北京航空航天大学 BEIHANG UNIVERSITY

第5讲　MATLAB符号运算

2019年10月9日

符号运算简介

- 求一元二次方程 $ax^2 + bx + c = 0$ 的根

```
>> syms a b x;
>> solve('a*x^2+b*x+c')
ans =
 -(b + (b^2 - 4*a*c)^(1/2))/(2*a)
 -(b - (b^2 - 4*a*c)^(1/2))/(2*a)
```

符号计算：是指在运算时，无须事先对变量赋值，而将所得到结果以标准的符号形式来表示

- 求 $f(x) = (\cos x)^2$ 的一次导数

```
>> x=sym('x'); >> diff(cos(x)^2)  ans =-2*cos(x)*sin(x)
```

- 计算 $f(x) = x^2$ 在区间 $[a, b]$ 上的定积分

```
>> syms a b x; >> int(x^2,a,b)      ans = b^3/3 - a^3/3
```


符号运算简介

- Matlab 符号运算是通过建立在功能强大的 Maple 软件的基础上的符号数学工具箱（Symbolic Math Toolbox）来实现的，当 Matlab 进行符号运算时，请求 Maple 软件去计算并将结果返回给 Matlab
- Matlab 的符号数学工具箱可完成几乎所有符号运算功能，主要包括：符号表达式的运算，符号表达式的复合、化简，符号矩阵的运算，符号微积分、符号作图，符号代数方程求解，符号微分方程求解等
- 支持可变精度运算，即支持以指定的精度返回结果

符号运算简介

- 优势：
  - 符号运算不需要进行数值运算，不会出现截断误差，因此非常准确(数值型每运算一次就会有一次截断误差)
  - 计算以推理方式进行，不受计算误差累积所带来的困扰
  - 符号计算可以给出完全正确的封闭解，或任意精度的数值解（封闭解不存在时）
  - 指令的调用比较简单，与数学教科书上的公式相近
- 劣势：
  - 符号计算所需运行时间相对较长，而数值运算速度较快

内容提要

- 符号对象
  - 符号常量、符号变量、符号表达式、符号矩阵
- 符号表达式的计算与操作
  - 基本运算：变量确定、精度控制
  - 主要操作：因式分解、简化、通分、展开等
  - 微积分运算：符号极限、微分、积分和级数
- 符号方程的求解
  - 符号代数方程求解、符号微分方程求解
- 符号函数可视化

符号对象及表达式

- 在进行符号运算时，必须先定义基本的符号对象，可以是符号常量、符号变量、符号表达式等
- 符号对象是一种数据结构
- 符号矩阵/数组：元素为符号表达式的矩阵/数组
- 符号表达式
  - 含有符号对象的表达式；Matlab 在内部把符号表达式表示成字符串，以与数字变量或运算相区别

符号对象的创建

- sym函数：用于创建单个符号对象
  - S=sym(A , flag)：定义符号对象S
  - A可以是常量、变量、函数或表达式
  - flag可选择d、f、e或r，缺省/默认为r
    - d表示返回最接近的十进制数值（默认为32位）
    - f表示返回最接近的浮点值(N*2^e形式，N和e都为整数)
    - e表示返回最接近的带有机器浮点误差的有理值
    - r表示返回最接近的有理表示（两个整数p和q构成的p/q 、 p*q 、 10^q 、 pi/q 、 2^q 、 sqrt(p) 等形式）

符号对象的创建

- sym函数：参数A示例
  - A可以是常量、变量、函数或表达式

```
>> a=sym('a')      → a 是符号变量
>> b=sym(1/3)      → b 是符号常量
>> c=sym('2*a+b')  → c 是符号表达式
>> d=sym('sin(1)') → c 是符号函数
```

符号对象的创建

- sym函数：参数flag示例
  - flag可选择d、f、e或r，缺省/默认为r

```
>> a=sym(1/3)        a =1/3
>> b=sym(1/3, 'd')   b =0.33333333333333331482961625624739
>> c=sym(1/3, 'f')   c =6004799503160661/18014398509481984
>> d=sym(1/3, 'e')   d =1/3 - eps/12
>> e=sym(1/3, 'r')   e =1/3
```


