb(4);          %定义符号矩阵
>>A=sym(A);
>>inv(A)
```

输出结果如下。

```
A =
   [  1, 1/2, 1/3, 1/4]
   [ 1/2, 1/3, 1/4, 1/5]
   [ 1/3, 1/4, 1/5, 1/6]
   [ 1/4, 1/5, 1/6, 1/7]
ans =
     [   16,  -120,   240,  -140]
     [ -120,  1200, -2700,  1680]
     [  240, -2700,  6480, -4200]
     [ -140,  1680, -4200,  2800]
```

2．det 函数

MATLAB 提供了 det 函数指令用于计算符号矩阵的行列式，其具体用法如下。

❑ **det(A)** 计算矩阵 A 的行列式。

【例 3-28】 计算【例 3-5】中符号矩阵的行列式，输入指令如下。

```
>> A=hilb(4);          %定义符号矩阵
>>A=sym(A);
>>det(A)
```

输出结果如下。

```
A =
   [  1, 1/2, 1/3, 1/4]
   [ 1/2, 1/3, 1/4, 1/5]
   [ 1/3, 1/4, 1/5, 1/6]
   [ 1/4, 1/5, 1/6, 1/7]
ans =
    1/6048000
```

3．diag 函数

diag 函数指令用于实现对符号矩阵对角线元素的操作，其具体用法如下。

❑ **diag(v,k)** 若 v 是由 n 个元素构成的矢量，则结果是 n+abs(k)阶方阵，当 k=0 时，将矢量 v 至于主对角线上；当 k<0 时，将矢量 v 置于主对角线之下；当 k>0 时，将矢量 v 置于主对角线之下。

❑ **diag(v)** 与 k=0 相同，将矢量 v 置于主对角线上。

❑ **diag(A,k)** A 是矩阵，结果是由矩阵 A 的第 k 条对角线上的元素组成的列矢量。

❑ **diag(A)** A 是矩阵，是 diag(A,k)用法中 k=0 的情况，结果是由矩阵 A 的主对角线元素组成的列矢量。

【例 3-29】 如果 a 是由四个元素构成的矢量，利用 diag 函数指令求解符号矩阵的对角线，输入指令如下。

```
>>  syms a b c;          %定义符号矩阵
>>  A=[a b+1 c*3 4];
>>  diag(A,1)
```

输出结果如下。

```
ans =
     [ 0, a,     0,   0, 0]
     [ 0, 0, b + 1,   0, 0]
     [ 0, 0,     0, 3*c, 0]
     [ 0, 0,     0,   0, 4]
     [ 0, 0,     0,   0, 0]
```

【例 3-30】 利用 diag 函数指令将矢量 a 置于主对角线上，输入指令如下。

```
>> syms a b c;          %定义符号矩阵
>>A=[a b+1 c*3 4]
>>diag(A)
```

输出结果如下：

```
A =
    [ a, b + 1, 3*c, 4]
ans =
     [ a,     0,   0, 0]
     [ 0, b + 1,   0, 0]
     [ 0,     0, 3*c, 0]
     [ 0,     0,   0, 4]
```

【例 3-31】 假设 A 为四阶希尔伯特矩阵，利用 diag 函数指令求矩阵的对角线，输入指令如下。

```
>>A=hilb(4)
>>diag(A)
```

输出结果如下。

```
A =
    1.0000    0.5000    0.3333    0.2500
    0.5000    0.3333    0.2500    0.2000
    0.3333    0.2500    0.2000    0.1667
    0.2500    0.2000    0.1667    0.1429
ans =
    1.0000
    0.3333
    0.2000
    0.1429
```

【例 3-32】 A 是随机矩阵，分别找出由矩阵 A 的第 1、2、3、4 条对角线上的元素组成的列矢量，输入指令如下。

```
>> s=sym(4);          %定义符号矩阵
>> class(s);
>> A=rand(s);
>> diag(A,3);
>> diag(A,2);
>> diag(A,1);
>> diag(A,0)
```

输出结果如下。

```
A =
    0.8147    0.6324    0.9575    0.9572
    0.9058    0.0975    0.9649    0.4854
    0.1270    0.2785    0.1576    0.8003
    0.9134    0.5469    0.9706    0.1419
ans =
    0.9572
ans =
    0.9575
    0.4854
ans =
```

```
    0.6324
    0.9649
    0.8003
ans =
    0.8147
    0.0975
    0.1576
    0.1419
```

【例 3-33】 假设 A 是一个三阶魔方矩阵，利用 diag 函数指令找出矩阵 A 主对角线上元素的列矢量，输入指令如下。

```
>> s=sym(4);          %定义符号矩阵
>> class(s);
>> A=magic(s);
>> diag(A)
```

输出结果如下。

```
A =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
ans =
    16
    11
     6
     1
```

4．triu 函数

MATLAB 提供 triu 函数对符号矩阵的上三角部分进行操作，其具体用法如下。

- **triu(A)** 抽取矩阵 A 主对角线上的三角部分重新组成一个新矩阵，其他部分用 0 来填充。
- **triu(A,k)** 抽取矩阵 A 的第 k 条对角线上的部分重新组成一个新矩阵，其他部分用 0 来填充。当 k>0 时，抽取的元素是在主对角线上且在 k 条对角线上的元素，其他部分用 0 来填充。当 k<0 时，抽取的元素是在主对角线下且在 k 条对角线上的元素，其他部分用 0 来填充。当 k=0 时，即 triu(A,0),与 triu(A)相同，抽取主对角线上的部分。

【例 3-34】 假设 A 是一个五阶魔方矩阵，利用 triu 函数指令生成一个由矩阵 A 主对角线上的元素组成的矩阵，输入指令如下。

```
>> s=sym(5);          %定义符号矩阵
>> class(s);
>> A=magic(s);
>> triu(A)
```

输出结果如下。

```
A =
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
ans =
    17    24     1     8    15
     0     5     7    14    16
     0     0    13    20    22
     0     0     0    21     3
     0     0     0     0     9
```

【例 3-35】 假设 A 是一个五阶魔方矩阵，利用 triu 函数指令生成由矩阵 A 主对角线上的元素组成的矩阵，以及由矩阵 A 主对角线下的元素组成的矩阵和由矩阵 A 主对角线的元素组成的矩阵，输入指令如下。

```
>> s=sym(5);          %定义符号矩阵
>> class(s);
>> A=magic(s);
>> triu(A,2);
>> triu(A,-2);
>> triu(A,0);
```

输出结果如下：

```
A =
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
ans =
     0     0     1     8    15
     0     0     0    14    16
     0     0     0     0    22
     0     0     0     0     0
     0     0     0     0     0
ans =
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
     0    12    19    21     3
     0     0    25     2     9
ans =
    17    24     1     8    15
     0     5     7    14    16
     0     0    13    20    22
     0     0     0    21     3
     0     0     0     0     9
```

【例 3-36】 假设 A 是一个三阶随机矩阵，利用 triu 函数指令生成由矩阵 A 主对角线上的元素组成的矩阵，以及由矩阵 A 主对角线下的元素组成的矩阵和由矩阵 A 主对角线的元素组成的矩阵，输入指令如下。

```
>> s=sym(3);          %定义符号矩阵
>> class(s);
>> A=rand(s);
>> triu(A,1);
>> triu(A,-1);
>> triu(A,0);
```

输出结果如下。

```
A =
   0.4218    0.9595    0.8491
   0.9157    0.6557    0.9340
   0.7922    0.0357    0.6787
ans =
         0    0.9595    0.8491
         0         0    0.9340
         0         0         0
ans =
   0.4218    0.9595    0.8491
   0.9157    0.6557    0.9340
         0    0.0357    0.6787
ans =
   0.4218    0.9595    0.8491
         0    0.6557    0.9340
         0         0    0.6787
```

【例 3-37】 利用 triu 函数生成由符号矩阵 A 主对角线上的元素组成的矩阵，以及由符号矩阵 A 主对角线下的元素组成的矩阵和由符号矩阵 A 主对角线的元素组成的矩阵，对比它们的不同，输入指令如下。

```
>> syms a b c;          %定义符号矩阵
>> A=[a^2 b+c 6 exp(c);a+b b a 5;4 b c 1;a^b c a 8]
>> triu(A);
>> triu(A,1);
>> triu(A,-1)
```

输出结果如下。

```
A =
   [   a^2, b + c, 6,exp(c)]
   [ a +b,      b, a,     5]
   [     4,     b, c,     1]
   [   a^b,     c, a,     8]
ans =
   [   a^2, b + c, 6,exp(c)]
   [     0,     b, a,     5]
   [     0,     0, c,     1]
```

```
        [ 0,    0, 0,     8]
  ans =
        [ 0, b +c, 6,exp(c)]
        [ 0,    0, a,     5]
        [ 0,    0, 0,     1]
        [ 0,    0, 0,     0]
  ans =
        [   a^2, b + c, 6,exp(c)]
        [ a +b,     b, a,     5]
        [    0,     b, c,     1]
        [    0,     0, a,     8]
```

5．tril 函数

MATLAB 提供了 tril 函数生成一个新矩阵，该新矩阵式抽取原矩阵的下三角部分，其他部分用 0 来填充，其具体用法如下。

- **tril(A)** 抽取矩阵 A 的主对角线下的三角部分重新组成一个新矩阵，其他部分用 0 来填充。
- **tril(A,k)** 抽取矩阵 A 的第 k 条对角线下的部分重新组成一个新矩阵，其他部分用 0 来填充。当 k>0 时，抽取的元素是在主对角线上且 k 条对角线下的元素，其他部分用 0 来填充。当 k<0 时，抽取的元素是在主对角线下且 k 条对角线下的元素，其他部分用 0 来填充。当 k=0 时，即 tril(A,0),与 tril 相同，抽取主对角线下的部分。

【例 3-38】 利用 tril 函数生成由矩阵 A 主对角线下的元素所组成的矩阵，输入指令如下。

