第4讲 MATLAB数值运算

2019年10月9日

内容提要

MATLAB 有出色的数值计算能力，

占据世界上数值计算软件的主导地位

内容提要

- 数据统计处理
- 复数及其运算
- 多项式及其运算
- 曲线拟合与插值
- 函数优化
- 代数方程组求解
- 常微分方程求解

数据统计处理

- 求最大与最小元素
- 矩阵的平均值与中值
- 矩阵元素求和与求积
- 矩阵元素的累加和与累乘积
- 标准方差
- 相关系数
- 元素排序

求最大与最小元素

- max函数(min函数用法与之相同)
- 求向量的最大与最小元素
  - C=max(A)：返回向量A的最大元素，存入C
  - [C, I] = max(A)：返回向量A的最大元存入C，最大元素的序号存入I
- 求矩阵的最大与最小元素
  - max(A)：返回一个行向量，向量的第i个元素是A矩阵的第i列上的最大元素
  - C = max(A, [ ], dim)：dim取1或2
  - [C, I] = max(A)：返回两个行向量，C向量记录A的每列的最大元素，I向量记录每列最大元素的行号

求最大与最小元素

- 两个向量或矩阵对应元素的比较
  - U = max(A,B)：其中A、B是同型向量或矩阵。结果U是与A、B同型的向量或矩阵，U的每个元素等于A、B对应元素的较大者
  - U=max(A,n)：其中n是一个标量，结果U是与A同型的向量或矩阵，U的每个元素等于A对应元素和n中的较大者
- 示例

```
>> a=[1,2,3;4,5,6]；  max(a,2),  max(a,[],2)
    ans =   2   2   3
            4   5   6
    ans =   3
            6
```


矩阵的平均值

- mean函数：求矩阵或向量的平均值
- 调用格式
  - M = mean(X)：返回向量X的算术平均值
  - M = mean(A)：返回一个行向量，其第i个元素是A的第i列的算术平均值
  - M = mean(A, dim)：当dim为1时，该函数等同于mean(A)；当dim为2时，返回一个列向量，其第i个元素是A的第i行的算术平均值

矩阵的中值

- median函数：求矩阵或向量的中值
- 调用格式
  - M = median(X)：返回向量X的中值
  - M = median(A)：返回一个行向量，其第i个元素是A的第i列的中值
  - M = median(A, dim)：当dim为1时，该函数等同于mean(A)；当dim为2时，返回一个列向量，其第i个元素是A的第i行的中值

矩阵元素求和

- sum函数：用于矩阵和向量求和的基本函数
- 调用格式
  - B = sum(X)：返回向量X各元素的和
  - B = sum(A)：返回一个行向量，其第i个元素是A的第i列的元素和
  - B = sum(A, dim)：当dim为1时，该函数等同于sum(A)；当dim为2时，返回一个列向量，其第i个元素是A的第i行的各元素和

矩阵元素求积

- prod函数：用于矩阵和向量求积的基本函数
- 调用格式
  - B = prod(X)：返回向量X各元素的乘积
  - B = prod(A)：返回一个行向量，其第i个元素是A的第i列的元素乘积
  - B = prod(A, dim)：当dim为1时，该函数等同于prod(A)；当dim为2时，返回一个列向量，其第i个元素是A的第i行的各元素乘积

矩阵元素的累加和

- cumsum函数：求向量与矩阵元素的累加和向量
- 调用格式
  - B = cumsum(X)：返回向量X累加和向量
  - B = cumsum(A)：返回一个矩阵，其第i列是A的第i列的累加和向量
  - B = cumsum(A, dim)：当dim为1时，该函数等同于cumsum(A)；当dim为2时，返回一个矩阵，其第i行是A的第i行的累加和向量

矩阵元素的累乘积

- cumprod函数：计算向量与矩阵元素的累乘积向量
- 调用格式
  - B = cumprod(X)：返回向量X累乘积向量
  - B = cumprod(A)：返回一个矩阵，其第i列是A的第i列的累乘积向量
  - B = cumprod(A, dim)：当dim为1时，该函数等同于cumsum(A)；当dim为2时，返回一个矩阵，其第i行是A的第i行的累乘积向量