符号对象的创建

- 符号常量与数值常量的区别

```
>> pi1=sym('pi'); k1=sym('8');k2=sym('4');%定义符号常量
>> pi2=pi;          r1=8;          r2=4;          %定义数值常量
>> whos
```

Name	Size	Bytes	Class	Attributes
k1	1x1	112	sym	
k2	1x1	112	sym	
pi1	1x1	112	sym	
pi2	1x1	8	double	
r1	1x1	8	double	
r2	1x1	8	double	

符号对象的创建

- 符号常量与数值常量的区别

```
>> pi1=sym('pi'); k1=sym('8');k2=sym('4');%定义符号常量
>> pi2=pi;          r1=8;          r2=4;          %定义数值常量
>> a1=sin(pi1/5),          a2=sin(pi2/5)
a1 =    (2^(1/2)*(5 - 5^(1/2))^(1/2))/4
a2 =  0.5878
>> b1=sqrt(k1+sqrt(k2)),  b2=sqrt(r1+sqrt(r2))
b1 =    10^(1/2)
b2 =  3.1623
```


符号对象的创建

- syms函数：用于一次创建多个符号对象
  - syms ('符号变量名1','符号变量名2',...,'符号变量名n')：同时定义多个符号变量名
  - syms '符号变量名1' '符号变量名2'…'符号变量名n'：符号串可以是常量、变量、函数或表达式
  - syms 符号变量名1 符号变量名2 ... 符号变量名n clear：同时清除MATLAB工作空间的变量

例：`>> syms a b c` ⟺

```
>> a=sym('a');
>> b=sym('b');
>> c=sym('c');
```

只能用空格隔开

符号表达式的创建

- 符号表达式的创建
  - 利用单引号来生成符号表达式

```
>> f1='sin(x)+cos(x)'
```

用这种方法创建的符号表达式对空格很敏感，不要在字符间随意添加空格！

  - 利用sym函数建立符号表达式

```
>> f2=sym('sin(x)+cos(x)')
```

  - 使用已经定义的符号变量组成符号表达式

```
>> syms x
>> f3=3*sin(x)+cos(x)
```

推荐！

  - 符号表达式或符号方程可赋给符号变量，以方便调用；也可以不赋给符号变量，直接参与运算

符号矩阵的创建

- 符号矩阵的创建
  - 使用 sym 函数直接生成

```
>> A=sym('[1+x, sin(x); 5, exp(x)]')
```

  - 将数值矩阵转化成符号矩阵

```
>> B=[2/3, sqrt(2); 5.2, log(3)];
>> C=sym(B)
```

与数值矩阵类似：
- 分号区别行
- 逗号或空格区别列
- 用下标引用或赋值

- 符号矩阵中元素的引用和修改

```
>> A=sym('[1+x, sin(x); 5, exp(x)]');
>> A(1,2)                    %引用
>> A(2,2)=sym('cos(x)')      % 重新赋值
```


符号基本运算

- 基本运算
  - Matlab 符号运算采用的运算符和基本函数，在形状、名称和使用上，与数值计算中运算符和基本函数完全相同
  - 矩阵运算：“+”、“-”、“*”、“\”、“/”、“^”
  - 数组运算：“+”、“-”、“.*”、“./”、“.\”、“.^”
  - 矩阵转置：“'”、“.'”

```
>> X=sym('[x11,x12; x21,x22; x31,x32]');
>> Y=sym('[y11,y12,y13; y21,y22,y23]');
>> Z1=X*Y;
>> Z2=X'.*Y;
```


符号基本运算

- 基本运算函数
  - 三角函数与反三角函数、指数函数、对数函数等

sin、cos、tan、cot、sec、csc、…
asin、acos、atan、acot、asec、acsc、…
exp、log、log2、log10、sqrt
abs、conj、real、imag
rank、det、inv、eig、lu、qr、svd
diag、triu、tril、expm

符号表达式的操作

- 通分并提取符号表达式分子和分母： numden函数
- [N,D]=numden(f): N 为通分后的分子，D 为通分后的分母

```
>> syms x y;
>> f=x/y+y/x;
>> [N,D]=numden(f)
N =x^2 + y^2
D =x*y
>> [n,d]=numden(sym(112/1024))
n = 7
d =  64
```


符号表达式的操作

- 因式分解：factor(s)函数

```
>> syms x; f=x^6+1, factor(f)
  f =      x^6 + 1
  ans =   (x^2 + 1)*(x^4 - x^2 + 1)
```

