附录A　MATLAB初步

这是一个MATLAB的简短介绍。建议在阅读本书的同时使用介绍的命令。若要详细地了解有关命令，请参见每一节的内容和/或使用*help*或*helpdesk*。所有的命令见附录D和命令列表。

A.1　启动和退出MATLAB

根据使用的计算机，单击图标或输入 `matlab`就可以启动MATLAB。详细内容见2.1节。以下信息出现在MATLAB的命令窗口中：

```
                    < M A T L A B (R) >
(c) Copyright 1984-98 The MathWorks, Inc.
                    All Rights Reserved
                    Version 5.2.0.3084
                        Jan 17 1998


To get started, type one of these: helpwin, helpdesk, or demo.
For product information, type tour or visit www.mathworks.com.
```

使用这些命令是非常好的。命令 `tour`和`demo`也是非常有实用价值的。在MATLAB提示符后键入一个命令，当按下回车键后，就执行该命令。

```
>> tour ↩
```

在本书的其他节将不给出MATLAB提示符，因为这将使读者阅读起来比较困难。命令应在提示符>>后键入，但提示符不在每一行中列出。

注意，本书使用不同的字体以区别用户在提示符处输入的命令和 MATLAB返回的运行结果。见1.2节。

退出MATLAB，只需输入`quit`并回车即可。从这以后将不强调在命令行后输入回车键。

```
>>quit
```

如果想要终止MATLAB的运行，就要同时按下‘ CTRL ’和‘ c ’键。MATLAB将停止其运行的所有工作，并且在屏幕上给出提示符，等待输入。

```
>>
```

A.2　基本赋值和计算

通常，MATLAB能被当作计算器使用：

```
5011 + 13


ans =
        5024
```

同一行上可以有多条命令：

```
2^5, 2*(3+2)

ans =
    32

ans =
    10
```

通常，变量用于保存所赋的值和结果。如果没有赋值， MATLAB将结果存放在名为*ans*的变量中。现在定义变量并赋值：

```
x = 14

x =
    14

y = 3*x
y =
    42
```

所有的基本数学函数在MATLAB中有定义(见2.4节)：

```
sin(x)

ans =
    0.9906
```

圆括号’()’，可在数学式子中使用。

```
u = 2*x - y;
w = 2*(x-y);
exp((2-u)/(w-2))

ans =
    0.7589
```

注意，在命令行尾的分号‘ ; ’是MATLAB‘ quietly ’执行赋值命令，即在屏幕上不回显信息，但计算照常执行。

MATLAB中的变量通常为向量或矩阵：

```
vcol = [1;2;3;4], vrow = [5 6 7 8]

vcol =
     1
     2
     3
     4

vrow =
     5     6     7     8

A = [1 2 3 4;5 6 7 8;9 10 11 12]
```

```
A =
     1     2     3     4
     5     6     7     8
     9    10    11    12
```

注意，各行要用分号隔开。

在单个命令中，函数可用于向量或矩阵。

```
sqrt(vcol)

ans =
    1.0000
    1.4142
    1.7321
    2.0000
```

到现在为止，已经定义了许多变量。输入以下命令，可以得到变量列表：

```
who

Your variables are:

A          u          vrow       x
ans        vcol       w          y
```

命令whos也将显示当前的变量，同时还显示出每个变量的其他信息。试使用这个命令，并看看如何区别变量是标量还是向量。

MATLAB在程序运行过程中保存所有的变量，清除变量应输入：

```
Clear
```

先前的变量现在全被清除。此时，如果输入 who，将不会返回任何信息；见 2.3节。向量可通过使用元素操作运算符来生成；见 4.3节。

```
vector = 0:8

vector =
     0     1     2     3     4     5     6     7     8

vector2 = 0:0.5:2

vector2 =
         0    0.5000    1.0000    1.5000    2.0000
```