示 例

- >> a=magic(3)
- a =

8	1	6
3	5	7
4	9	2

- >> sum(a)
- ans =

15	15	15

- >> prod(a)
- ans =

96	45	84

- >> cumsum(a)
- ans =

8	1	6
11	6	13
15	15	15

- >> cumprod(a)
- ans =

8	1	6
24	5	42
96	45	84

- >> cumprod(a,2)
- ans =

8	8	48
3	15	105
4	36	72

标准方差

- std函数：计算数据序列的标准方差
- 调用格式
  - S = std(X)：对于向量X返回一个标准方差
  - S = std(A)：对于矩阵A，即返回一个行向量，其各个元素便是矩阵A各列或各行的标准方差
  - S = std(A, flag, dim)：当dim为1时，即求各列元素的标准方差；当dim为2时，则求各行元素的标准方差； flag为0时利用N-1标准化， flag为1时利用N标准化

相关系数

- corrcoef函数：实现求数据的相关系数矩阵
- 调用格式：
  - R = corrcoef(X)：返回从矩阵X形成的一个相关系数矩阵。此相关系数矩阵的大小与矩阵X一样。它把矩阵X的每列作为一个变量，然后求它们的相关系数
  - R = corrcoef(x,y)：在这里，x, y是向量，它们的作用与corrcoef([x,y])中一样

元素排序

- sort函数：实现对向量或矩阵的排序
- 调用格式：
  - B=sort(X)：返回对X向量中的元素按升序排列的新向量
  - B=sort(A)：对矩阵A的各列（或行）进行重新排序
  - [B, IX] = sort(A, dim)：dim指明对A的列还是行进行排序，若dim为1，则按列排序；若dim为2，则按行排序。B是排序后的矩阵，而IX记录B中的元素在A中的位置

复数及其运算

- 创建复数
- 复数基本运算
- 复数绘图
- 留数的基本运算

创建复数

- Matlab表示复数时，可用字母i或j表示虚部(等价)
- 创建复数可直接输入或利用函数complex(a,b)

```
>> 2-3i        %虚部与虚数单位之间不能留空格
   ans =
        2.0000 - 3.0000i
>>a=2; b=-3; complex(a, b)
   ans =
        2.0000 - 3.0000i
>>a=2-3i; b=2-3j; a-b
   ans =
        0
```

z=complex(3) ⟷ z=complex(3,0)

复数基本运算

- 复数的加、减、乘、除法、复数上的三角函数计算、复数的幂的计算、指数与对数的计算等，与实数函数的语法相同
  - 提取实部与虚部可以用函数real、imag
  - 计算模可以用函数abs
  - 计算辐角可以用函数angle
  - 复数的共轭可以用函数conj

复数绘图

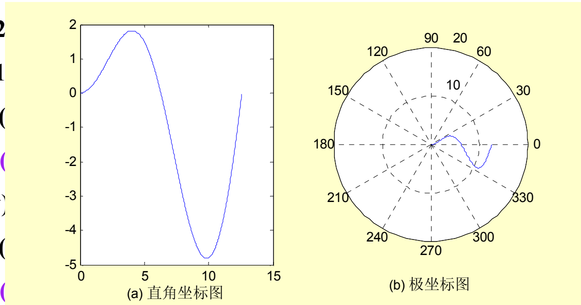
- 复数函数的绘图主要有两种形式：一种是直角坐标图，即分别以复数的实部和虚部为坐标作出复数的表示图；另一种为极坐标图，即分别以复数的模和幅角为坐标作图
- MATLAB提供了绘制极坐标图的函数polar，该函数还可绘制出极坐标栅格线，其调用格式如下：
  - polar(theta,rho)
  - polar(theta,rho,LineSpec)

  其中，theta为极坐标极角，rho为极坐标矢径，LineSpec为绘制线型

复数绘图

• 示例：函数y=2t+i$t$sin(t)在两种坐标下的效果图

```
clear all;
t=0:0.02
y=2*t+1
subplot(
xlabel('(
r=abs(y)
subplot(
xlabel('(
```