– 也可用于正整数的分解，大整数分解要转化成符号常量

```
>> s=factor(100)
  s =      2     2     5     5
>> factor(sym('12345678901234567890'))
  ans =  2*3^2*5*101*3541*3607*3803*27961
>> factor(12345678901234567890)
  Error using factor (line 27)
  When n is single or double, its maximum allowed value is
  FLINTMAX(9.0072e+15).
```


符号表达式的操作

- 符号表达式展开：expand(s)函数

```
>> syms x; f=(x+1)^6; expand(f)    %多项式展开
   ans =       x^6 + 6*x^5 + 15*x^4 + 20*x^3 + 15*x^2 + 6*x+ 1
>> syms x y; f=sin(x+y); expand(f)  %三角函数展开
   ans =       cos(x)*sin(y) + cos(y)*sin(x)
>> syms x y; f=exp((x+y)^3); expand(f) %指数函数展开
   ans =       exp(3*x*y^2)*exp(3*x^2*y)*exp(x^3)*exp(y^3)
>> syms x y; f=log((x+3)*y); expand(f) %对数函数展开
   ans =        log(3*y + x*y)
```


符号表达式的操作

- 合并同类项： collect函数
  - collect(s)： 对符号表达式s合并同类项
  - collect(s, v) ：对符号表达式s按变量v合并同类项

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

符号表达式的操作

- 符号表达式的化简
  - y= simplify(f)：利用函数规则对表达式 *f* 进行化简
  - y=simple(f)： 对 *f* 尝试多种不同的算法进行简化，返回其中最简短的形式(包含最少数目的字符)
  - [How,y]=simple(f)： y 为 *f* 的最简短形式，How 中记录的为简化过程中使用的方法

f	y	How
2*cos(x)^2-sin(x)^2	3*cos(x)^2-1	simplify
(x+1)*x*(x-1)	x^3-x	combine(trig)
x^3+3*x^2+3*x+1	(x+1)^3	factor
cos(3*acos(x))	4*x^3-3*x	expand

符号表达式的操作

- 符号表达式的化简-示例

```
>> 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)
```

```
>> simple(f)
   simplify:  1
   radsimp:  cos(x)^2 +
   sin(x)^2
   simplify(Steps = 100):1
   combine(sincos): 1
   factor: cos(x)^2 + sin(x)^2
   expand: cos(x)^2 +
   sin(x)^2
   …
   collect(x): cos(x)^2 +
   sin(x)^2

   ans =1
```


符号表达式的操作

- 嵌套形式的多项式： horner 多项式

$$f(x) = x^n + x^{n-1} + \cdots + x + 1$$
$$= x(\cdots x(x(x+1)+1)\cdots)+1$$

```
>> 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
```

- 书写格式美化pretty(f)：显示为习惯的书写形式

```
>> syms x;  f=x^4+2*x^3+4*x^2+x+1;  pretty(f)
        4      3      2
       x  + 2 x  + 4 x  + x + 1
```


符号表达式的操作

- 符号表达式与多项式的转换: sym2poly和poly2sym

```
>> f=sym('2*x+3*x^2+1')
  f =2*x+3*x^2+1
>> sym2poly(f) %将符号表达式转换为按降幂排列行向量
  ans =    3    2    1
>> g=poly2sym([1 3 2])        %将行向量转换符号表达式
  g = x^2+3*x+2          %符号变量默认为x，可指定

>> f1=sym('a*x^2+b*x+c') ; sym2poly(f1)
  ??? Error using ==> sym/sym2poly
  Input has more than one symbolic variable.
```

注：只能对含有一个变量的符号表达式进行转换

符号数值精度控制

- 算数运算方式
  - 数值型： MATLAB的浮点运算
  - 有理数型： Maple的精确符号运算
  - VPA型： Maple的任意精度运算

- 精度控制
  - digits： 用来显示当前计算精度位数，默认为32位
  - digits(n)： 设置计算数值型结果时以n位相对精度进行
  - xr=vpa(x): 给出x在digits指定精度下的数值型结果xr
  - xr=vpa(x,n): 给出x在n位相对精度下的数值型结果xr