命令linspace和logspace也可用来创建向量；见 4.3节。通过使用双操作符向量也可直接计算。

```
values = 2.^vector

values =
     1     2     4     8    16    32    64   128   256
```

注意，先前使用的运算符 ^，表示应对向量中的每一个元素进行操作。见 3.5节。

借助箭头键，能重复先前所给的命令；见 2.1节。如果输入有误，它就能避免再写过长的

表达式，这样能节省很多时间。

将向量或矩阵放入括号中能定义一个新的表达式，但是大小必须匹配：

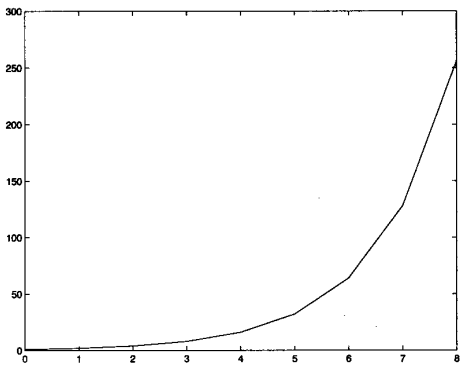
```
Table = [vector;vector.^2;vector.^3]

Table =
     0     1     2     3     4     5     6     7     8
     0     1     4     9    16    25    36    49    64
     0     1     8    27    64   125   216   343   512
```

A.3 简单图形

MATLAB对于处理简单图形和高级图形都是一种非常好的工具。在 MATLAB中画图形，其数据必须存储在向量或矩阵中。例如，画出 2的乘幂的图形有三步。首先，创建一个有值的向量。第二，用这些值对函数求值。第三，画出向量图形；见 13.1节。因为已经在A.2节中创建了变量vector 和values，所以就能直接使用以下命令来画图形：

```
plot(vector, values)
```

结果是一个简单图，其刻度是自动给定的；见图 A-1。



图A-1 函数2^x的分段线性图形

为了使图形更加光滑，应使用更多的值，例如使用 0～8之间的100个数值。linspace命令对于创建长向量是非常有效的，输入：

```
vector = linspace(0,8,100);
values = 2.^vector;
clf;
plot(vector,values);
```

用命令clf在画图之前清除图形窗口。现在试着使用不同的线型。给出命令 plot，及额外的参数‘：’或‘+’。这些能结合颜色；例如，用兰色的点划曲线：

```
plot(vector, values,'b-.');
```

其他线型和颜色可以在13.1节，特别是在表13-1中找到。

三维图形可用相同的方法画出。例如，使用 surf和mesh命令，生成数据放入矩阵中并画出图形。当在画带有两个变量的函数图形时，生成所需的数据可稍微灵活些。这些值或网

格点在平面上是离散的点。生成这些网格最好的方法是使用命令 meshgrid，参见13.4节。

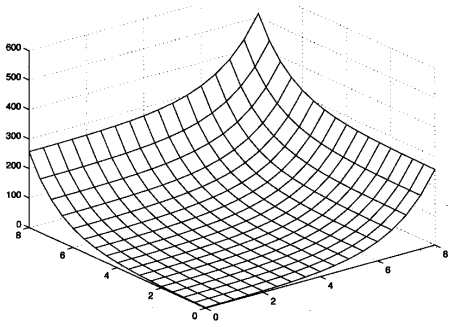
```
vector2 = 0:0.5:8;
[X,Y] = meshgrid(vector2);
mesh(X,Y,2.^X+2.^Y);
```

括号‘ [] ’用于接收多个返回参数。分号‘ ；’用来控制结果的回显。如果是大矩阵，就有必要记住这些。给出的网格图形如图 A-2所示。

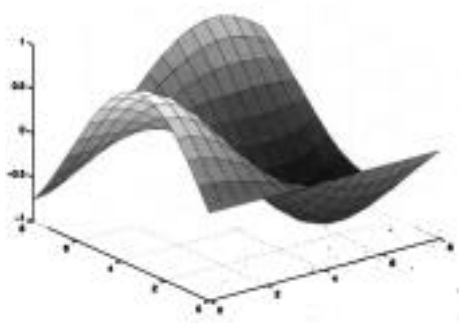
曲面图形和网状图形非常相似，画出的是实际的曲面而不是网格线，见图 A-3。曲面图形是用命令surf生成的。

```
surf(X,Y,cos(X./2).*sin(Y./2));
```