(a) 直角坐标图　　(b) 极坐标图

多项式运算

• 多项式的创建

• 多项式的算术运算与求导

• 多项式的求值与求根

• 多项式的微积分

• 多项式部分分式展开

多项式运算

• Matlab语言把多项式表达成一个**行向量**，该向量中的元素是按多项式**降幂**排列的

– 如多项式　$f(x)=a_nx^n+a_{n-1}x^{n-1}+\ldots\ldots+a_1x+a_0$

– 可用行向量　$p=[a_n\ a_{n-1}\ \ldots\ldots\ a_1\ a_0]$　表示

例　$p(x)=x^3+2x^2+5$

```
>> p=[1 2 0 5]
```


多项式运算

• **poly函数：** 产生特征多项式系数向量

– 调用格式：　p=poly(a)　　　%a为多项式的解

– 特征多项式一定是n+1维的，且第一个元素一定是1

• **poly2str函数：** 显示数学多项式的形式

– 调用格式：p1=poly2str(p, 'x') %p为多项式向量，x为变量

• 示例：

```
>>a=[1 2 3]; p=poly(a)
p = 1    -6    11    -6
>> p1= poly2str(p, 'x')    p2= poly2sym(p, 'x')
p1 = x^3 - 6 x^2 + 11 x – 6
P2 = x^3 - 6*x^2 + 11*x - 6
```

多项式运算

- 多项式的加、减运算
  - 多项式的加、减运算直接用“+”、“-”来实现
  - 多项式相加、减时，两个系数向量必须大小相等
  - 阶次不同时，低阶多项式必须用0填补，使其与高阶多项式有相同的阶次
- 示例： 已知：$a(x)=x^2+2x+3$; $b(x)=5x+6$;

  求解：c=a+b; d=a-b.

```
>>a=[1 2 3]; b=[0 5 6];
>>c=poly2str(a+b, 'x'), d=poly2str(a-b, 'x')
  c =   x^2 + 7 x + 9
  d =   x^2 - 3 x - 3
```


多项式运算

- 多项式的乘运算
  - 多项式的乘法用函数conv(p1, p2)来实现，相当于执行两个数组的卷积
- 示例： 已知：$a(x)=x^2+2x+3$; $b(x)=4x^2+5x+6$;

  求解：c = (x2+2x+3)(4x2+5x+6)

```
>>a=[1 2 3];b=[4 5 6];
>>c=conv(a, b)  %或 c=conv([1 2 3],[4 5 6])
  c = 4  13  28   27   18
>>p=poly2str(c,'x')
  p = 4 x^4 + 13 x^3 + 28 x^2 + 27 x + 18
```


多项式运算

- 多项式的除运算
  - 多项式的除法用函数 deconv(p1, p2) 来实现，相当于执行两个数组的解卷
- 示例

```
>> a=[1 2 3];   c = [4  13  28   27  18]
>> d=deconv(c,a)
  d =4   5   6
```

[d,r]=deconv(c,a)

r：余数

d：c除a后的整数

多项式运算

- 多项式的根：roots函数
  - n次多项式具有n个根，这些根可以是实根，也可以含有若干对共轭复根
- 调用格式：
  - r = roots(c)   % c 为多项式的系数向量
  - c = poly(r)   % r 为多项式的根

  - 对于一个方阵s，可以用函数poly来计算矩阵的特征多项式系数。特征多项式的根即为特征值，可以用roots函数来计算

多项式运算

- roots函数+ poly函数
  - 其结果可互相转换
- 示例

```
>>a=[1 2 3];p=poly(a)
p = 1   -6   11   -6
>>r=roots(p)
r = 3
    2
    1
>>p2=poly(r)
p2 = 1.0000   -6.0000   11.0000   -6.0000
```

Matlab规定：

•多项式系数向量用行向量表示

•一组根用列向量表示

多项式运算

- 多项式的求值：polyval函数
  - y = polyval(p,x)：对多项式p在x处求值（按数组方式）
- 示例

```
>>p=[1 2 1]; y=polyval(p,2)
y=9
>>y=polyval(p, 0:3)
y=   1   4   9   16
>>x=[1 2;3 4]; y=polyval(p,x)
y=   4    9
     16   25
```