符号数值精度控制

- 示例：用三种运算方式表达式比较 2/3 的结果

```
>> a1 = 2/3                    %数值型
   a1 =    0.6667
>> a2 = sym(2/3)               %有理数型
   a2 =    2/3
>> a3 = vpa('2/3', 32)         %VPA型
   a3 =    0.66666666666666666666666666666667
```

- 程序分析
  - 数值型：运算速度最快
  - 有理数型：符号运算结果准确，计算时间、占用内存最大
  - VPA型：比较灵活，可设置任意有效精度，当保留的有效位数增加时，每次运算的时间和使用的内存也会增加

符号表达式的操作

- 符号表达式与数值表达式之间的转换
  - 将数值对象转换为符号对象
    - sym函数：把数值型对象转换成有理数型符号对象
    - vpa函数：将数值型对象转换为任意精度的VPA型符号对象
  - 将符号对象转换为数值对象
    - double函数：将有理数型和VPA型符号对象转换成数值对象

符号表达式的操作

- 符号表达式与数值表达式之间的转换

```
>> a1=sym('2*sqrt(5)+pi')
>> b1=double(a1)          %转换为数值变量
   b1 =      7.6137
>> a2=vpa(a1, 32)         %转换为VPA型数值变量
   a2 =      7.6137286085893726312809907207421
>> b2=double (a2)         %转换为数值变量
   b2 =      7.6137
>> c2=sym(b2)             %转换为符号型数值变量
   c2 =      2143074082783949/281474976710656
```

符号表达式的操作

- 符号变量的确定：findsym函数
  - 帮助用户查找一个符号表达式中的符号变量
  - findsym(s, n)：函数返回符号表达式s中的n个符号变量，若没有指定n，则返回s中的全部符号变量(不按顺序)
  - 小写字母i和j不能作为自由变量
  - 符号表达式中如果有多个字符变量，则按照以下顺序选择自由变量：首先选择x作为自由变量；如果没有x，则选择在字母顺序中最接近x的字符变量；如果与x相同距离，则在x后面的优先
  - 大写字母比所有的小写字母都靠后

符号表达式的操作

- 符号变量的确定：findsym函数

```
>> syms x a y z b w; S1=3*x+y; findsym(S1)
   ans =      x, y
>> syms l m o p q r s t u v w x y z A B
>> aa=l+m-o*p/q+r-s*t/u+v-w*x/y+z-A*B
   aa =       l + m + r + v + z - A*B - (o*p)/q - (s*t)/u - (w*x)/y
>> findsym(aa, 7)
   ans = x,y,w,z,v,u,t     %注意符号变量的顺序
>> c=sym('4'); findsym(a*x+b*y+c)
   ans =      a, b, x, y          %符号常量c不在结果中出现
```


符号表达式的操作

- 符号变量的替换：subs函数
  - subs(f,a)：用*a*替换符号表达式*f*中的(第一)自由变量
  - subs(f,x,a)：用*a*替换符号表达式*f*中指定的符号变量*x*
  - *a*可以是 数/数值变量/表达式 或 字符变量/表达式
  - 若*x*是一个由多个字符变量组成的数组或矩阵，则*a*应该具有与*x*相同的形状的数组或矩阵

符号表达式的操作

- subs函数-示例

```
>> f=sym('2*u');                  f=2*u
>> subs(f,'u',2)                  ans=4
>> f2=subs(f,'u','u+2')           f2=2*(u+2)
>> a=3;
>> subs(f2,'u',a+2)               ans=14
>> subs(f2,'u','a+2')             ans=2*((a+2)+2)
>> syms x y
>> f3=subs(f,'u',x+y)             f3=2*x+2*y
>> subs(f3,[x,y],[1,2])           ans=6
```

符号表达式的操作

- subs函数-示例：数字信号处理中的应用
  - 移位

    f(t) → f(t-t0) ====> subs(f,t,t-t0)
  - 反折

    f(t) → f(-t) ====> subs(f,t,-t)
  - 尺度变化

    f(t) → f(at) ====> subs(f,t,a*t)

符号表达式的高级运算

- 复合函数运算：compose函数
  - compose(f, g)：返回复合函数f(g(y))
  - compose(f, g, z)：返回自变量为z的复合函数f(g(z))
  - compose(f, g, x, z)：返回复合函数f(g(z))，并使x成为f函数的独立变量
  - compose(f, g, x, y, z)：返回复合函数f(g(z))，并使x与y分别成为f与g函数的独立变量