```
>> A=magic(4)
>> B=sym(A)
>> tril(B)
```

输出结果如下。

```
A =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
B =
     [ 16,  2,  3, 13]
     [  5, 11, 10,  8]
     [  9,  7,  6, 12]
     [  4, 14, 15,  1]
ans =
     [ 16,  0,  0, 0]
     [  5, 11,  0, 0]
     [  9,  7,  6, 0]
     [  4, 14, 15, 1]
```

【例 3-39】 利用 tril 函数生成由符号矩阵 A 主对角线上的元素组成的矩阵，以及由符号矩阵 A 主对角线下的元素，组成的矩阵 A 主对角线的元素组成的矩阵，输入指令如下。

```
>> syms a b c;
>> A=[a^b c a 7;a+b b c 3;a^3 b+a 6 exp(c);2 a c 5]
>> B=sym(A)
>> tril(B,1)
>> tril(B,2)
>> tril(B,-1)
>> tril(B,-2)
>> tril(B,0)
```

输出结果如下。

```
A =
      [   a^b,      c, a,      7]
      [ a + b,      b, c,      3]
      [   a^3, a + b, 6,exp(c)]
      [     2,      a, c,      5]
B =
      [   a^b,      c, a,      7]
      [ a + b,      b, c,      3]
      [   a^3, a + b, 6,exp(c)]
      [     2,      a, c,      5]
 ans =
      [   a^b,      c, 0,      0]
      [ a + b,      b, c,      0]
      [   a^3, a + b, 6,exp(c)]
      [     2,      a, c,      5]
 ans =
      [   a^b,      c, a,      0]
      [ a + b,      b, c,      3]
      [   a^3, a + b, 6,exp(c)]
      [     2,      a, c,      5]
 ans =
      [     0,      0, 0,      0]
      [ a + b,      0, 0,      0]
      [   a^3, a + b, 0,      0]
      [     2,      a, c,      0]
  ans =
      [     0,      0, 0,      0]
      [     0,      0, 0,      0]
      [   a^3,      0, 0,      0]
      [     2,      a, 0,      0]
 ans =
      [   a^b,      0, 0,      0]
      [ a + b,      b, 0,      0]
      [   a^3, a + b, 6,      0]
      [     2,      a, c,      5]
```

6．rank 函数

在线性代数中，一个矩阵 A 的列秩是 A 的线性无关的纵列的极大数目。类似地，行秩是 A 的线性无关的横行的极大数目。矩阵的列秩和行秩总是相等的，因此，它们可以简单

地被称作矩阵 A 的秩。通常表示为 r(A)、rk(A)或 rank A。

在 MATLAB 中提供了 rank 函数指令用来计算符号矩阵的秩，其具体用法如下。

❑ **rank(A)** 返回矩阵 A 的秩。

【例 3-40】 利用 rank 指令计算四阶希尔伯特矩阵的秩。输入指令如下。

```
>> A=hilb(4)
>> A=sym(A)
>> rank(A)
```

输出结果如下。

```
A =
    1.0000    0.5000    0.3333    0.2500
    0.5000    0.3333    0.2500    0.2000
    0.3333    0.2500    0.2000    0.1667
    0.2500    0.2000    0.1667    0.1429
A =
   [   1, 1/2, 1/3, 1/4]
   [ 1/2, 1/3, 1/4, 1/5]
   [ 1/3, 1/4, 1/5, 1/6]
   [ 1/4, 1/5, 1/6, 1/7]
ans =
     4
```

【例 3-41】 利用 rank 指令计算符号矩阵的秩，输入指令如下。

```
>> syms a b c;
>> A=[b*2 c a 2;c b a 3;c^2 7 b 1;b*3 a c 8]
>> rank(A)
```

输出结果如下。

```
A =
   [ 2*b, c, a, 2]
   [   c, b, a, 3]
   [ c^2, 7, b, 1]
   [ 3*b, a, c, 8]
ans =
     4
```

7．rref 函数

矩阵的简化行阶梯式是高斯-约旦消元法解线性方程组的结果，其形式如下。

$$\begin{pmatrix} 1 & \cdots & 0 & * \\ \vdots & \ddots & \vdots & * \\ 0 & \cdots & 1 & * \end{pmatrix}$$

MATLAB 提供了 rref 函数返回符号矩阵的简化行阶梯矩阵，其具体用法如下。

❑ **R=rref(A)** 在计算过程中利用高斯-约当消元法和行主元素法，返回矩阵的简化行阶梯矩阵 R。

❑ **[R,jb]=rref(A)** 返回矩阵的简化行阶梯矩阵 R 和矢量 jb。R(l:r,jb)为 r×r 阶不确矩

阵，矩阵 A 的秩为 r=length(jb)，x(jb)为线性系统 Ax=b 的边界向量。

- **[R,jb]=rref(A,tol)** 返回矩阵的简化行阶梯 R 和矢量 jb 的要求与以上提到的相同，tol 指明了返回矩阵元素的误差。

【例 3-42】 使用 rref 函数返回符号矩阵 A 的简化行阶梯矩阵，输入指令如下。

```
>> syms a b c;
>> A=[b*2 c a 2;c b a 3;c^2 7 b 1;b*3 a c 8]
>> R=rref(A)
```

输出结果如下。

```
A =
    [ 2*b, c, a, 2]
    [   c, b, a, 3]
    [ c^2, 7, b, 1]
    [ 3*b, a, c, 8]
 R =
    [ 1, 0, 0, 0]
    [ 0, 1, 0, 0]
    [ 0, 0, 1, 0]
    [ 0, 0, 0, 1]
```

【例 3-43】 使用 rref 函数返回魔方矩阵的简化行阶梯矩阵，输入指令如下。

```
>> A=magic(4)
>> A=sym(A)
>> R=rref(A)
```

输出结果如下。

```
A =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
A =
    [ 16,  2,  3, 13]
    [  5, 11, 10,  8]
    [  9,  7,  6, 12]
    [  4, 14, 15,  1]
 R =
    [ 1, 0, 0,  1]
    [ 0, 1, 0,  3]
    [ 0, 0, 1, -3]
    [ 0, 0, 0,  0]
```

【例 3-44】使用 rref 函数返回三阶希尔伯特矩阵 A 的简化行阶梯矩阵，输入指令如下。

```
>> A=hilb(3)
>> A=sym(A)
>> R=rref(A)
```

输出结果如下。

```
A =
    1.0000    0.5000    0.3333
    0.5000    0.3333    0.2500
    0.3333    0.2500    0.2000
A =
   [   1, 1/2, 1/3]
   [ 1/2, 1/3, 1/4]
   [ 1/3, 1/4, 1/5]
 R =
   [ 1, 0, 0]
   [ 0, 1, 0]
   [ 0, 0, 1]
```

【例 3-45】 使用 rref 函数返回三阶随机矩阵 A 的简化行阶梯矩阵，输入指令如下。

```
>> A=rand(3)
>> A=sym(A)
>> R=rref(A)
```

输出结果如下。

```
A =
    0.8147    0.9134    0.2785
    0.9058    0.6324    0.5469
    0.1270    0.0975    0.9575
A =
   [7338378580900475/9007199254740992,8226958330713791/9007199254740992,
    627122237356493/2251799813685248]
   [8158648460577917/9007199254740992,1423946432832521/2251799813685248,
    153933462881711/281474976710656]
   [1143795557080799/9007199254740992, 109820732902227/1125899906842624,
    8624454854533211/9007199254740992]
 R =
    [ 1, 0, 0]
    [ 0, 1, 0]
    [ 0, 0, 1]
```

8．null 函数

与线性系统相联系的两个子空间是值域和零空间。如果 A 为 m×n 的矩阵，其秩为 r，那么，A 的向量空间就是由 A 的列划分的线性空间，这个空间的维数是 r，也就是 A 的秩。如果 r=n，则 A 的列线性无关。A 的零空间是由满足 Ax=0 的所有向量 x 组成的线性子空间。在 MATLAB 中可以用 null 函数求得零空间的正交基，其具体用法如下。

- **N=null(A)** 计算矩阵 A 的零空间的正交基，运算依赖矩阵 A 的奇异值分解。
- **N=null(A,'r')** 计算矩阵 A 的零空间的正交基，运算依赖矩阵 A 的简化行阶梯矩阵。

【例 3-46】 使用 null 函数返回符号矩阵 A 的零空间正交基，输入指令如下。

```
>> syms a b c;
>> A=[b*2 c a 2;c b a 3;c^2 7 b 1;b*3 a c 8]
```

```
>> N=null(A)
```

输出结果如下。

```
A =
    [ 2*b, c, a, 2]
    [   c, b, a, 3]
    [ c^2, 7, b, 1]
    [ 3*b, a, c, 8]
 N =
     [ empty sym ]
```

【例 3-47】 使用 null 函数返回魔方矩阵 A 的零空间正交基，输入指令如下。

```
>> A=magic(4)
>> A=sym(A)
>> N=null(A)
```

输出结果如下。

```
A =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
A =
    [ 16,  2,  3, 13]
    [  5, 11, 10,  8]
    [  9,  7,  6, 12]
    [  4, 14, 15,  1]
N =
    -1
    -3
     3
     1
```

9．colspace 函数

MATLAB 提供了 colspace 函数计算矩阵空间的基，其具体用法如下。

❑ **C=colspace(A)** 返回符号矩阵 A 的列空间的基。

【例 3-48】 使用 colspace 函数计算符号矩阵 A 的列空间的基，输入指令如下。

```
>> A=rand(4)
>> A=sym(A)
>> C=colspace(A)
```

输出结果如下。

```
A =
    0.8147    0.6324    0.9575    0.9572
    0.9058    0.0975    0.9649    0.4854
    0.1270    0.2785    0.1576    0.8003
    0.9134    0.5469    0.9706    0.1419
```