结果如图A-3所示。



图A-2 函数2^x+2^y的网状图形

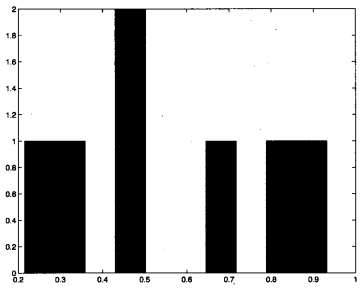


图A-3 使用surf 命令生成函数$\cos(x/2)\sin(y/2)$的图形曲面

画统计图表应使用命令 hist，见6.5节。举例说明，用命令 rand生成元素值在区间 [0, 1]中的随机向量， rand命令见4.1节。

```
random = rand(1,7)

random =
    0.4553 0.3495 0.4523 0.8089 0.9317 0.6516 0.2152

hist(random);
```

hist给出的结果如图A-4所示。试着使用别的命令，如stairs和bar，见6.5节。



图A-4 频数统计直方图

A.4 线性系统和矩阵特征值

定义矩阵：

$$\mathbf{A}=\begin{pmatrix}3 & 4 & 5\\ 5 & 2 & 2\\ 1 & 2 & 3\end{pmatrix} \qquad \mathbf{B}=\begin{pmatrix}1 & 2 & 3\\ 1 & 1 & 1\end{pmatrix}$$

用下面的语句：

```
A = [3 4 5;5 2 2;1 2 3];
B = [1 2 3;1 1 1];
```

用命令B*A来做矩阵相乘，结果如下：

```
ans =
    16    14    18
     9     8    10
```

但是A*B的结果为：

```
??? Error using ==> *
Inner matrix dimensions must agree.
```

因为这种矩阵的乘法没有定义，见3.2节。

计算A的行列式值，可使用命令det:

```
det(A)

ans =
    -6
```

B的行列式值没有定义。也可试着对同一矩阵使用 trace、null、orth和inv命令，这些命令在7.1节有定义。

下面来看看线性方程组：

$$\begin{aligned}3x_1+4x_2+5x_3&=25\\ 5x_1+2x_2+2x_3&=18\\ x_1+2x_2+3x_3&=13\end{aligned}$$

或者是矩阵表达式Ax=b，A是上面定义的，b如下：

$$\mathbf{b}=\begin{pmatrix}25\\ 18\\ 13\end{pmatrix}$$

可用反斜杠运算符‘\’来求解方程组。

```
b = [25;18;13];
A\b

ans =
    2.0000
    1.0000
    3.0000
```

可以用同样的方法来解超定方程组。因为通常这些方程组没有真解， MATLAB用最小二

乘法来解。见7.2和7.7节。

特征值和特征向量可由命令eig求得。例如，对上面的矩阵**A**求：

```
[EigenVectors,EigenValues] = eig(A)

EigenVectors =
   -0.7111   -0.4501   -0.0210
   -0.6185    0.8459   -0.7756
   -0.3342   -0.2863    0.6309

EigenValues =
    8.8291         0         0
         0   -1.3373         0
         0         0    0.5082
```

矩阵**EigenVectors**的列就是**A**的特征向量，矩阵**EigenValues**的对角线上的元素就是A的特征值；见8.1节。

A.5 曲线拟合及多项式

多项式可由系数矩阵来表示；见10.1节。多项式：

$p(x) = 2x^3 + x^2 + 5x + 17$

可用向量**p**=(2 1 5 17)来表示，而且可用polyval命令对任何值进行求值：

```
p = [2 1 5 17];
polyval(p,0), polyval(p,2)

ans =
    17

ans =
    47
```