符号表达式的高级运算

- 复合函数运算-示例 $f=\cos(x/t),\ g=\sin(y/u)$

```
>> syms x y z u t
>> f=cos(x/t); g=sin(y/u);
>> compose(f,g)          ans=cos(sin(y/u)/t)
>> compose(g,f)          ans=sin(cos(x/t)/u)
>> compose(f,g,z)        ans=cos(sin(z/u)/t)
>> compose(f,g,x,z)      ans=cos(sin(z/u)/t)
>> compose(f,g,t,z)      ans=cos(x/sin(z/u))
>> compose(f,g,t,u)      ans=cos(x/sin(y/u))
>> compose(f,g,t,u,z)    ans=cos(x/sin(y/z))
```


符号表达式的高级运算

- 反函数运算：finverse函数
  - g = finverse(f,v)：求 f 关于指定变量 v 的反函数，满足 g(f(v))=v
  - g = finverse(f)：求 f 关于默认变量的反函数，满足 g(f(x))=x
  - 例：计算函数 $f=x^2+2t$ 的反函数

```
>> syms x t; f=x^2+2*t;
>> g1=finverse(f),  g2=finverse(f,t)
g1 =  (x - 2*t)^(1/2)
g2 =  - x^2/2 + t/2
```

符号微积分

- 符号极限
- 符号导数
- 符号积分
- 积分变换

符号极限

- 符号极限函数：limit
  - limit(f)：求符号函数 *f* 在变量*x*=0的极限值
  - limit(f, x, a)：求符号函数 *f* 在变量*x*趋向于 *a* 时的极限值
  - limit(f, x, a, 'left')：求符号函数f(x)在*a*处的左极限值
  - limit(f, x, a, 'right')：求符号函数f(x)的右极限值

表达式	函数格式	说明
$\lim_{x\to 0} f(x)$	limit(f)	对 x 求趋近于 0 的极限
$\lim_{x\to a} f(x)$	limit(f,x,a)	对 x 求趋近于 a 的极限，当左右极限不相等时极限不存在。
$\lim_{x\to a^-} f(x)$	limit(f,x,a, 'left')	对 x 求左趋近于 a 的极限
$\lim_{x\to a^+} f(x)$	limit(f,x,a, 'right')	对 x 求右趋近于 a 的极限

符号极限

- 符号极限函数：limit-示例
  - 计算 $L=\lim_{h\to 0}\frac{\ln(x+h)-\ln(x)}{h}$，$M=\lim_{n\to\infty}\left(1-\frac{x}{n}\right)^n$

```
>> syms x h n;
>> L=limit((log(x+h)-log(x))/h, h, 0)
   L =    1/x
>> M=limit((1-x/n)^n, n, inf)
   M =       exp(-x)
```


符号导数

- 符号导数函数：diff
  - g=diff(f)： 求符号表达式 *f* 关于默认变量的导数
  - g=diff(f,v)： 求符号表达式 *f* 关于变量 *v* 的导数
  - g=diff(f, n)：求 *f* 关于默认变量的 *n* 阶导数
  - g=diff(f,v,n)：求 *f* 关于 变量*v* 的 *n* 阶导数

符号导数

• 符号导数函数diff-示例

```
>> f=sym('2*a*x^3+b*x^2+x+c')
   f = 2*a*x^3 + b*x^2 + x + c
>> f=diff(f)          %对默认变量x求一阶微分
   f = 6*a*x^2 + 2*b*x + 1
>> f1=diff(f,'a')          %对符号变量a求一阶微分
   f1 =6*x^2
>> f2=diff(f,'x',2)        %对符号变量x求二阶微分
   f2 = 12*a
>> f3=diff(f,3)            %对默认变量x求三阶微分
   f3 =  0
```


符号积分

• 符号积分函数 int()