```
A =
   [7338378580900475/9007199254740992, 1423946432832521/2251799813685248,
    8624454854533211/9007199254740992, 8621393422876569/9007199254740992]
   [8158648460577917/9007199254740992, 109820732902227/1125899906842624,
    8690943295155051/9007199254740992, 4371875181445801/9007199254740992]
   [1143795557080799/9007199254740992, 627122237356493/2251799813685248,
    354913107955861/2251799813685248, 1802071410739743/2251799813685248]
   [8226958330713791/9007199254740992,  153933462881711/281474976710656,
    2185580645132801/2251799813685248, 638999261770491/4503599627370496]
C =
    [ 1, 0, 0, 0]
    [ 0, 1, 0, 0]
    [ 0, 0, 1, 0]
    [ 0, 0, 0, 1]
```

【例 3-49】 使用 colspace 函数计算魔方矩阵 A 的列空间的基，输入指令如下。

```
>> A=magic(4)
>> A=sym(A)
>> C=colspace(A)
```

输出结果如下。

```
A =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
A =
    [ 16,  2,  3, 13]
    [  5, 11, 10,  8]
    [  9,  7,  6, 12]
    [  4, 14, 15,  1]
C =
    [ 1, 0,  0]
    [ 0, 1,  0]
    [ 0, 0,  1]
    [ 1, 3, -3]
```

10．transpose 函数

MATLAB 提供了 transpose 函数计算矩阵的转置，其具体用法如下。

❑ **T=transpose(A)** 返回符号矩阵 A 的转置。

【例 3-50】 利用 transpose 函数计算符号矩阵 A 的转置矩阵，输入指令如下。

```
>> syms a b c;
>> A=[ a^b c a 7;a+b b c 3;a^3 b+a 6 exp(c);2 a c 5]
>> T=transpose(A)
```

输出结果如下。

```
A =
    [   a^b,     c, a,      7]
```

```
    [ a + b,     b, c,       3]
    [   a^3, a + b, 6, exp(c)]
    [     2,     a, c,       5]
T =
    [ a^b, a + b,    a^3, 2]
    [   c,     b,  a + b, a]
    [   a,     c,      6, c]
    [   7,     3, exp(c), 5]
```

【例 3-51】 利用 transpose 函数计算符号矩阵 A 的共轭的转置矩阵。

这里需要说明的是 A 与 A'不同，A'代表矩阵 A 的共轭矩阵，因此，在求解转置矩阵过程中所求得也是不同的矩阵。

输入指令如下。

```
>> syms a b c;
>> A=[ a^b c a 7;a+b b c 3;a^3 b+a 6 exp(c);2 a c 5]
>> T=transpose(A')
```

输出结果如下。

```
A =
    [   a^b,     c, a,       7]
    [ a + b,     b, c,       3]
    [   a^3, a + b, 6, exp(c)]
    [     2,     a, c,       5]
 T =
    [          conj(a^b),           conj(c), conj(a),            7]
    [ conj(a) + conj(b),           conj(b), conj(c),            3]
    [          conj(a)^3, conj(a) + conj(b),       6, exp(conj(c))]
    [                  2,           conj(a), conj(c),            5]
```

其中，Conj 用于计算共轭。

11．eig 函数

MATLAB 提供了 eig 函数对符号矩阵进行特征值分解，即计算矩阵的特征值和特征向量，其具体用法如下。

- **E=eig(A)** 返回由方阵 A 的特征值组成的矩阵。
- **[V,D]=eig(A)** 返回方阵 A 的特征值矩阵 D 和特证矢量矩阵 V，其中，特征值矩阵 D 是由 A 的特征值为对角线组成的对角矩阵，V、D 和 A 之间满足:AV=VD。

【例 3-52】 利用 eig 函数计算 3 阶随机矩阵的特征值和特征向量，输入指令如下。

```
>> syms a b c;
>> A=[a 1;b d]
>> E=eig(A)
>> [V,D]=eig(A)
```

输出结果如下。

```
A =
  [ a, 1]
```

```
   [ b, d]
E =
   a/2 + d/2 - (a^2 - 2*a*d + d^2 + 4*b)^(1/2)/2
   a/2 + d/2 + (a^2 - 2*a*d + d^2 + 4*b)^(1/2)/2
V =
   [ (a/2 + d/2 - (a^2 - 2*a*d + d^2 + 4*b)^(1/2)/2)/b - d/b,
   (a/2 + d/2 + (a^2 - 2*a*d + d^2 + 4*b)^(1/2)/2)/b - d/b]
   [               1,                    1]
D =
   [ a/2 + d/2 - (a^2 - 2*a*d + d^2 + 4*b)^(1/2)/2,                          0]
   [ 0,                         a/2 + d/2 + (a^2 - 2*a*d + d^2 + 4*b)^(1/2)/2]
```

12．jordan 函数

MATLAB 提供了 jordan 函数将矩阵变换为约当标准型，计算约当标准型也就是找一个非奇异矩阵 V，是 J=V/A*V 最接近对角矩阵，其中，V 称为转换矩阵。利用矩阵分块可以简化很多有关矩阵的证明和计算，任何仿真都可以通过相似变换变为约当标准型。jordan 函数的具体用法如下。

- **J=jordan(A)** 返回矩阵 A 的约当标准型。
- **[V,J]=jordan(A)** 返回矩阵 A 的约当标准型并且给出变换矩阵 V，满足 J=V/AV。

【例 3-53】 利用 jordan 函数计算三阶魔方矩阵的特征值和特征向量，输入指令如下。

```
>> A=magic(3)
>> A=sym(A)
>> [V,J]=jordan(A)
```

l

输出结果如下。

```
 A =
     8     1     6
     3     5     7
     4     9     2
A =
     [ 8, 1, 6]
     [ 3, 5, 7]
     [ 4, 9, 2]
V =
     [ (2*6^(1/2))/5 - 7/5,   - (2*6^(1/2))/5 - 7/5, 1]
     [ 2/5 - (2*6^(1/2))/5,     (2*6^(1/2))/5 + 2/5, 1]
     [                   1,                        1, 1]
J =
     [ -2*6^(1/2),        0,  0]
     [         0, 2*6^(1/2),  0]
     [         0,        0, 15]
```

13．svd 函数

在 MATLAB 中利用 svd 函数来进行矩阵的奇异值分解，奇异值分解在矩阵分解中占有极其重要的地位。

- **[U,S,V] = svd (X)** 返回一个与 X 同大小的对角矩阵 S，两个酉矩阵 U 和 V，且满足= U*S*V'。若 A 为 m×n 阵，则 U 为 m×m 阵，V 为 n×n 阵。奇异值在 S 的对角线上，非负且按降序排列。
- **[U,S,V] = svd (X,0)** 得到一个“有效大小”的分解，只计算出矩阵 U 的前 n 列，矩阵 S 的大小为 n×n。

【例 3-54】 利用[U,S,V]=svd(A)函数对符号矩阵 A 进行奇异值的分析，输入指令如下。

```
>> A=[9 8 7;6 5 4;3 2 1]
>> digits(30)
>> [U,S,V]=svd(A)
```

输出结果如下。

```
A =
    9    8    7
    6    5    4
    3    2    1
U =
   -0.8263    0.3879    0.4082
   -0.5206   -0.2496   -0.8165
   -0.2148   -0.8872    0.4082
S =
   16.8481         0         0
         0    1.0684         0
         0         0    0.0000
V =
   -0.6651   -0.6253   -0.4082
   -0.5724    0.0757    0.8165
   -0.4797    0.7767   -0.4082
```

3.4.3 科学计算

极限、微分和积分是微积分学中的核心和基础，MATLAB 提供了强大的函数指令来对其进行计算，下面做以简单介绍。

1．符号极限的计算

极限是高等数学的出发点和基础，高等数学的许多内容部是建立在极限理论基础上的。在 MATLAB 中提供了 limit 函数指令来对极限进行运算，其用法如下。

- **limit(fx,a)** 当变量 x 趋近于常数 a 时，计算符号函数 f(x)的极限值。
- **limit(f,a)** 相当于变量 x 趋近于 a 时，计算符号函数 f(x)的极限值。在没有指定符号函数 f(x)的自变量时，使用此函数来计算符号函数的极限，系统按 findsym 函数指示的默认变量来确定符号函数 f(x)的变量。
- **limit(f)** 在没有指定变量的目标值时，系统默认变量趋近于 0 相当于变量 x 趋近于 0，计算符号函数 f(x)的极限值，系统按 findsym 函数指示的默认变量来确定符号函数 f(x)的变量。

❑ **limit(f,x,a,'right')和 llimitf,x,a,'left')** 由于求极限可以从两边趋近，对于某些从左面趋近和从右面 A 趋近得到的结果是不同的，针对这种情况，函数 limit 专门提供了本语句来计算函数的左极限和右极限，'right'表示符号函数 f(x)的右极限，即变星 x 从右边趋近于 a, 'left'表示符号函数 f(x)的左极限，即变量 x 从左边趋近于 a。

【例 3-55】 计算 $\lim_{x\to\infty}\frac{3x^2-2x-1}{2x^3-x^2+5}$，输入指令如下。

```
>> syms x;
>> f=(3*x*x-2*x-1)/(7*x*x*x-5*x*x+3);
>> limit(f,x,inf)
```

输出结果如下。

```
ans =
    3/7
```

【例 3-56】 计算 $\lim_{x\to1}(2x-1)$，输入指令如下。

```
>> syms x;
>> f=2*x-1;
>> limit(f,x,1)
```

输出结果如下。

```
ans =
    1
```

【例 3-57】 计算 $\lim_{x\to0}\left(\frac{a^x+b^x+c^x}{3}\right)^{\frac{1}{x}}$，输入指令如下。

```
>> syms a b c x;
>> f=((a^x+b^x+c^x)/3)^(1/x);
>> limit(f,a)
```

输出结果如下。

```
ans =
    (a^a + b^a + c^a)^(1/a)/3^(1/a)
```

【例 3-58】 计算 $\lim_{x\to2}\frac{x^2+5}{x-3}$，输入指令如下。