多项式运算

- 多项式的求值：polyvalm函数
  - y=polyvalm(p,x)：对多项式p在x(方阵)处求值(按矩阵方式)
- 示例

```
>>p=[1 2 1]; y=polyvalm(p,2)
y=9
>>y=polyvalm(p, 0:3)
Error using polyvalm (line 28)
Matrix must be square.
>>x=[1 2;3 4]; y=polyvalm(p,x)
y=   10   14
     21   31
```

$$y=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}^2+2\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}+1E$$

多项式运算

- 多项式的微分:polyder函数
  - k = polyder(p)：对多项式p进行微分运算
  - k = polyder(a,b)：对多项式a与b乘积进行微分运算
  - [p, q]=polyder(a,b): 对多项式a与b的商进行微分运算，以q/d格式表示
- 示例:

```
>>a=[1 2 3 4 5];  poly2str(a, 'x')
ans = x^4 + 2 x^3 + 3 x^2 + 4 x + 5
>>b=polyder(a),  poly2str(b, 'x')
b = 4    6    6    4
ans =4 x^3 + 6 x^2 + 6 x + 4
```

多项式运算

- 多项式的积分：polyint函数
  - polyint(p, k)：返回以向量p为系数的多项式的积分，积分的常数项为k
  - polyint(p)：返回以向量p为系数的多项式的积分，积分的常数项为默认值0
- 示例：

```
>>b = [4  6  6 4]; poly2str(b, 'x')
   ans =4 x^3 + 6 x^2 + 6 x + 4
   >>a = poly2int(b, 'x'), poly2str(a, 'x')
   a= 1   2   3   4   5
   ans = x^4 + 2 x^3 + 3 x^2 + 4 x + 5
```


多项式运算

- 多项式部分分式展开
  - 在信号处理和控制系统的分析应用中，常常需要将分母多项式和分子多项式构成的传递函数进行部分分式展开。在MATLAB中可运用residue函数实现部分分式展开
- [r, p, k] = residue(b, a)：求多项式之比b/a的分式展开，b、a分别为分子和分母多项式系数的行向量，r为留数行向量，p是部分分式的极点，k是常数项
- 如果多项式a没有重根，展开的形式如下：

$$\frac{b(x)}{a(x)} = \frac{r_1}{x-p_1} + \frac{r_2}{x-p_2} + \cdots + \frac{r_n}{x-p_n} + k_s$$

- 多项式分母有重根时使用resi2命令

多项式运算

- 多项式部分分式展开-示例

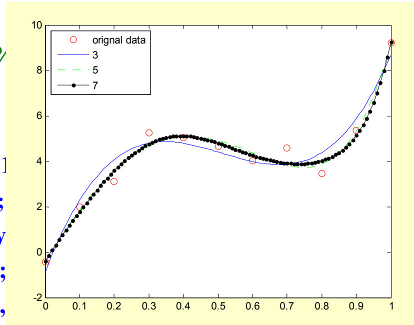
```
          3S+8
      -------------
         S2+5S+6
>>B=[3 8];A=[1 5 6];
>>[R, P,K]=residue(B,A)
   R=  1      2
   P=  -3    -2
   K= [ ]

   1          2
-------- + --------
  S+3       S+2
```


曲线拟合

- 曲线拟合：用一个参数化的曲线来逼近一组给定的数据点。若参数化曲线是多项式，称为多项式拟合
- polyfit函数
  - p=polyfit(x, y, n)   %
- 示例

```
>>x0=0:0.1:1;
>>y0=[-.447 1.978 3.11
>>p1=polyfit(x0, y0, 3);
>>x=0:0.01:1;y1=poly
   …y3=polyval(p3,x);
>>plot(x,y1,'-b',x0,y0,
```



曲线拟合-示例

- 问题：测得某单分子化学反应速度数据如下表所示,x表示从实验开始算起的时间，y表示对应时刻反应物的量，满足指数函数 y=a*exp(b*x)，求参数a,b的最小二乘解