用命令polyder可对多项式进行微分，并且用命令conv对其进行乘法运算。

```
pprim = polyder(p)

pprim =
     6     2     5
```

这代表$p'(x)=6x^2+2x+5$。

```
psquare = conv(p,p)

psquare =
     4     4    21    78    59   170   289
```

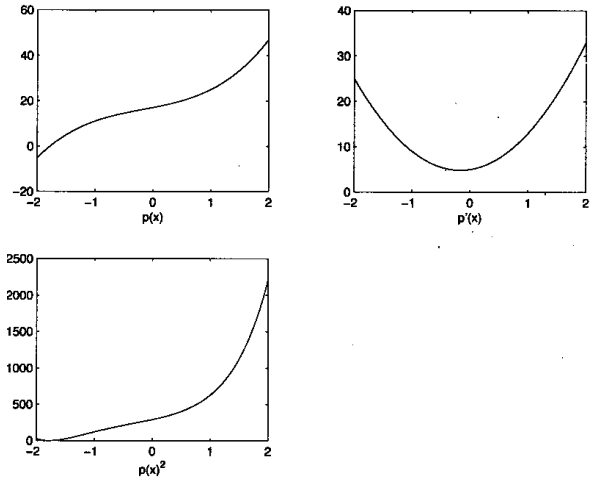
这代表$p(x)^2=4x^6+4x^5+21x^4+78x^3+59x^2+170x+289$。

下面在同一个图中用三个子图画出这三个多项式。命令 subplot见13.3节。

```
x = linspace(-2,2,50);
```

```
clf;
subplot(2,2,1);
plot(x,polyval(p,x));
xlabel('p(x)');

subplot(2,2,2);
plot(x,polyval(pprim,x));
xlabel('p''(x)');

subplot(2,2,3);
plot(x,polyval(psquare,x));
xlabel('p(x)^2');
```

命令xlabel可在当前图形中x轴下方写一字符串；如图A-5所示。



图A-5 多项式$p(x)=2x^3+x^2+5x+17$及其微分和多项式$p(x)^2$的图形

多项式可用来拟合数据曲线。假设以下数据在一次实验中获得，并且输入到 MATLAB中：

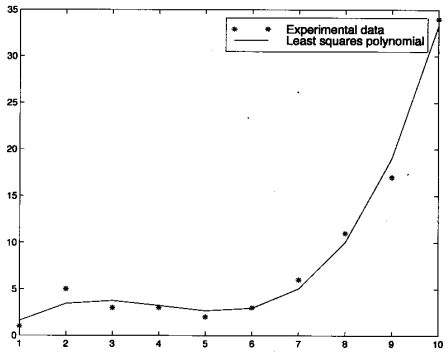
```
x = 1:10;
y = [1 5 3 3 2 3 6 11 17 34];
```

命令polyfit可返回多项式，也就是多项式的系数，该多项式表示对指定的次数拟合最好的最小二乘曲线；见 10.4节。例如此例，最好的 4次多项式已经找到。同时，原始数据点也在多项式的同一图中画出。

结果如图A-6所示。

```
clf;
ypol = polyfit(x,y,4);
plot(x,y,'*',x,polyval(ypol,x),'b-');
legend('Experimental data','Least squares polynomial');
```

注意，有些向量可用相同的 plot命令画出彼此的连线图形。命令 legend给出一个含有对不同向量表示图解释的工具包。此命令定义在 13.3节中。对不同次数的多项式和不同的数据集合使用polyfit和polyval命令。



图A-6 数据集合和拟合的4次多项式

内插值法和外插值法可用不同的 `interp`命令来实现。若想对 x=4.3在上面的数据集合中内插求值，可输入：

```
interp1(x,y,4.3)

ans =
    2.7000
```