```
>> syms x;
>> f=(x^2+5)/(x-3);
>> limit(f,x,2)
```

输出结果如下。

```
ans =
     -9
```

【例 3-59】 计算 $f=\lim_{x\to 0}\frac{\tan x-\sin x}{x^3}$ 的极限，输入指令如下。

```
>> syms x;
>> f=(tan(x)-sin(x))/x^3;
>> limit(f,x,0,'left')
```

输出结果如下。

```
ans =
    1/2
```

【例 3-60】 计算【例 3-59】中函数的右极限，输入指令如下。

```
>> syms x;
>> f=(tan(x)-sin(x))/x^3;
>> limit(f,x,0,'right')
```

输出结果如下。

```
ans =
    1/2
```

【例 3-61】计算 $f=\frac{1}{x^3}$ 当 x 趋于 0 时的极限值，并分别求出函数的左极限和右极限，输入指令如下。

```
>> syms x;
>> f=1/x^3;
>> limit(f,x,0)
>> limit(f,x,0.'left')
>> limit(f,x,0,'right')
```

输出结果如下。

```
ans =
      NaN
ans =
     -Inf
ans =
     Inf
```

2．符号微分的计算

MATLAB 提供了 diff 函数指令计算符号表达式的微分，具体用法如下。

- **diff(s)** 没有指定变量和导数阶数，则系统按 findsym 函数指定的默认变量对符号表达式 s 求阶导数。
- **diff(s,'v')** 以 v 为自变量，对符号表达式 s 求一阶微分。
- **diff(s,n)** 按 findsym 函数指示的默认变量对符号表达式 s 求 n 阶微分，且 n 为正整数。
- **diff(s,'v',n)** 以 v 为自变量，对符号表达式 s 求 n 阶微分，n 为正整数。

【例 3-62】 计算 $2x^3+4x^2+\cos(x)+\sin2(x)$函数关于 x 的微分，输入指令如下。

```
>> f=sym('2*x^3+4*x^2+cos(x)+sin(x)^2');
>> df=diff(f)
```

输出结果如下。

```
df =
   8*x - sin(x) + 2*cos(x)*sin(x) + 6*x^2
```

【例 3-63】 计算 $f=x-\ln(1+x)$的二阶微分，输入指令如下。

```
>> f=sym('x=ln(1+x)');
>> df=diff(f,2)
```

输出结果如下。

```
df =
0 = -1/(x + 1)^2
```

【例 3-64】 计算 $f=\sqrt{3-x}+\arctan\frac{1}{x}$关于 x 的三阶微分，输入指令如下。

```
>> f=sym('sqar(3-x)+arctan(1/x)');
>> df=diff(f,3)
```

输出结果如下。

```
df =
 14/(x^6*(1/x^2 + 1)^2) - 6/(x^4*(1/x^2 + 1)) - (D@@3)(sqar)(3 - x) - 8/(x^8*
(1/x^2 + 1)^3)
```

【例 3-65】 计算 $f=3ax^2+5bx+6$ 关于 b 的微分，输入指令如下。

```
>> f=sym('3*a*x^2+5*b*x+6');
>> df=diff(f,'b')
```

输出结果如下。

```
df =
   5*x
```

【例 3-66】 计算 $f=10x$ 关于 b 的微分，输入指令如下。

```
>> syms x
>> f=10*x*exp(-x/2);
>> df=diff(f)
```

输出结果如下。

```
df =
   10/exp(x/2) - (5*x)/exp(x/2)
```

3．符号积分的计算

积分运算是微分运算的逆运算，MATLAB 提供了 int 函数指令计算符号表达式的积分。

该函数既可以计算定积分，也可以计算不定积分和广义积分，其具体用法如下。

- **int(s)** 没有指定积分变量和积分阶数，系统按 findsym 函数指示的默认变量对被积函数或符号表达式 s 求不定积分。
- **int(s,v)** 以 v 为自变量，计算被积函数或符号表达式 s 的不定积分。
- **int(s,v,a,b)** 计算表达式 s 的定积分，该函数是求在[a,b]区间上的定积分，a 和 b 分别是定积分的下限和上限。a 和 b 可以是两个具体的数，也可以是一个符号表达式或是无穷(inf)，当 a 和 b 有一个或两个是 inf 时，函数返回一个广义积分；当 a 和 b 中有一个符号表达式时，函数返回一个符号函数。系统按 findsym 函数指示的默认变量来确定表达式的变量，当表达式 s 是符号矩阵时，则对矩阵的各元素分别进行积分。
- **int(s,v,a,b)** 符号表达式采用符号标量 v 作为标量，求 v 从 a 变到 b 时，符号表达式 s 的定积分值，a 和 b 的规定同上。

【例 3-67】 计算 f=tan3(x)关于 x 的不定积分，输入指令如下。

```
>> f=sym('tan(x)^3');
>> int(f)
```

输出结果如下。

```
ans =
     log(cos(x)) - (cos(x)^2 - 1)/(2*cos(x)^2)
```

【例 3-68】 计算 f=x3+acos(x)+bsin2(x)关于 b 的不定积分，输入指令如下。

```
>> f=sym('x^3+a*cos(x)+b*sin(x)^2');
>> int(f,'b')
```

输出结果如下。

```
ans =
     b* (a*cos(x) + x^3) - b^2* (cos(x)^2/2 - 1/2)
```

【例 3-69】 计算定积分 $\int_{\frac{\pi}{4}}^{\frac{\pi}{3}} \frac{x}{\sin^2 x}\mathrm{d}x$，输入指令如下。

```
>> syms x;
>> f=x/sin(x)^2;
>> int(f,x,pi/4,pi/3)
```

输出结果如下。

```
ans =
     pi/4 + log(6^(1/2)/2) - (pi*3^(1/2))/9
```

【例 3-70】 计算定积分 x^2+16y^2，输入指令如下。

```
>> syms x y;
>> f=x^2+16*y^2;
>> int(f,x)
```

```
>> int(f,y)
>> a=5;b=10
>> int(f,x,a,b)
```

输出结果如下。

```
ans =
      x^3/3 + 16*x*y^2
ans =
      x^2*y + (16*y^3)/3
ans =
     80*y^2 + 875/3
```

4．级数求和的计算

MATLAB 提供了 symsum 函数指令用于计算符号表达式的和，其具体用法如下。

- **r=symsum(s)** 自变量是由 findsym 函数所确定的符号变量，默认自变量为 k，计算表达式 s 从 0 到 k-1 的和。
- **r=symsum(s,v)** 计算表达式 s 从 0 到 v-1 的和。
- **r=symsum(s,a,b)** 计算表达式 s 默认变量从 a 到 b 的和。
- **r=symsum(s,v,a,b)** 计算表达式 s 变量 v 从 a 到 b 的和。

【例 3-71】 利用 symsum 函数计算符号表达式的和，输入指令如下。

```
>>  syms x k;
>>  symsum(x+3*x^4)
>>  symsum(1/x^k,k,0,inf)
>>  symsum(1/x^k,k,1,inf)
```

输出结果如下。

```
ans =
     (3*x^5)/5 - (3*x^4)/2 + x^3 + x^2/2 - (3*x)/5
ans =
     x/(-1+x)
ans =
     x/(-1+x)
```

【例 3-72】 对∑(*a***n*^2+*b***n*)其中 *n* 从 1 变到 20，输入指令如下。

```
>> syms a b n;
>> f=a*n^2+b*n;
>> symsum(f,n,1,20)
```

输出结果如下。

```
ans =
    2870*a + 210*b
```

5．taylor 级数的计算

MATLAB 提供了 taylor 函数用来计算符号表达式的泰勒级数展开式，其具体用法如下。

- **r=taylor(f)** f是符号表达式，自变量是由 findsym 函数所确定的符号变量，该函数将返回 f 在变量等于 0 处进行 5 阶泰勒展开时的展开式。
- **r=taylor(f,n,v)** 符号表达式 f 以符号标量 v 作为自变量，返回 f 的 n-1 阶麦克劳林级数(即在 v=0 处进行泰勒展开)展开式。
- **r=taylor(f,n,v,a)** 返同符号表达式 f 在 v=a 处进行 n-1 阶泰勒展开的展开式。

【例 3-73】 使用 taylor 函数指令计算符号表达式的泰勒级数展开式，输入指令如下。

```
>> syms x;
>> f=f*x^3;
>> T=taylor(f)
```

输出结果如下。

```
T =
    x^4/3 + x^2
```

【例 3-74】 使用 taylor 函数指令计算符号表达式的泰勒级数展开式，输入指令如下。

```
>> syms x a;
>> f=x^a;
>> T=taylor(f,4,a)
```

输出结果如下。

```
T =
    (a^3*log(x)^3)/6 + (a^2*log(x)^2)/2 + a*log(x) + 1
```

3.5 符号表达式积分变换

积分变换是工程设计和计算常用的工具，常见的积分变换有傅里叶（Fourier）变换、拉普拉斯（Laplace）变化和 Z 变换。

3.5.1 傅里叶变换及其反变换

傅里叶变换是一种分析信号的方法，它可分析信号的成分，也可用这些成分合成信号。许多波形可作为信号的成分，如正弦波、方波、锯齿波等，傅里叶变换用正弦波作为信号的成分。

$f(x)$是关于 x 的函数，如果 x 满足狄里赫莱条件，具有有限个间断点；具有有限个极值点；绝对可积。则有如下公式成立：$F(\omega)=\int_{-\infty}^{+\infty} f(x)e^{-j\omega x}\mathrm{d}x$。

时域中的 $f(x)$ 和频域中的 $F(\omega)$ 的博里叶反变换存在如下关系：

$$f(x)=\frac{1}{\pi}\int_{-\infty}^{\infty} F(\omega)e^{-j\omega x}\mathrm{d}\omega$$

1．傅里叶变换

在 Matlab 中进行傅里叶变换的函数指令如下。

❑ **fourier(f)** 默认函数 f 的自变量是 x，对默认变量计算傅里叶变换时，并且默认输出结果 F 是变量 ω的函数，记为 $F(\omega)=\int_{-\infty}^{+\infty}f(x)e^{-j\omega x}\mathrm{d}x$。

❑ **fourier(f,v)** 在默认函数 f 的自变量是 u、制定参数是 v，求函数 f 的傅里叶变换，记为 $F(v)=\int_{-\infty}^{+\infty}f(x)e^{-jvx}\mathrm{d}x$。

❑ **fourier(f,u,v)** 制定函数 f 的自变量是 u，指定参数变为 v，对函数 f 进行傅里叶变换，记为 $F(v)=\int_{-\infty}^{+\infty}f(u)e^{-jvu}\mathrm{d}u$。

【例 3-75】 分别用默认 fourier 函数和指定参数 fourier(f,v)函数计算 f(x)的傅里叶变换，输入指令如下。