序号	1	2	3	4	5	6	7	8
x	3	6	9	12	15	18	21	24
y	57.6	41.9	31.0	22.7	16.6	12.2	8.0	6.5

- **解**：对y=a*exp(b*x)两边取对数得：　ln(y)=ln(a)+b*x

  令y'=ln(y), a'=b, b'=ln(a), 则　　y'=a'*x+b'

  曲线拟合求得a'，b'，即可反求得a'和b'

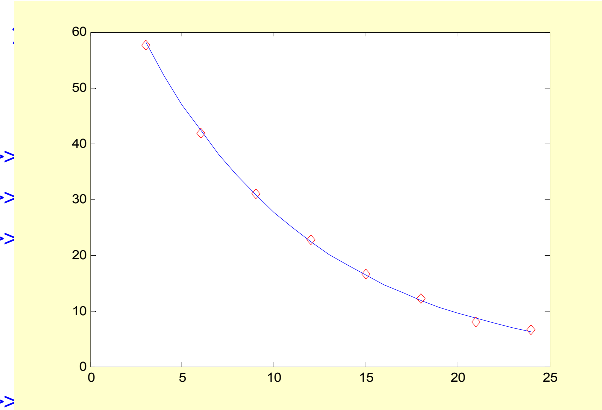
曲线拟合-示例

-

\>>

\>>

\>>



\>>

插 值

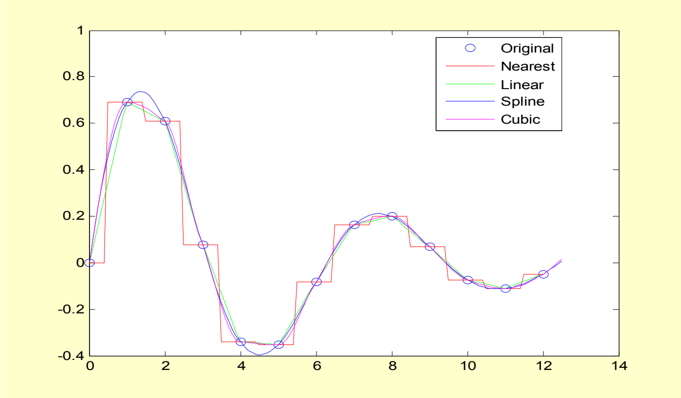
- 定义：是对某些集合给定的数据点之间函数的估值方法，即利用已知点确定未知点
- 当不能很快地求出所需中间点的函数时，插值是一个非常有价值的工具
- Matlab提供了一维、二维、三次样条等许多插值选择
- 用途：图像旋转、缩放等

插　值

- 一维插值：**interp1函数**
  - yi=intper1(x,y,xi,method)：向量x是数据点的x坐标，向量y是数据点的y坐标，xi是插值点，字符串method则规定插值的方法，共有4种
    - 最近邻内插法(method= ' nearest ')
    - 线性内插法(method= ' linear ')，为默认设置
    - 三次样条内插法(method= ' spline ')
    - 三次多项式内插法(method= ' cubic ')

插　值

- 一维插值：interp1函数-四种插值法比较

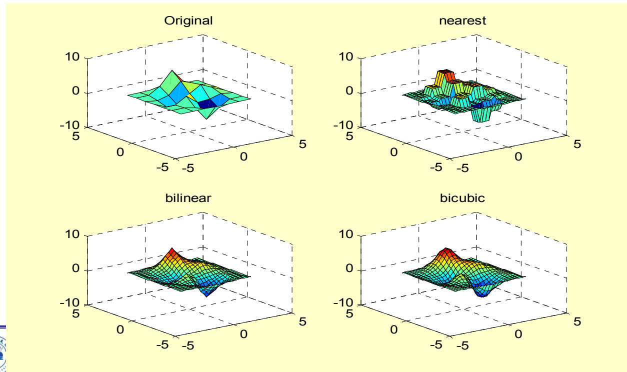


插　值

- 二维栅格点插值：intper2函数
  - zi=intper2(x,y,z,xi,yi,method)：其中z是一个矩阵，代表一个函数的高度，矩阵x,y是此函数在栅格点的x,y坐标，字符串method规定插值的方法，主要有4种
    - 临近点内插法(method= 'nearest')
    - 二维线性内插法(method= ‘ bilinear’)，为默认设置
    - 二维样条内插法(method= ' spline ')
    - 二维三次多项式内插法(method= ' bicubic ')