这是线性内插值法，也可看成是‘向上查阅表’。当然，其他可能性也是存在的，输入`help interp`来看看。内插值法定义在10.4节中。

A.6 简单的程序

一段程序是MATLAB语句的序列，它们由条件语句和循环语句来控制。在 M文件中方便地就能保存这些命令。其中M文件是后缀名为**.m**的文本文件。

下面来看看保存在**draw.m**文件中的MATLAB语句：

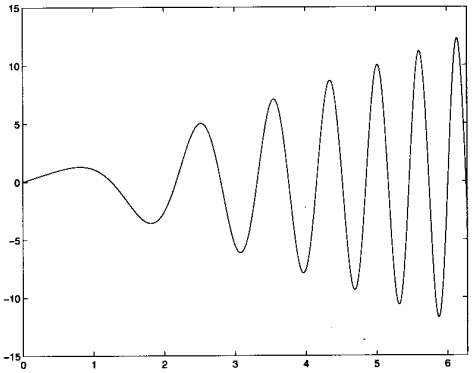
```
% 绘制函数图形的程序
% 在%标记后做注释

% 显示解释文本

disp('This program plots f(x) in the interval [a,b]');

% 从用户读入数据

ftext = input('Give a function: ','s');
a     = input('Give lower bound a: ');
b     = input('Give upper bound b: ');

clf; fplot(ftext,[a b]);        % 绘制该函数图形
```

在MATLAB提示符后输入`draw`便可运行该程序。例如下列情况：

```
This program plots f(x) in the interval [a,b].
Give a function: 2.*x.*cos(x.^2)
Give lower bound a:  0
```

```
Give upper bound b:  2.*pi
```

根据上面的输入，程序运行的结果如图 A-7所示。



图A-7 用M文件draw绘制函数$2x\cos x^2$的图形

程序draw.m是一个命令文件的例子。还有函数文件，它的后缀名也必须是 .m，并且在第一行有关键字function。现在举个例子，下面的函数保存在 M文件div.m中：

```
function y = div(n,d);

% 该函数用来计算整数的除法
% 因此 n= rem(n, d)+y*d

y = (n-rem(n,d))/d;
```

该函数可以作为MATLAB中的一个函数来被调用：

```
div(1234,7), check = 1234/7


ans =
   176

check =
  176.2857
```

有关M文件的信息见2.9 和12.3节。

A.7 函数分析

可用命令fmin来寻找单变量函数的局部最小值，对于多变量函数可用命令 fmins；见10.3节。

求函数的最小值：

$$f(x) = \frac{x - 1.96}{x^2 + 1.15}$$

首先要创建一个计算函数$f(x)$值的文件。该文件取名为f.m。

```
function y = f(x)

y = (x-1.96)./(x.^2+1.15);
```

命令：

```
fmin('f',-1,1)
```

给出的结果如下：

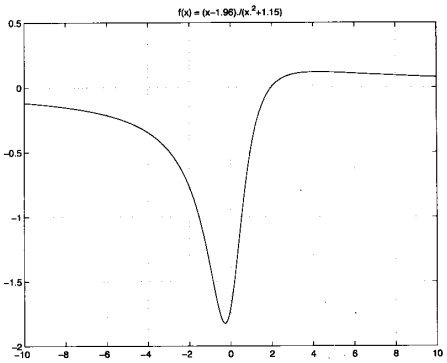
```
ans =
   -0.2742
```

用下面给出的命令画出函数图形，就能知道所得的结果是否正确：

```
fplot('f',[-10 10]);
title('f(x) = (x-1.96)./(x.^2+1.15)');
grid;
```

title命令可给出图形的标题，此命令在13.3节中介绍。命令grid可给图形加上网格，它的介绍也在13.3节中。也能找到0值的准确位置，即对$f(x)=0$成立的x值。这是用命令fzero来求的，它需要一个迭代的初始值(见10.2节)。在图A-8中，可以看到2接近于$f(x)=0$，于是就输入：

```
fzero('f',2)

ans =
    1.9600
```