```
>> syms x w u v;
>> f=sin(x)-cos(x)+1;
>> fourier(f)
>> fourier(f,v)
```

输出结果如下。

```
ans =
      2*pi*dirac(w) - pi* (dirac(w - 1) + dirac(w + 1)) - pi* (dirac(w - 1)
     - dirac(w + 1)) *i
 ans =
      2*pi*dirac(v) - pi* (dirac(v - 1) + dirac(v + 1)) - pi* (dirac(v - 1)
       - dirac(v + 1)) *i
```

【例 3-76】使用 fourier(f,u,v)函数在指定自变量和变换参数的情况下计算 f(x)的傅里叶变换，输入指令如下。

```
>> syms t u v;
>> f=exp(-1/3* (t+u)^2);
>> fourier(f,t,v)
```

输出结果如下。

```
ans =
    (3^(1/2)*pi^(1/2))/exp((3* (- v + (2*u*i)/3)^2)/4 + u^2/3)
```

2．Fourier 反变换

❑ **ifourier(F)** 在系统默认自变量和变换参数的情况下，计算函数的傅里叶反变换，记为 $f(x)=\frac{1}{2\pi}\int_{-\infty}^{+\infty}F(\omega)e^{-j\omega x}\mathrm{d}\omega$。

❑ **ifourier(F,v)** 在系统默认自变量，并指定变换参数是 v 的情况下，计算函数的傅里叶反变换，记为 $f(v)=\frac{1}{2\pi}\int_{-\infty}^{+\infty}F(\omega)e^{-j\omega v}\mathrm{d}\omega$。

❑ **ifourier(F,w,v)** 在系统的自变量为 w，并制定变换参数是 v 的情况下，计算函数的傅里叶反变换，记为 $f(v)=\frac{1}{2\pi}\int_{-\infty}^{+\infty}F(w)e^{-j\omega v}\mathrm{d}w$。

【例 3-77】 使用函数 f= ifourier(F)在系统默认自变量和变换参数的情况下，计算函数的博里叶反变换，输入指令如下。

```
>> syms x w u v;
>> f=sin(x)-cos(x)+1;
>> ifourier(f)
```

输出结果如下。

```
ans =
    (2*pi*dirac(t) - pi* (dirac(t - 1) + dirac(t + 1)) + pi* (dirac(t - 1)
     - dirac(t + 1)) *i)/(2*pi)
```

【例 3-78】 使用 ifourier(f,v)函数在指定自变量和变换参数的情况下计算 f(x)的傅里叶反变换，输入指令如下。

```
>> syms x y;
>> f=exp(-(x+y)^2/3);
>> ifourier(f,v)
```

输出结果如下。

```
ans =
    3^(1/2)/(2*pi^(1/2) *exp((3* (v + (2*y*i)/3)^2)/4 + y^2/3))
```

3.5.2 拉普拉斯变换及其反变换

拉普拉斯变换是工程数学中常用的一种积分变换，又名拉氏转换。拉氏变换是一个线性变换，可将一个有引数实数 $t(t\geqslant 0)$的函数转换为一个引数为复数 s 的函数。

如果函数 $f(t)$ 在区间$[0,+\infty)$ 上有定义，并且积分 $\int_0^{+\infty} f(t)e^{-st}\mathrm{d}t$ 在 s 的某一区域内收敛，则由这个积分确定函数 $F(s)$，即 $F(s)=\int_0^{+\infty} f(t)e^{-st}\mathrm{d}t$，此式称为函数 $f(t)$ 的拉普拉斯变换式，记为 $L[f(t)]=F(s)$。

拉普拉斯反变换定义为：$F(t)=\frac{1}{2\pi i}\int_{c-i\infty}^{c+i\infty} L(s)e^{st}\mathrm{d}t$，其中 c 为使函数 $L(s)$的所有奇点位于直线 $s=c$ 左边的实数，Laplace 反变换记为：$F(t)=L^{-1}[L(s)]$。

1．拉普拉斯变换

在 MATLAB 中提供了如下函数来进行拉普拉斯变换。

- **laplace(F)** 在默认自变量(自变量默认为 x)和参变量(参变量默认为 s)的情况下，计算符号函数的 Laplace 变换，记为 $L(s)=\int_0^{+\infty} F(x)e^{-sx}\mathrm{d}x$。
- **laplace(F,z)** 在默认自变量(自变量默认为 x)，并指定参变量为 z 的情况下，计算函数的拉普拉斯变换，记为 $L(z)=\int_0^{+\infty} F(x)e^{-zx}\mathrm{d}x$。
- **laplace(F,w,z)** 在指定自变量为 w，并指定参变量为 z 的情况下，计算函数的拉普拉斯变换，记为：$L(z)=\int_0^{+\infty} F(w)e^{-zw}\mathrm{d}w$。

【例 3-79】 使用函数 laplace(F)在默认自变量和参变量的情况下，计算符号函数的拉普拉斯变换，输入指令如下。

```
>> syms x;
>> f=2*sin(3*x);
>> laplace(f)
```

输出结果如下。

```
ans =
    6/(s^2 + 9)
```

【例 3-80】 使用函数 laplace(F,w,z)和 f(F,z)分别计算在指定自变量为 w，并指定参变量为 z 的情况下，计算符号函数的拉普拉斯变换，输入指令如下。

```
>> syms w z x;
>> f=1+log(x+2);
>> l=laplace(f,w,z)
>> l=laplace(f,z)
```

输出结果如下。

```
l =
   (log(x + 2) + 1)/z
l =
   1/z + laplace(log(x + 2), x, z)
```

2．进行 laplace 函数范变化的指令如下

- **ilaplace(L)** 在默认自变量(自变量默认为 s)和参变量(参变量默认为 t)的情况下，计算函数 L(s)的拉普拉斯反变换，记为 $L^{-1}[L(s)]=F(t)=\frac{1}{2\pi i}\int_{c-i\infty}^{c+i\infty}L(s)e^{st}\mathrm{d}s$，其中，c 为使函数 L(s)的所有奇点位于直线 s=c 左边的实数。
- **ilaplace(L,v)** 在默认自变量(自变量默认为 s)并指定参变量 v 的情况下，计算函数 L(s)的拉普拉斯反变换，记为 $L^{-1}[L(s)]=F(v)=\frac{1}{2\pi i}\int_{c-i\infty}^{c+i\infty}L(s)e^{sv}\mathrm{d}s$，其中，c 为使用函数 L(s)的所有奇点位于直线 s=c 左边的实数。
- **ilaplace(L,v,x)** 在指定自变量为 x 并指定参变量为 v 的情况下，计算函数 L(s)的拉普拉斯反变换，记为 $L^{-1}[L(x)]=F(v)=\frac{1}{2\pi i}\int_{c-i\infty}^{c+i\infty}L(x)e^{sv}\mathrm{d}x$，其中 c 为使用函数 L(s)的所有奇点位于直线 s=c 左边的实数。

【例 3-81】 使用函数 ilaplace(L)在默认自变量和参变量的情况下，计算函数 L(s)的拉普拉斯反变换，输入指令如下。

```
>> syms x;
>> f=cos(x-2);
>> L=ilaplace(f)
```

输出结果如下。

```
L =
    ilaplace(cos(x - 2), x, t)
```

【例 3-82】 使用函数 ilaplace(L,v)和 ilaplace(L,v,x)计算函数的拉普拉斯反变换，输入指令如下。

```
>> syms v x w;
>> L=(v^3+w^2+3);
>> ilaplace(L,v)
```

输出结果如下。

```
ans =
    3*dirac(v) + v^3*dirac(v) + dirac(v, 2)
```

3.5.3 Z 变换及其反变换

Z 变换的算法：当 $t<0$ 时，$f^*(t)=f(t)=0$；当 $t\geqslant 0$ 时，$f^*(t)=\sum_{k=0}^{+\infty}f(kT)\sigma(t-kT)$。

对该式进行拉普拉斯变换，得到 $F^*(s)=\sum_{k=0}^{+\infty}f(kT)e^{-skT}$，此时令 $z=e^{sT}$，于是函数变为 $F^*(z)=\sum_{k=0}^{+\infty}f(kT)z^{-k}$，在函数 $F^*(z)$ 收敛的情况下，称 $F^*(z)$ 是 $f^*(t)$ 的 Z 变换，Z 变换记为 $F(z)=\sum_{k=0}^{\infty}f(n)z^{-n}$。

1．Z 变换

在 MATLAB 中进行 Z 变换的函数如下。

- **ztrans(f)** 在默认自变量(默认自变量为 n)和参变量(参变量默认为 z)的情况下，计算符号函数的 Z 变换，记为 $F(z)=\sum_{n=0}^{+\infty}f(n)z^{-n}$。
- **ztrans(f,v)** 在默认自变量(默认自变量为 n)并制定参变量为 v 的情况下，计算符号函数的 Z 变换，记为 $F(v)=\sum_{n=0}^{+\infty}f(n)v^{-n}$。
- **ztrans(f,k,v)** 在制定自变量为 k，并制定参变量为 v 的情况下，计算符号函数的 Z 变换，记为 $F(z)=\sum_{n=0}^{+\infty}f(k)v^{-k}$。

【例 3-83】 使用 ztrans(f)函数指令在默认自变量和参变量的情况下对函数进行 Z 变换，输入指令如下。

