间的运算．点运算要求参与运算的变量在结构上必须是相似的．

例1　>>a=[1,2,3；4,5,6；7,8,9]

```
a =
     1     2     3
     4     5     6
     7     8     9
>> b=[1 2 3
   4 5 6
   7 8 9]
b =
     1     2     3
     4     5     6
     7     8     9
```

```
>> c=a*b
c =
    30    36    42
    66    81    96
   102   126   150
>> d=a.*b
d =
     1     4     9
    16    25    36
    49    64    81
```

例2　>>a=[1:3；4:6；7:9]；

```
>>x=5；

>>y=ones(3)*5；

>>xa=x<=a
xa =
     0     0     0
     0     1     1
     1     1     1

>>b=[0 1 0 ;1 0 1 ;0 0 1]；
```

```
>>ab=a&b
ab =
     0     1     0
     1     0     1
     0     0     1
>>nb=~b
nb =
     1     0     1
     0     1     0
     1     1     0
```

例3　>>a=magic(5)；

```
>>a(:,3)=zeros(5,1)
a =
    17    24     0     8    15
```

```
>>a2=all(a>3)
a2 =
     1     1     0     0     0
>>a11=any(a(:,1)>10)
```

```
    23     5     0    14    16        a11 =
    4      6     0    20    22              1
    10     12    0     21    3     >>a22=any(a>10)
    11     18    0     2     9        a22 =
>>a1=all(a(:,1)<10)                    1    1    0    1    1
  a1 =
       0
```

6）字符串

在 MATLAB 中的字符串一般是 ASCII 值的数值数组，它作为字符串表达式显示出来．字符串用单引号输入或赋值；字符串的每个字符都是字符数组的一个元素；字符串和字符数组基本等价．字符串的每个字符（包括空格）都是字符数组的一个元素，字符串的相关操作见表 1-7．例如：

```
>> s='i love you'                    >> s(2)
    s =                                 ans =
     i love you                     >> s(10)
>> size(s)                             ans =
    ans =                                  u
       1      10
```

表 1-7　字符串的相关操作

函数名	相关操作	函数名	相关操作
strcat	链接字符串	strvcat	垂直链接字符串
strcmp	比较两个字符串是否相等．当相等时，系统将返回值 1，不相	strncmp	比较两个输入字符串的前几个字符是否相等．当相等时，系统将

	等时，返回值 0		返回值 1，不相等时，返回值 0
findstr	在其他的字符串中寻找该字符串	strjust	证明字符数组
strmatch	查找可能匹配的字符串	strrep	用其他字符串代替该串
strtok	查找字符串中的记号	blanks	生成空的字符串
deblank	删除字符串内的空格	ischar	字符串检验
iscellstr	字符串的单元检验	isletter	字母检验
isspace	空格检验	strings	strings 函数的帮助
upper	转换串为大写	lower	转换串为小写
double	字符串转换为数值代码	num2str	数字转换为字符串
int2str	整数转换为字符串	mat2str	矩阵转换为字符串

1.1.2　MATLAB 的矩阵运算

1）矩阵的表示

方式 1：直接输入小矩阵（最简便的方法）. 例如，矩阵 $\boldsymbol{A}=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\end{bmatrix}$ 可以这样输入：

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

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

需要注意如下几点：

（1）输入矩阵时要以“[ ]”为其标识，即矩阵的元素应在“[]”内部．

（2）矩阵的同行元素之间可由空格或“ ,”分隔，行与行之间用“ ;”或回车符分隔．

（3）矩阵元素可为运算表达式．无任何元素的空矩阵也合法．

方式 2：利用中括号将小矩阵合成一个大矩阵．例如：

```
>> b=[1,2 ; 3,4] ;

>> a=[b,b' ; b^2,b*3]

a =
     1     2     1     3
     3     4     2     4
     7    10     3     6
    15    22     9    12
```