– int(f,v,a,b): 计算定积分 $\int_a^b f(v)dv$

– int(f,a,b): 计算关于默认变量的定积分

– int(f,v): 计算不定积分 $\int f(v)dv$

– int(f): 计算关于默认变量的不定积分

符号积分

• 符号积分函数 int()-示例

```
>>syms x;  int(-2*x/(1 + x^2)^2)
   ans =     1/(x^2 + 1)
>>syms x z;        int(x/(1 + z^2), z)
   ans =     x*atan(z)
>>syms x;  int(x*log(1 + x), 0, 1)
   ans =     1/4
>>syms x t;        int(2*x, sin(t), 1)
   ans =     cos(t)^2
```


符号级数求和

• 符号级数求和函数：symsum()

– symsum(f,v,a,b): 求 f 关于 变量v的级数和 $\sum_{v=a}^{b} f(v)$

– symsum(f,a,b): 关于默认变量求和

– 例：计算级数 $S=\sum_{n=1}^{\infty}\frac{1}{n^2}$ 及其前10项的部分和

```
>> syms n; f=1/n^2;
>> S=symsum(f,n,1,inf)
   S =pi^2/6
>> S10=symsum(f,n,1,10)
   S10 =     1968329/1270080
```

积分变换

- Fourier变换/反变换：

$$F(\omega)=\int_{-\infty}^{+\infty} f(t)e^{-j\omega t}dt \qquad f(t)=\frac{1}{2\pi}\int_{-\infty}^{+\infty} F(\omega)e^{j\omega t}d\omega$$

  - fourier(fx,x,t) 求函数f(x)的傅立叶像函数F(t)
  - ifourier(Fw,t,x) 求傅立叶像函数F(t)的原函数f(x)

```
>>syms x t; y=abs(x);
>>Ft=fourier(y,x,t)    %求y的傅立叶变换
      Ft =  -2/t^2
>>fx=ifourier(Ft,t,x)  %求Ft的傅立叶逆变换
      fx =   x*(2*heaviside(x) - 1)
```


积分变换

- Laplace变换/逆变换：用于连续系统（微分方程）

$$F(s)=\int_{0}^{+\infty} f(t)e^{-st}dt \qquad f(t)=\frac{1}{2\pi j}\int_{c-j\infty}^{c+j\infty} F(s)e^{st}ds$$

  - laplace(fx,x,t) 求函数f(x)的拉普拉斯像函数F(t)
  - ilaplace(Fw,t,x) 求拉普拉斯像函数F(t)的原函数f(x)

```
>> x=sym('x');y=x^2;
>> Ft=laplace(y,x,t)      %对y=x^2进行拉普拉斯变换
   Ft = 2/t^3
>> fx=ilaplace(Ft,t,x)    %对函数Ft进行拉普拉斯逆变换
   fx = x^2
```


积分变换

- Z变换/逆变换：用于离散系统（差分方程）

$$F(z)=\sum_{n=0}^{\infty} f(n)z^{-n} \qquad f(t)=Z^{-1}\{F(z)\}$$

  - ztrans(fn,n,z)： 求fn的Z变换像函数F(z)
  - iztrans(Fz,z,n) ： 求Fz的z变换原函数f(n)

```
>> syms n z, fn=exp(-n);
>> Fz=ztrans(fn,n,z)      %求fn=e^-n的Z变换
   Fz = z/(z - exp(-1))
>> f=iztrans(Fz,z,n)      %求Fz的逆Z变换
f = kroneckerDelta(n, 0) - exp(-
   1)*(exp(1)*kroneckerDelta(n, 0) - exp(-1)^n*exp(1))
```


Taylor级数

- 泰勒级数展开为幂级数: taylor函数
  - taylor(f)：返回表达式f中的默认符号变量v(v=findsym(f))的6阶的Maclaurin多项式（即在零点附近v=0）近似式
  - taylor(f, n, v)：返回表达式f中指定符号自变量v（若表达式f中有多个变量时）的n-1阶的Maclaurin多项式（即在零点附近v=0）近似式，其中v可为字符串或符号变量
  - taylor(f, n, v, a)：返回符号表达式f中的指定符号自变量v的n-1阶的Taylor级数（在指定的a点附近v=a）的展开式

Taylor 级 数

• taylor函数-示例：求e^x的泰勒展开式

```
>> syms x
>>s1=taylor(exp(x),8)   %展开前8项
  s1 =      1+x+1/2*x^2+1/6*x^3+1/24*x^4+1/120*x^5
      +1/720*x^6+1/5040*x^7
>>s2=taylor(exp(x)) , pretty(s2)          %默认展开前6项
  s2 =      1+x+1/2*x^2+1/6*x^3+1/24*x^4+1/120*x^5
  ans =       5     4     3     2
              x     x     x     x
            ---- + --- + --- + --- + x + 1
            120    24    6     2
```