```
>> syms x;
>> f=cos(4*x);
>> ztrans(f)
```

输出结果如下。

```
ans =
    (z* (z - cos(4)))/(z^2 - 2*cos(4) *z + 1)
```

【例 3-84】 使用 ztrans(f,v)函数与 ztrans(f,k,v)分别对函数进行 Z 变化，输入指令如下。

```
>> syms k m v;
>> f=tan(3*k*m);
>> ztrans(f,v)
```

输出结果如下。

```
ans =
    ztrans(tan(3*k*m), m, v)
```

2．Z 变换的反变换

Z 反变换记为：$f(n)=Z^{-1}(F(z))$。

MATLAB 提供了 iztrans 函数指令来实现 Z 反变换，其调用方法如下。

- **iztrans(F)** 在默认自变量(默认自变量为 n)和参变量(参变量为 z)的情况下，对函数进行 Z 反变换，记为 $f(n)=\frac{1}{2\pi i}\int_{|z|=R}F(z)z^{n-1}\mathrm{d}z, n=1,2,3,...$
- **itrans(F,v)** 在默认自变量(默认自变量为 n)并指定参变量为 v 的情况下，对函数进行 Z 反变换，记为 $f(n)=\frac{1}{2\pi i}\int_{|v|=R}F(v)v^{n-1}\mathrm{d}v, v=1,2,3,...$
- **itrans(F,w,v)** 在指定自变量为 w 并指定参变量为 v 的情况下，对函数进行 Z 反变换，记为 $f(w)=\frac{1}{2\pi i}\int_{|v|=R}F(v)v^{w-1}\mathrm{d}v, v=1,2,3,...$

【例 3-85】使用 iztrans(F)函数指令在默认自变量和参变量的情况下对函数进行 Z 反变换，输入指令如下。

```
>> syms w,v;
>> f=w* (w-3) * (w/2+f*w-2);
>> iztrans(f,v)
```

输出结果如下。

```
ans =
    3*iztrans(x^2, x, n) - iztrans(x^3, x, n)
```

【例 3-86】 使用函数指令 iztrans(F,v)和 iztrans(F,w,v)分别计算函数的 Z 反变换，输入指令如下。

```
>> syms w,v;
>> f=w* (w-3) * (w/2+f*w-2);
>> iztrans(f,v)
```

输出结果如下。

```
ans =
     (w^3*kroneckerDelta(v, 0))/2 - (7*w^2*kroneckerDelta(v, 0))/2 +
     (3*w^2 - w^3) *iztrans(x^3, x, v) - (9*w^2 - 3*w^3) *iztrans(x^2, x, v)
      + 6*w*kroneckerDelta(v, 0)
```

3.6 符号函数的图形绘制

图形是解决数学问题的必要途径，MATLAB 除了为解决代数方程提供了支持外还对函数图像的绘制提供了强大的支持。本节对简单的符号函数绘制加以简单介绍，详细的图形绘制在第 4 章进行系统学习。

3.6.1 符号函数的曲线绘制

MATLAB 提供了 ezplotl 函数和 ezplot3 函数用于绘制符号函数的二维曲线和三维曲线。

1．二维曲线的绘制

MATLAB 提供了 ezplot 函数来绘制符号函数的二维曲线，此函数可以绘制显函数图形、隐函数图形和参数方程的图形，具体用法如下。

- **ezplot(f)** 绘制显函数 f 在区间[–2,2]的二维曲线；绘制参数方程 $x=x(t)$，$y=y(t)$在区间 $0<t<2$ 的曲线。
- **ezplot(f,[min,max])**，**ezplot(f,[xmin,xmax,ymin,ymax,])和 ezplot(x,y,[min,tmax])** 第一种用法是绘制显函数 f 在指定区间[min,max]的二维曲线；第二种用法是绘制隐函数 f 在指定区间 xmin<x<xmax、ymin<y<ymax 的曲线；第三种用法是绘制参数方程 x=x(t)、y=y(t)在区间 tmin<t<tmax 的曲线。

【例 3-87】 使用 ezplot(f)函数指令绘制显函数的二维曲线，输入指令如下。

```
>> syms x;
>>f=sin(x);
>> ezplot(f);
>> grid;
>> title('sin(x)');
```

输出图形如图 3-1 所示。

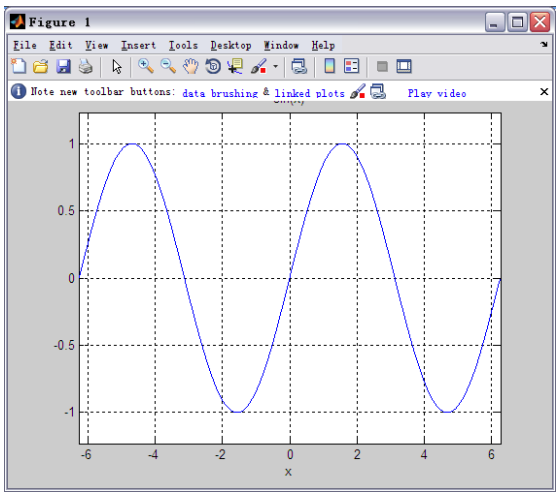

图 3-1 ezplot(f)函数二维曲线

【例 3-88】 绘制函数 $y=x^3-x^2-x+1$ 的二维曲线，输入指令如下。

```
>> syms x;
>>f=x^3-X^2-x+1;
>> ezplot(f);
>> grid;
>> title('exp');
```

输出图形如图 3-2 所示。

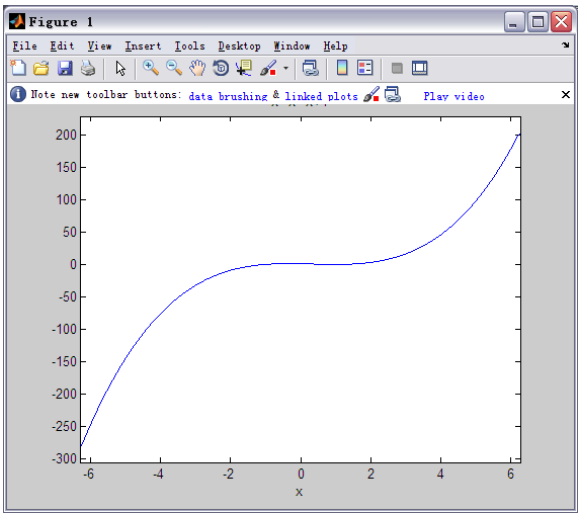

图 3-2 二维曲线

2．三维曲线的绘制

ezplot3 函数用于绘制符号函数的三维曲线，具体用法如下。

- **ezplot3(x,y,z)** 绘制参数方程 $x=x(t)$、$y=y(t)$、$z=z(t)$在默认区间 $0<t<2$ 的三维曲线。
- **ezplot3(x,y,z[t,min,tmaxJ)** 绘制参数方程 $x=x(t)$, $y=y(t)$, $z=z(t)$在区间 tmin<t<tmax 的三维曲线。ezplot3(...'animate'), 生成空间曲线的动态轨迹。

【例 3-89】 绘制函数的三维曲线，输入指令如下。

```
>> syms t;
>>x=cos(t);
>> y=tan(t);
>> z=t;
>> ezplot3(x,y,z)
```

输出图形如图 3-3 所示。

3.6.2 符号函数等值线的绘制

MATLAB 提供了 ezcontour 函数和 ezcontourfd 函数用于绘制符号函数的等值线，两个函数的使用方法类似，区别在于 ezcontourfd 函数绘制带有填充区域的等值线。ezcontour 函数的具体用法如下。

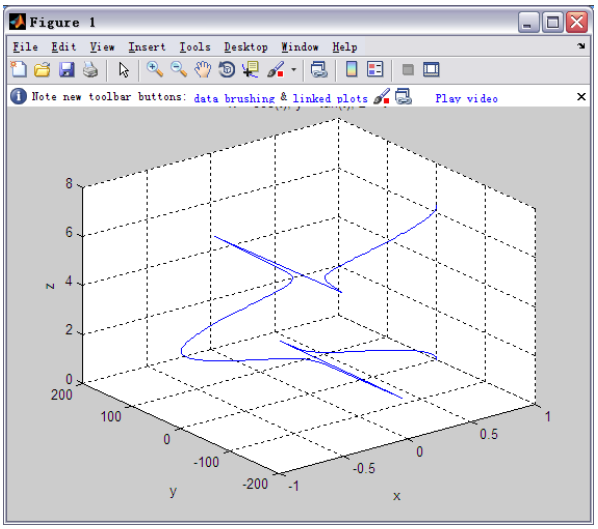

图 3-3　三维曲线

- ❑ **ezcontour(f)**　绘制二元函数 f(x,y)在默认区域的等值线。
- ❑ **ezcontour(f,domain)**　绘制二元函数 f(x,y)在指定区域的等值线。
- ❑ **ezcontour(...,n)**　绘制等值线图，并指定等值线的条数。

【例 3-90】　使用 ezcontour 函数绘制符号函数的等值线，输入指令如下。

```
>> syms x;
>>f=x^3-x^2-x+1;
>> ezcontour(f);
```

输出图形如图 3-4 所示。

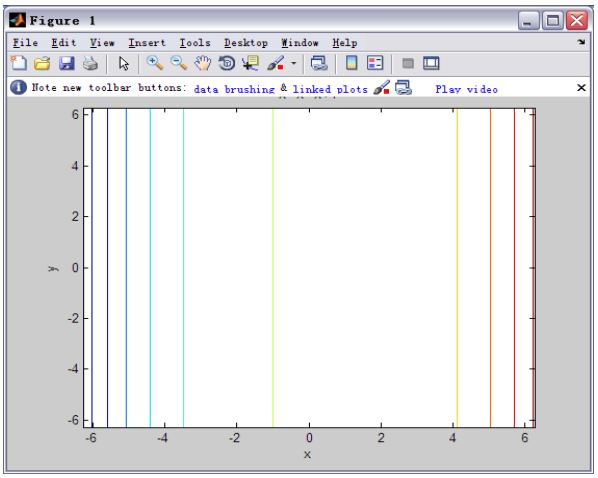

图 3-4　ezcontour 函数绘制的等值线

【例 3-91】　使用 ezcontourf 函数绘制符号函数的等值线，输入指令如下。