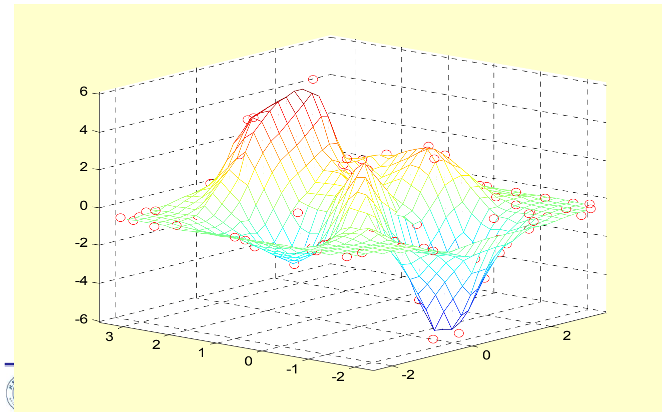
插　值

- 二维栅格点插值：intper2函数
  - 示例:用二维栅格点内插演示4种内插方法的比较



插　值

- 二维散布点插值： griddata函数



插　值

- 三维/高维栅格点插值：**intper3/ interpn函数**
  - vi=interp3(x, y, z, v, xi, yi, zi, method)：其中，x，y，z，v是三维矩阵，前三者表示数据点的输入部分，v是数据点的输出部分。 xi, yi, zi, 是内插点，字符串method指定不同的内插方法，共有四种
  - 临近点内插　（method='nearest'）
  - 三维线性内插（method='linear'），　为默认设置
  - 三维样条内插（method='spline'）
  - 三维三次内插（method='cubic'）
  - 注：使用interp3时，矩阵x，y，z必须是严格的递增或递减。一般地说， x，y，z 数据由ndgrid 命令产生，以保证格式的正确性

函数优化

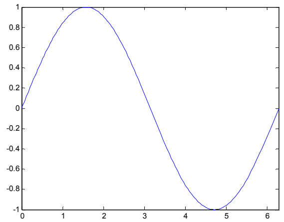
- **fminbnd 函数**：无约束单变量寻优函数
- 调用格式
  - x=fminbnd (fname, x0,x1)：fname 是要求极小值的函数名，x0和x1是指定的搜索范围
- 示例

>>fminbnd ('sin', 0,1)

ans=0

>>fminbnd ('sin', 4,5)

ans=4.7124



>>fplot ('sin', [0,2*pi]) %画出函数sin(x)在区间[0 2*pi]上的图形

函数优化

- **fminsearch函数**：可计算多元函数最小值点
- 调用格式
  - [X,FVAL,EXITFLAG,OUTPUT] = fminsearch(FUN,X0)
  - 输入参数：
    - FUN是函数名,
    - X0指定初始的搜索位置
  - 输出参数：
    - X最小值点
    - FVAL是最小值点处的函数值
    - EXITFLAG=1表示计算成功
    - OUTPUT是一个结构数组，记录计算过程的参数

函数优化

- **MATLAB求解优化问题的主要函数**

类　型	模　　型	基本函数名
一元函数极小	Min F（x）s.t.x1<x<x2	x=fminbnd('F',x_1,x_2)
无约束极小	Min F(X)	X=fminunc('F',X_0) X=fminsearch('F',X_0)
线性规划	Min $c^T X$ s.t.AX<=b	X=linprog(c,A,b)
二次规划	Min $x^T Hx+c^T x$ s.t. Ax<=b	X=quadprog(H,c,A,b)
约束极小（非线性规划）	Min F(X) s.t. G(X)<=0	X=fmincon('FG',X_0)
达到目标问题	Min r s.t. F(x)-wr<=goal	X=fgoalattain('F',x,goal,w)
极小极大问题	Min max $\{F_i(x)\}$ X　{Fi(x)} s.t. G(x)<=0	X=fminimax('FG',x_0)