符号方程的求解

符号代数方程的求解-solve函数

– MATLAB符号运算能够求解一般的线性方程、非线性方程及代数方程和代数方程组。当方程组不存在符号解，又无其他自由参数时，则给出数值解

– 在MATLAB中提供了solve函数，用于求解符号表达式表示的代数方程

– solve(f)：求方程关于指定自变量的解，f 可以是用字符串表示的方程、符号表达式或符号方程

– solve('eq1', 'eq2',…, 'v1','v2',…) : 求方程组关于指定变量的解，注意解向量的顺序

符号方程的求解

• 符号代数方程的求解-示例

– 求方程ax2+bx+c=0和sinx=0的解

```
>> syms x y z
>> f1=sym('a*x^2+b*x+c')        %无等号
f1 =a*x^2+b*x+c
>> solve(f1)                    %求方程的解x
ans =     [ 1/2/a*(-b+(b^2-4*a*c)^(1/2))]
          [ 1/2/a*(-b-(b^2-4*a*c)^(1/2))]
>> f2=sym('sin(x)')
f2 =sin(x)
>> solve(f2,'x')
ans =0
```


符号方程的求解

• 符号代数方程的求解-示例

– 求三元非线性方程组的解

```
>> syms x y z
>> eq1=sym('x^2+2*x+1');
>> eq2=sym('x+3*z=4');
>> eq3=sym('y*z=-1');
>> [x,y,z]=solve(eq1,eq2,eq3)      %解方程组并赋值给x,y,z
x =          -1
y =          -3/5
z =          5/3
```

如果按MATLAB数值运算求解，怎么解？

符号方程的求解

- 符号微分方程求解：dsovle函数
  - dsolve(‘eqn1’,‘condition’,‘var’) ：求解微分方程的特解
  - dsolve(‘eqn1‘,’,…,’eqnN‘,’cond1‘,…,’condN‘,’var1‘,…,’varN‘)：求微分方程组的特解
    - var描述方程中的自变量符号，省略时按缺省原则处理
    - 若没有给出初值条件condition，则求方程组的通解
    - 当y是因变量时，微分方程’eqn’的表述规定为：
      - y的一阶导数或表示为Dy; y的n阶导数或表示为Dny
    - 初始条件少于微分方程数时，解中将出现任意常数符C1，C2……，其数目等于所缺少的初始条件数

符号方程的求解

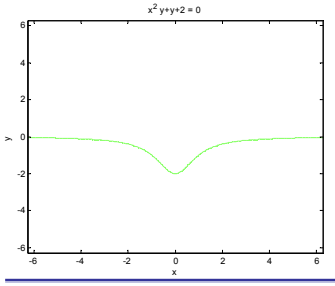
- 符号微分方程求解-示例
  - 求微分方程 $x\frac{d^2y}{dx^2}-3\frac{dy}{dx}=x^2$ ，y(1)=0，y(5)=0的解

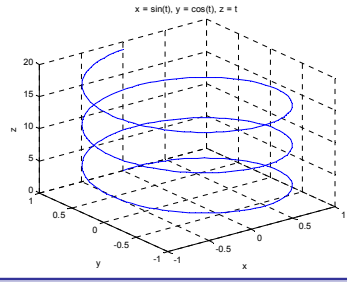
```
>> y=dsolve('x*D2y-3*Dy=x^2','x') %求微分方程的通解
y = -1/3*x^3+C1+C2*x^4
>> y=dsolve('x*D2y-3*Dy=x^2','y(1)=0,y(5)=0','x') %求微分方程的特解
y = -1/3*x^3+125/468+31/468*x^4
```


符号函数可视化

- ezplot(3)函数
  - 用于绘制符号表达式的自变量与对应的函数值的曲线

```
>> ezplot('x^2*y+y+2')
>> ezplot3('sin(t)','cos(t)','t',[0,6*pi])
```