```
>> syms x y;
>>f=2* (1+y)^2*exp(-(x^3)-(x+1)^2);
>> ezcontourf(f);
```

输出图形如图 3-5 所示。

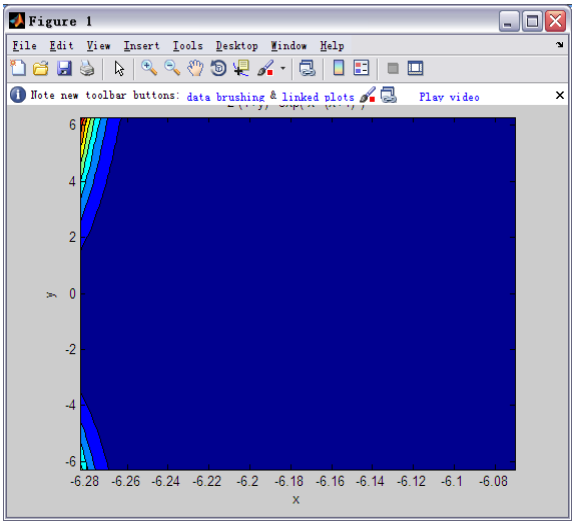

图 3-5　ezcontourf 函数绘制的等值线

3.6.3　符号函数曲面图及表面图的绘制

MATLAB 提供了 ezmesh 函数和 ezmeshc 函数用于绘制符号函数的三维曲面图，以及 ezsurf 函数和 ezsurfc 函数用于绘制符号函数的三维表面图。

ezmesh 函数和 ezmeshc 函数的区别在于 ezmeshc 函数在绘制三维曲面图的同时绘制等值线，ezsurf 函数和 ezsurfc 函数的区别在于 ezsurfc 函数在绘制三维表面图的同时绘制等值线。

1．ezmesh 函数和 ezsurf 函数

ezmesh 函数用于绘制三维曲面图，ezsurf 函数绘制三维表面图。ezmesh 的具体用法如下。

- **ezmesh(f)**　绘制 $f(x,y)$的图像。
- **ezmesh(f,domain)**　在指定区域绘制 $f(x,y)$的图像。
- **ezmesh(x,y,z)**　在默认区域绘制三维参数方程的图像。
- **ezmesh(x,y,z, [smin,smax,tmin,tmax])或 ezmesh(x,y,z,[min,max])**　在指定区域绘制维参数方程的图像。

【例 3-92】　利用 ezmesh 函数和 ezsurf 函数绘制三维曲面图和三维表面图，输入指令如下。

```
>> syms x y;
>>ezmesh(x*exp(x^3+y^2),[-2.5,2.5]);
>> ezsurf(x*exp(x^3+y^2),[-2.5,2.5]);
```

输出图形如图 3-6 所示。

2．ezmeshc 函数和 ezsurfc 函数

MATLAB 提供了 ezmeshc 函数用于绘制带等值线的三维曲面图，ezsurfc 函数绘制带等值线的三维表面图，两个函数使用的方法类似。ezmeshc 函数的具体用法如下。

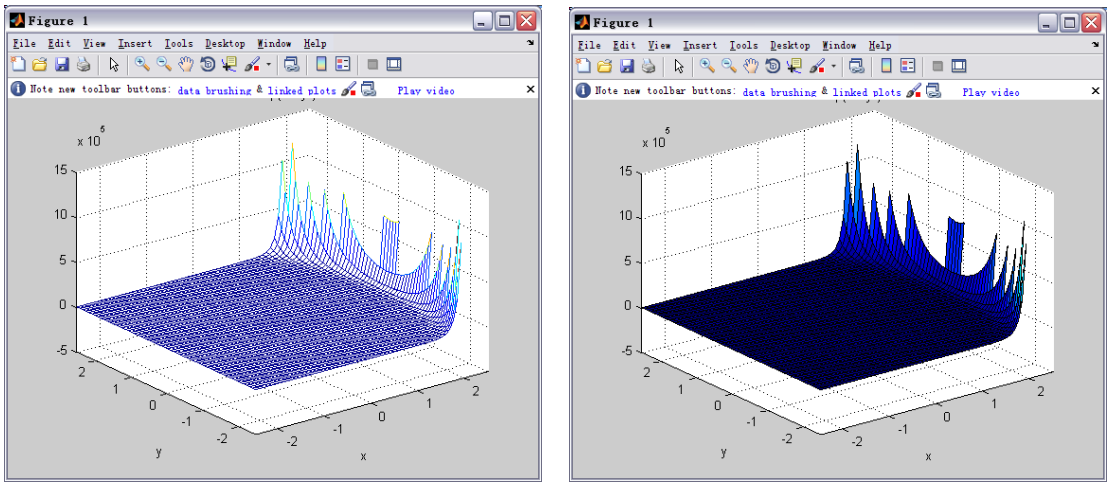

图 3-6　三维曲面图和三维表面图

- **ezmeshc(f)**　在默认区域-2π<x<2π，-2π<y<2π绘制二元函数 f(x,y)的图像。
- **ezmeshc(f,domain)**　在指定区域绘制二元函数 f(x,y)的图像。
- **ezmeshc(x,y,z)**　在默认区域-2π<s<2π，-2π<t<2π绘制三维参数方程 x=x(s,t), y=y(s,t),z=z(s,t)的图像。
- **ezmeshc(x,y,z,[smin,smax,tmin,tmax])或 ezmeshc(x,y,z,[min,max])**　在指定区域绘制三维参数方程的图像。

【例 3-93】使用 ezmeshc 函数和 ezsurfc 函数绘制带等值线的三维曲面图和三维表面图，输入指令如下。

```
>> syms x y;
>>ezmeshc(x*exp(x^3+y^2),[-2.5,2.5]);
>> syms x y;
>> ezsurfc(x*exp(x^3+y^2),[-2.5,2.5]);
```

输出图形如图 3-7 所示。

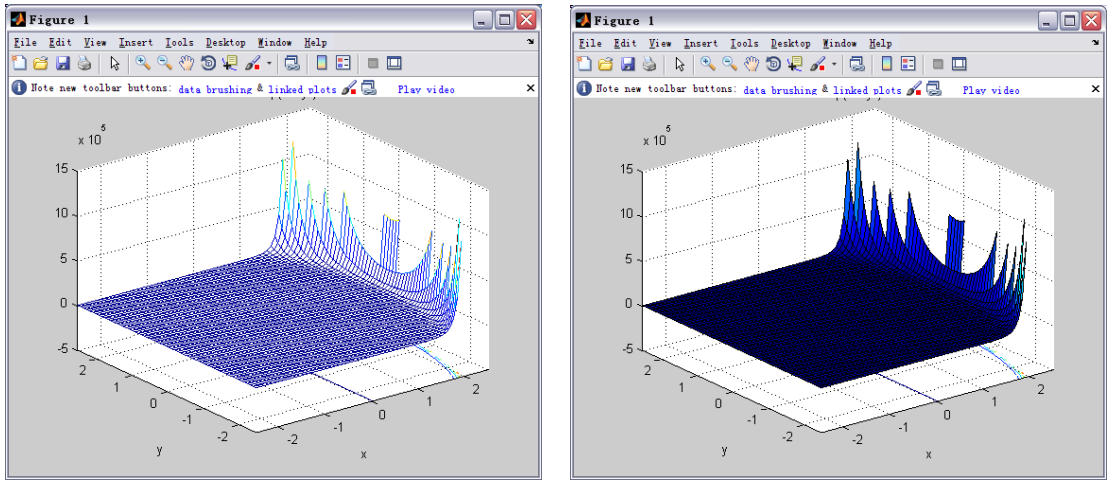

图 3-7　三维曲面图和三维表面图

3.7 符号方程的求解

在学习代数的过程中，我们一直在探索关于方程求解的各种理论与方法，从最基础的消元法到高斯迭代，方程成为我们在数学计算中必不可少的步骤，其重要性不言而喻。MATLAB 为方程的求解提供了更便利的方法。

3.7.1 代数方程的求解

代数方程分为线性方程、非线性方程以及超越方程。对于代数方程，MATLAB 提供了一种对方程的求解指令 solve，其格式如下。

- **g=solve(eq)** 其中 eq 可以是符号表达式或不带符号的字符串，该函数用于求解方程 eq=0，其自变量采用默认变量，也可以通过 findsym 函数来确定。
- **g=(eq,var)** 求解方程 eq=0，其自变量参数由 var 来指定，其中 eq 和上一种调用方式相同，返回值 g 是由方程所有解构成的列向量。

【例 3-94】 使用 solve 函数解代数方程 $2ax2+3bx-c=0$，输入指令如下。

```
>> syms x a b c;
>>x=solve('2*a*x^2+3*b*x-c');
```

输出结果如下。

```
x=
   -1/4* (3*b-(9*b^2+8*a*c)^(1/2))/a
   -1/4* (3*b-(9*b^2+8*a*c)^(1/2))/a
```

【例 3-95】 使用 solve 函数解代数方程 $4ax2+bx+c=0$，其中 b 为自变量，输入指令如下。