图A-8 用fplot绘制的M文件函数图形

A.8 积分

可以使用11.1节中介绍的命令quad来计算定积分。计算：

$$\int_0^{\pi} \frac{\sin(x)}{x}\,dx$$

必须在名为sinx_x.m的M文件中定义函数$\sin(x)/x$：

```
function y = sinx_x(x)

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

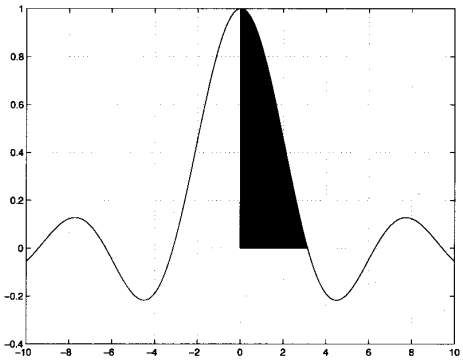
现在只需输入：

```
quad('sinx_x',1E-8,pi)
```

给出的结果为：

```
ans=
1.8520
```

注意，该积分不能被解析。函数图形如图 A-9所示。



图A-9　在区间－10≤x≤10内函数sin(x)/x的图形

标记的区间为积分值。这个区域使用了定义在 14.2节中的图形对象patch来填充。首先用fplot命令画出函数sinx_x的图形，并加上网格。

```
fplot('sinx_x',[-10 10]); grid on;
```

现在给出patch命令，并且保留一个‘句柄’，以便稍后能改变补片的属性。

```
h = patch([0.001 0.001:0.01:pi pi], ...
    [0 sinx_x(0.001:0.01:pi) 0], 'r' )


h =
   73.0005
```

patch的第一个参数定义了图形各边的x坐标。第二个参数定义了y坐标。最后一个参数定义了画图使用的颜色，这里的‘r’表示红色。现在可以输入下面的命令来改变补片的边颜色：

```
set(h,'EdgeColor',[0 0 1]);
```

要想查看由set命令设置的其他属性，可输入get(h)。这些命令在14.2节中介绍。

A.9　普通微分方程

假设现在有一个存放在文件xprim.m中的函数xprim，并定义如下：

```
function y = xprim(x,t)

y = (x-1.96)./(x.^3+1.15);
```

除了x，还有t作为输入参数，虽然函数值并不与 t有关。这只是因为要求解下面的常微分方程：

$$\frac{dx}{dt} = \frac{x - 1.96}{x^3 + 1.15}$$

用定义在11.2节中的MATLAB命令ode45来解。

先在区间0≤t≤10内、初始值为x(0)=1来求解(*)，调用ode45命令如下：

```
[x1,t1]=ode45('xprim',[0 10],1);
```

结果是两个向量**x1**和**t1**，并绘制出它们的图形：

```
clf;
plot(x1,t1);
```

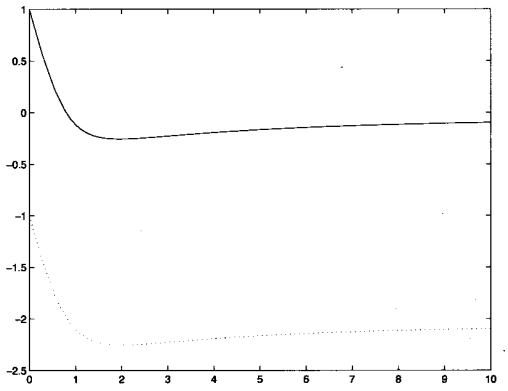
现在用相同的t区间、不同的初始值$x(0)=-1$来求解方程$(*)$：

```
[x2,t2]=ode45('xprim',[0 10],-1);
```

将这个函数图形绘制在同一个图中，这样就要在绘制新的向量图形之前给出`hold on`命令：

```
hold on;
plot(x2,t2,':');
hold off;
```

结果如图A-10所示。



图A-10　不同初始值的常微分方程的解的图形