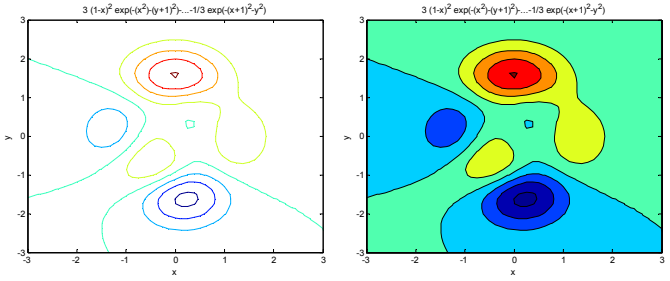
符号函数可视化

- ezcontour(f)函数
  - 用于绘制(带填充颜色)的等高线，可自动添加标题和坐标轴标准

```
>> f = @(x,y) 3*(1-x).^2.*exp(-(x.^2) - (y+1).^2) ...
   - 10*(x/5 - x.^3 - y.^5).*exp(-x.^2-y.^2) ...
   - 1/3*exp(-(x+1).^2 - y.^2);
>> ezcontour(f,[-3,3],49)
>> ezcontourf(f,[-3,3],49)
```

符号函数可视化

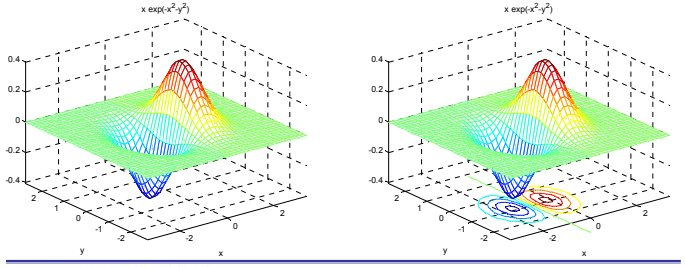
- ezcontour(f)函数
  - 用于绘制(带填充颜色)的等高线，可自动添加标题和坐标轴标准



符号函数可视化

- ezmesh(c)函数
  - 用于绘制(带等位线)的三维网格图

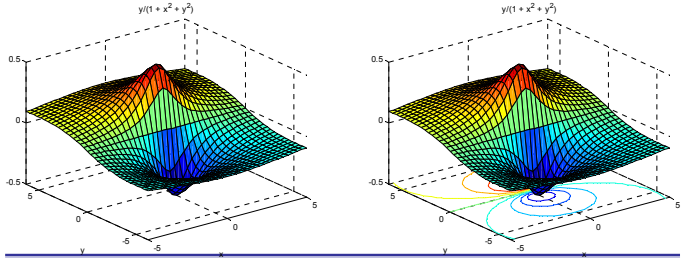
```
>> fh = @(x,y) x.*exp(-x.^2-y.^2);
>> ezmesh(fh,40), figure, ezmeshc(fh,40)
```



符号函数可视化

- ezsurf(c)函数
  - 用于绘制(带等位线)的曲面图

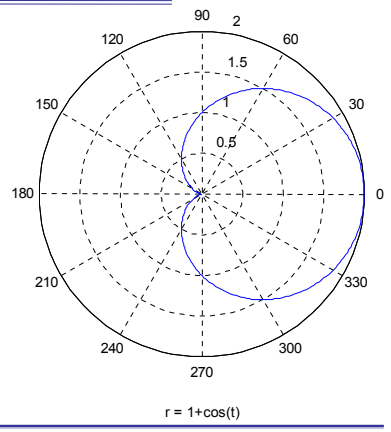
```
>>ezsurf('y/(1 + x^2 + y^2)',[-5,5,-2*pi,2*pi],35)
>> figure, ezsurfc('y/(1 + x^2 + y^2)',[-5,5,-2*pi,2*pi],35)
```



符号函数可视化

- ezpolar函数
  - 用于绘制 极坐标曲线

```
>> ezpolar('1+cos(t)')
```



课外实验四

- **实验名称：MATLAB的符号运算**
- **实验目的：通过实验使了解MATLAB符号运算基本流程与主要方法**
- **实验内容：**
  - **掌握如何创建、修改符号常量、变量、表达式和矩阵；**
  - **掌握符号表达式的基本运算和常用操作和转换；**
  - **掌握符号极限、符号微积分和级数运算；**
  - **用数值和符号积分法求y=-x^2+115在x从0到10之间所围面积，并讨论步长和积分方法对精度的影响。**