函数优化-示例

- **fminsearch函数-示例**

  >>fminsearch (@sin,3,optimset('disp','iter'))

  ans = 4.7124

Iteration	Func-count	min f(x)	Procedure
0	1	0.14112	
1	2	-0.00840725	initial simplex
2	4	-0.303542	expand
3	6	-0.788525	expand
4	8	-0.998054	reflect
5	10	-0.998054	contract outside
…	…		
14	28	-1	contract inside
15	30	-1	contract inside
16	32	-1	contract inside
17	34	-1	contract inside

函数优化-示例

- **fminsearch函数-示例**

  >>X = fminsearch('-sin(sqrt(x(1)^2+x(2)^2))/sqrt(x(1)^2+x(2)^2)',[2 2])

  >>FUN=inline('-sin(sqrt(x(1)^2+x(2)^2))/sqrt(x(1)^2+x(2)^2)')

  >>X = fminsearch(FUN,[2 2])

  X = 1.0e-04 * -0.4133 -0.1015

  – **注意：在fminsearch中的函数FUN函数的变量为x(1), x(2),...,不能用其它的变量名**

代数方程组求解

- **对于代数方程**

  ax-b=0，其中a 为n×m矩阵

  有三种情况：

  – **当n=m时，此方程成为“恰定”方程**

  – **当n>m时，此方程成为“超定”方程**

  – **当n<m时，此方程成为“欠定”方程**

- **Matlab定义的除运算可方便地求解上述三种方程**

  – **Matlab中有两种除运算左除和右除**

恰定方程组的解

- **方程ax-b=0(a为非奇异)的解**

  $x=a^{-1}\,b$

  矩阵逆

- **Matlab求解方法**

  – **x=inv(a)*b： 采用求逆运算解方程**

  – **x=a\b： 采用左除运算解方程**

  – **注：除法解方程的速度要比求逆法快2.5倍！**

恰定方程组的解-示例

$$\begin{cases} x_1+2x_2=8 \\ 2x_1+3x_2=13 \end{cases} \longrightarrow \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 8 \\ 13 \end{bmatrix}$$

方程 ax=b　　　　a x = b

```
>>a=[1 2;2 3];b=[8;13];
>>x=inv(a)*b      %或  x=a\b
   x =  2
        3
```


超定方程组的解

- 方程 ax=b , n>m时的解：不存在唯一解
- 方程解 (a'a) x=a'b

  $x=(a'a)^{-1}a'b$ —— 求逆法
- Matlab求解方法
  - x=a\b ：Matlab用最小二乘法找一个准确的基本解
  - $x=(a'*a)^{-1}$ * a' * b：利用求逆法直接求解

超定方程组的解-示例

$$\begin{cases} x_1+2x_2=1 \\ 2x_1+3x_2=2 \\ 3x_1+4x_2=3 \end{cases} \longrightarrow \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

方程 ax=b　　　　a x = b

```
>>a=[1 2;2 3;3 4];b=[1;2;3];
>>x=a\b     %或者 x=inv(a'*a)*a'*b
   x =  1
        0
```


欠定方程组的解

- 方程 ax=b , n<m时的解：有无穷多个解存在
- Matlab求解方法
  - 用除法求得的解x：是具有最多零元素的解
  - 基于伪逆pinv求得的解：是具有最小长度或范数的解

欠定方程组的解-示例

$$\begin{cases} x_1+2x_2+3x_3=1 \\ 2x_1+3x_2+4x_3=2 \end{cases} \longrightarrow \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

方程 ax=b　　　　a　x　=　b

```
>>a=[1 2 3;2 3 4];b=[1;2];
>>x=a\b
x =  1
     0
     0
```

```
>>x=pinv(a)b
x =  0.8333
     0.3333
    -0.1667
```