方式 3：从外部文件（EXCEL、TXT 等）引用矩阵．

除了以上的三种表示方法外，对于特殊矩阵，可以用如下命令（表 1-8）进行自动生成．

表 1-8　特殊矩阵的生成表

命　令	说　明
A=[ ]	空矩阵
A=eye(n)	*n* 维单位矩阵
A=ones(n,m)	元素全为 1 的矩阵
A=rand(n,m)	元素服从 0 和 1 之间均匀分布的随机矩阵
A=rand(n,m)	元素服从零均值单位方差正态分布的随机矩阵
A=zeros(n,m)	元素全为 0 的矩阵

例如：

```
>> eye(2,3)
 ans =
     1     0     0
     0     1     0
>> zeros(2,3)
 ans =
     0     0     0
     0     0     0
>> ones(2,3)
 ans =
     1     1     1
     1     1     1
```

```
>> eye(2)
 ans =
     1     0
     0     1
>> zeros(2)
 ans =
     0     0
     0     0
>> ones(2)
 ans =
     1     1
     1     1
```

2）矩阵的操作

矩阵大小操作：MATLAB 可以用表 1-9 中的命令来对一个矩阵的大小进

行查询．例如：

```
>>A = [1,2 ;  4,5 ;  7,8]
A =
     1     2
     4     5
     7     8
>>[n,m]=size(A)
  n=
     3
  m=
     2
>>a=length(A)
   a=
        3
>>[i,j]=find(A>6)
      i=
        3
        3
      j=
        1
        2
```

矩阵的块操作：MATLAB 可以对一个矩阵进行元素的更改、插入子块、提取子块、重排子块、扩大维数等操作．例如：

```
>>A = [1,2 ;  4,5 ;  7,8]
  A =
     1     2
     4     5
     7     8
>>B =[3,6]
  B=
     3    6
>>A(2,:)=B
  A=
     1     2
     3     6
     7     8
>>A(:,:)=9
A=
     9     9
     9     9
     9     9
>>A(3,2)=10
  A=
     9     9
     9     9
     9    10
```

常见的矩阵操作见表 1-9．

表 1-9　矩阵的相关操作

命令	操作结果
whos	显示工作空间中存在的变量及其大小
size(A)	返回矩阵 A 的行数和列数
length(A)	返回矩阵 A 的最大行、列数
find(A)	给出特殊要求的矩阵元素的行、列标记
A(i,j)	矩阵 A 的第 i 行、第 j 列的元素
A(r, :)	矩阵 A 的第 r 行
A(:,r)	矩阵 A 的第 r 列
A(:)	依次提取矩阵 A 的每一列，将 A 拉伸为一个列向量
A(i1:i2, j1:j2)	取矩阵 A 的第 i_1~i_2 行、第 j_1~j_2 列构成新矩阵
A(i2:-1：i1,:)	以逆序提取矩阵 A 的第 i_1~i_2 行，构成新矩阵
A(:,j2:-1:j1)	以逆序提取矩阵 A 的第 j_1~j_2 列，构成新矩阵
A(i1:i2,:)=[]	删除 A 的第 i_1~i_2 行，构成新矩阵
A(:,j1:j2)=[]	删除 A 的第 j_1~j_2 列，构成新矩阵

3）矩阵的运算

矩阵的运算包括矩阵与标量、矩阵与矩阵、矩阵的逆、矩阵的转置和矩

阵函数的计算等．例如：

```
>>A = [1,2,3 ;  4,5,6]
   A =
      1     2     3
      4     5     6
>>C = 2*A
   C =
      2     4     6
      8    10    12
>>D = B-A
   D =
      1     2     3
      4     5     6
```

```
>>B= [2,4 ;  1,5]
   B=
      2     4
      1     5
>>C=B^2
   C =
      8    28
      7    29
>>D=B^(-1)
   D=
    0.8333   -0.6667
   -0.1667    0.3333
```

```
>>A = [1,3,5 ;  2,4,6]
   A =
      1     3     5
      2     4     6
>> B= [1,2 ;  3,4]
   B=
      1     2
      3     4
>>C= B\A
   C=
      0   -2.0000   -4.0000
 0.5000    2.5000    4.5000
```

```
>>C=[1,1,3 ; 1,2,3 ; 4,5,6]
   C=
   1     1     3
   1     2     3
      4     5     6
>>D= A/C
   D=
0.000 0    2.3333   -0.3333
   0       2.0000        0
>>D=C'
   D=
      1     1     4
      1     2     5
      3     3     6
```