```
>> syms x a b c;
>>x=solve('4*a*x^2+b*x+c','b');
```

输出结果如下。

```
x=
   -(4*a*x^2+c)/x
```

solve 求解指令同样适用于求解代数方程组，指令格式如下。

- **g=(eq1,eq2,...eqn)** 求解由符号表达式或不带符号的字符串 eq1,eq2,...eqn 组成的方程组，其中，自变量为整个方程组的默认变量，即将函数 findsym 作用于整个方程组时返回的变量。
- **g=(eq1,eq2,...eqn,var1,var2...,varn)** 求解由符号表达或不带符号的字符串 eq1,eq2,...eqn 组成的方程组，其自变量由输入参数 var1,var2...,varn 指定。

【例 3-96】 使用 solve 函数解代数方程组 $\begin{cases} x+y=1 \\ x+3y=5 \end{cases}$，输入指令如下。

```
>> syms x y;
>>S=solve('x+y=1', 'x+3*y=5')
>> disp('S.x'),disp(S.x),disp('S.y'),disp(S.y)
```

输出结果如下。

```
S=
   x :  [1x1 sym]
   y :  [1x1 sym]
   S.x
   -1
   S.y
   2
```

【例 3-97】 使用 solve 函数解代数方程组 $\begin{cases} x+2y=3 \\ 3x+y=8 \end{cases}$，输入指令如下。

```
>> syms x y;
>>S=solve('x+2*y=3', '3*x+y=8')
>> disp('.x'),disp(S.x),disp('S.y'),disp(S.y)
```

输出结果如下。

```
S=
   x :  [1x1 sym]
   y :  [1x1 sym]
   S.x
   1/5
   S.y
   13/5
```

3.7.2 微分方程求解

从数值计算的角度来看，微分方程要比代数方程复杂困难得多，这时候不妨通过符号计算指令来进行求解。对于符号计算来说，无论是初值问题还是边值问题，其求解微分方程的指令行事都相同，而且比较简单。但是存在的问题为符号计算可能会花费较多的计算机资源，可能得不到简单的解析解或封闭形式的解，甚至无法求解，所以没有万能的求解微分方程的一般解法。既然没有万能的微分方程一般解法，那么，求解微分方程的符号法和数值法就有很好的互补作用。

对于微分方程，MATLAB 提供了 dsolve 函数指令来对其进行求解，其函数的调用格式如下。

- **r=dsolve('eq1,eq2,... ', 'cond1,cond2,... ', 'v')** 计算由 eq1,eq2,...指定的常微分方程的符号解。常微分方程以变量 v 作为自变量，参数 cond1,cond2,...用于指定方程的边界条件或者初始条件，如果不指定 v，将默认 t 为自变量。

❑ **r=('eq1', 'eq2',... 'cond1','cond2', 'v')** 计算由 eq1,eq2,...指定的常微分方程的符号解，这些常微分方程都以 v 作为自变量，这些单独输入的方程的最大允许个数为 12，其他参数调用方式同上。

在方程中，用 D 来表示一次微分，D2、D3 分别表示二次、三次微分，以此类推。例如符号 *D2y* 来表示 $\frac{D^2y}{Dt^2}$ 函数 dsolve 把 *D* 后面的字符当做因变量，并默认所有这些变量对 *t* 进行求导。

微分方程的初始条件或边界条件都以变量 *v* 作为自变量，其形式为 y(*a*)=*b* 或 Dy(*a*)=*b*，其中 *y* 是微分方程的因变量，*a* 和 *b* 是常数。如果指定的出事条件和边界条件比方程中的因变量个数少，那么所得的解中将包含积分常数 C1、C2 等。

函数 dsolve 的输出结果同 solve 相似，既可以用和因变量个数相同的输出参数分别接收每个变量的解，也可以把方程的解写入一个结构数组中。

【例 3-98】 使用 dsolve 函数指令求解微分方程 $\frac{dx}{dt}=-ax$，$\frac{d^2x}{dt^2}=\sin(t)$，$\left(\frac{dy}{ds}\right)^2+y^2=1$，输入指令如下。

```
>> dsolve('Dx=-a*x');
>>dsolve('D2x/d*t^2=sin(t)')
>> dsolve('(Dy)^2/+y^2=1','s')
```

输出结果如下。

```
ans =
     C2/exp(a*t)
ans =
     C5 - d*sinint(t) + (C4*t^3 + d*t^3*cosint(t) - d*t^2*sin(t))/t^2
ans =
    C7*exp(s)
 C9/exp(s)
```

【例 3-99】 使用 dsolve 函数指令在指定初值的情况下求解微分方程 $\frac{d^2y}{dt^2}=-a^2y$，输入指令如下。

```
>> dsolve('D2y=-a^2*y', 'y(0)=1', 'Dy(pi/2)=0');        %限定初值
```

输出结果如下。

```
ans =
    sin(1/2*a*pi)/cos(1/2*a*pi) *sin(a*t)+cos(a*t)
```

【例 3-100】 使用 dsolve 函数指令在计算微分方程 $\frac{dy}{dx}+3xy=xe^{-x^2}$ 的通解，输入指令如下。

```
>> dsolve('Dy+3*x*y=x*exp(-x^2)');
```

输出结果如下。

```
ans =
    (1/3*exp(-x* (x-3*t))+C1) *exp(-3*x*t)
```

由于系统默认的自变量是 t，显然系统把 x 当作常数，把 y 当做 t 的函数求解，故输入

```
>> dsolve('Dy+3*x*y=x*exp(-x^2)','x');
```

输出结果如下所示。

```
ans =
    exp(-x^2)+exp(-3/2*x^2) *C1
```

与代数方程组的求解相同，dsolve 同样也可以用于解微分方程组，函数指令的调用如下。

r=dsolve('eq1,eq2,... ', 'cond1,cond2,... ', 'v')，计算由 eq1,eq2,...指定的常微分方程组的解。v 作为自变量，参数 cond1,cond2,...用于指定方程的边界条件或初始条件。

【例 3-101】 使用 dsolve 函数指令求解微分方程组，输入指令如下。

```
>>syms x y;
>>S=dsolve('Dx=-y','Dy=x','x(0)=0','y(0)=1');
>> S.x          %查看 x 的值
>> S.y          %查看 y 的值
```

输出结果如下所示。

```
ans =
    -sin(t)
ans =
    cos(t)
```

3.7.3 复合方程的求解

了解了代数求解之后不禁会产生疑问，复合方程怎样求解呢？是否是多次代数方程求解的叠加？针对复合方程，MATLAB 提供了 compose 函数指令来求解复合方程，其调用格式如下。

- **compose(f,g)** 返回函数 $f(g(y))$，其中 x 是 f 的默认变量，即 $f=f(x)$，y 是 g 的默认变量，即 $g=g(y)$。
- **Compose(f,g,z)** 返回函数 $f(g(z))$，自变量指定为 z。
- **compose(f,g,x,z)** 返回函数 $f(g(z))$，指定 f 的自变量是 x。
- **Compose(f,gx,y,z)** 返回函数 $f(g(z))$，指定 f 的自变量是 x，指定 g 的自变量为 y。

【例 3-102】 使用 compose 函数求复合方程的解，其运算过程如下。

```
>> syms x y z u v;
>> f=1/(1-x^2);
>> g=sin(sqrt(x));
>> m=tan(x+1);
```

```
>> n=3*x+2 ;
>> compose(f,g);
>> compose(f,g,u);
>> compose(m,n,u,z);
>> compose(m,n,x,y,z);
>> compose(m,n,u,v,z)
```

输出结果如下：

```
ans =
     1/(1-sin(x^(1/2))^2)
ans =
     1/(1-sin(u^(1/2))^2)
ans =
     tan(3*z+3)
ans =
     tan(1+x)
ans =
     tan(3*x+3)
ans =
     tan(1+x)
```

3.7.4 反方程求解

在了解了代数方程、微分方程及复合方程的求解方式以后，有人会产生疑问，那么反方程是否有便捷的求解方式呢？

MATLAB 同样提供 finverse 函数指令来求解反方程，其调用方式如下。

- **g=finverse(f)** 在 f 函数的反函数存在的情况下，返回 f 函数的反函数自变量为默认变量。
- **g=finverse(f,v)** 在 f 函数的反函数存在的情况下，返回 f 函数的反函数，自变量设置为 v。

【例 3-103】 计算由反函数构成方程的解，输入指令如下。

```
>> syms x y u v X Y U V;
>> f1=x^3-2*y;
>> f2=exp(2*u+v);
>> g1=finverse(f1,y);
>> g2=finverse(f2,u);
>> g3=finverse(f2,v);
>> X= solve(g1,x);
>> Y= solve(g1,y);
>> U= solve(g2,u);
```

输出结果如下。

```
g1 =
    x^3/2-y/2
g1 =
```

```
     log(u)/2 - v/2
g3 =
     log(v) - v/2
X =
    y^(1/3) -1/2*y^(1/3)+1/2*i*3^(1/2) *y^(1/3) -1/2*y^(1/3)+1/2*i*3^(1/2)
   *y^(1/3)
Y =
    x^3
U =
    1/exp(v)
```

3.8 本章小结

本章首先学习了 MATLAB 的符号计算，它是对未赋值的符号对象（可以是常数、变量、表达式）进行运算和处理。数值型计算会在计算过程中产生舍入误差，而符号计算则可以避免这个问题，符号计算在计算过程中不会出现数值型计算，不存在舍入误差问题。然后学习了符号表达式、运算符号运算精度和符号矩阵的计算。最后学习了符号函数的图形绘制和符号方程的求解。通过本章的学习读者可初步掌握符号计算的方式和使用方法。符号运算与数值计算相同，都是科学研究中的重要内容。运用符号运算可以轻松解决许多公式和关系式的推导问题。

3.9 习题

（1）符号运算与数值运算的区别？

（2）求矩阵 $A=\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ 的行列式值、非共轭转置和特征值。

（3）符号表达式 $f=2x^2+3x+4$ 与 $g=5x+6$ 的代数运算。

（4）对表达式 $2\sqrt{5}+\pi$ 进行任意精度控制的比较。

（5）求微分方程 $x\frac{\mathrm{d}^2y}{\mathrm{d}x^2}-3\frac{\mathrm{d}y}{\mathrm{d}x}=x^2$，$y(1)=0$，$y(0)=0$ 的解。