常微分方程(ODE)求解

- 常微分方程

$$\frac{dy}{dt} = f(x, y),\ y(x_0) = y_0$$

- 解法：
  - 解析解：准确的解
  - 数值解：实际的微分方程较复杂，一般不能给出解析表达式，因此采用数值解法求近似解

常微分方程求解

- 常微分方程的解析解
  - 函数dsolve用来解符号常微分方程、方程组，如果没有初始条件，则求出通解；如果有初始条件，则求出特解
  - r = dsolve('eq1,eq2,...', 'cond1,cond2,...', 'v')
  - ‘eq1,eq2,...’为微分方程或微分方程组
  - ‘cond1,cond2,...’是初始条件或边界条件
  - ‘v’是独立变量，默认的独立变量是't'
-

常微分方程求解

- 常微分方程的解析解
  - r = dsolve('eq1,eq2,...', 'cond1,cond2,...', 'v')
  - 示例 求解常微分方程dy/dx=x
  - >>dsolve(‘Dy=x’,‘x’)
  - x^2/2 + C10
  - 示例 求解常微分方程组 $\begin{cases} \frac{dx}{dt}+2x-\frac{dy}{dt}=10\cos t, & x\big|_{t=0}=2 \\ \frac{dx}{dt}+\frac{dy}{dt}+2y=4e^{-2t}, & y\big|_{t=0}=0 \end{cases}$
  - >> [X,Y]=dsolve(‘Dx+2*x-Dy=10*cos(t), … Dx+Dy+2*y=4*exp(-2*t)', 'x(0)=2, y(0)=0', 't')
  - X =4*cos(t) - 2*exp(-2*t) + 3*sin(t) - 2*exp(-t)*sin(t)
  - Y =sin(t) - 2*cos(t) + 2*exp(-t)*cos(t)

常微分方程求解

- 常微分方程的数值解
  - 以上是常微分方程的精确解法，也称为常微分方程的符号解
  - 有大量的常微分方程虽然从理论上讲，其解是存在的，但却无法求出其解析解，此时，需要寻求方程的数值解
  - MATLAB有丰富的求解常微分方程数值解的函数，统称为solver
  - 一般格式为：[T,Y]=solver(odefun,tspan,y0)
    - 该函数表示在区间tspan=[t0,tf]上，用初始条件 y0 求解显式常微分方程 $y'=f(t, y)$

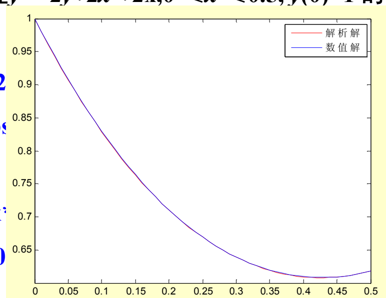
常微分方程求解

- 常微分方程的数值解
  - solver为求解器，不同求解器及其特点说明如下

求解器	特点	说明
ode45	一步算法，4,5阶Runge-Kutta方法，累积截断误差$(\Delta x)^3$	大部分场合的首选算法
ode23	一步算法，2,3阶Runge-Kutta方法，累积截断误差$(\Delta x)^3$	使用于精度较低的情形
ode113	多步法，Adams算法，高低精度均可达到10^{-3}~10^{-6}	计算时间比ode45短
ode23t	采用梯形算法	适度刚性情形
ode15s	多步法，Gear's反向数值积分，精度中等	若ode45失效时，可尝试使用
ode23s	一步法，2阶Rosebrock算法，低精度	当精度较低时，计算时间比ode15s短

常微分方程求解

- 常微分方程的数值解
  - 示例：求解ODE方程$y'=-2y+2x^2+2x, 0 \leq x \leq 0.5, y(0)=1$ 的解
  - y=dsolve('Dy=-2*y+2
  - x=0:0.01:0.5; yy=subs
  - fun=inline('-2*y+2*x
  - [x,y]=ode15s(fun,[0:0
  - plot(x,yy,'r',x,y,'b'),



课外实验三

- 实验名称：MATLAB的数值运算
- 实验目的：通过实验使了解MATLAB数值运算基本流程与主要方法
- 实验内容：
  - 多项式运算（拟合）
  - 插值运算与函数优化
  - 代数方程求解
  - 微分方程求解