常见的矩阵函数运算见表 1-10．

表 1-10　矩阵函数运算

命令	运算结果	命令	运算结果
a=eig(A)	矩阵特征值	orth(A)	正交化
[v,a]=eig(A)	矩阵特征值与特征向量	pinv(A)	伪逆
det(A)	矩阵的行列式	poly(A)	矩阵的特征多项式
expm(A)	矩阵求幂	schur(A)	矩阵的分解
inv(A)	矩阵的逆	sqrtm(A)	矩阵的平方根
logm(A)	矩阵的对数	svd(A)	矩阵的奇异值分解
norm(A)	矩阵的范数	trace(A)	矩阵的对角元素之和
norm(A,1)	矩阵的 1-范数	diag(A)	提取矩阵的对角元素 ,返回列向量
norm(A,2)	矩阵的 2-范数	tril(A)	提取矩阵的下三角矩阵
norm(A,inf)	矩阵的无穷范数	triu(A)	提取矩阵的上三角矩阵
norm(A,p)	矩阵的 p-范数	flipud(A)	将矩阵进行上下翻转
norm(A ,'fro')	矩阵的 F-范数	diag(V)	以列向量 V 作对角元素创建对角矩阵

null(A)	零空间	fliplr(A)	将矩阵进行左右翻转

例如：

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

   A =
         1     2     3
         4     5     6
         7     8     9
>>B= flipud(A)                    >>C= fliplr(A)
   B =                               C=
     7     8     9                      3     2     1
     4     5     6                      6     5     4
     1     2     3                      9     8     7
```

1.1.3　MATLAB 中的 M 文件

M 文件就是用 MATLAB 语言编写的可在 MATLAB 语言环境下运行的程序源代码文件．MATLAB 语言中的 M 文件可分为命令式（script）和函数式（function）两种形式．M 文件可在 MATLAB 的程序编辑器中编写，也可在其他的文本编辑器中编写，并以“.m”为扩展名加以保存．

1）命令式 M 文件

命令式文件就是命令行的简单叠加，MATLAB 会按顺序自动执行文件中的命令．值得注意的是，命令式文件在运行过程中可以调用 MATLAB 工作域内所有的数据，而且所产生的所有变量均为全局变量．

例如，将 a=2；b=[3 4 5 6]；c=a+b；d=b-a；保存为 abcd.m，然后在工作窗口中输入调用命令：

```
>>abcd
>> c
   c =
        5     6     7     8
>> d
   d =
        1     2     3     4
```

2）函数式 M 文件

函数式 M 文件的第一行必须从特殊字符 function 开始，格式为：

```
function    [y1,y2, …]=fun(x1,x2, …)
```

其中 fun 是函数名，要遵守 MATLAB 变量名的命名规则； x1,x2, …是输入变量； y1,y2, …是输出变量，从函数返回的唯一信息包含在输出参数中，要确保函数中包含一条给输出参数赋值的语句．

例如，将下面的函数保存为 abcd.m：

```
function abcd(a,b)
c=a+b
d=b-a
```

然后在工作窗口中输入调用命令：

```
>>a=2；b=[3,4,5,6]；
>>abcd(a,b)
   c =
```

```
     5     6     7     8
d =
     1     2     3     4
```

函数式 M 文件建立步骤如下：

（1）在 MATLAB 中，点击：File→New → M-file；

（2）在编辑窗口中输入程序内容；

（3）点 File → Save，存盘，M 文件名必须与函数名一致．

函数式 M 文件有自己的工作空间，与 MATLAB 的工作空间是分开的，M 文件的变量都是内部变量，不会被送到工作空间去．函数式 M 文件与工作空间的联系只是输入和输出变量；如果函数式 M 文件中含有 return 命令，那么函数将中断运行，返回工作空间．MATLAB 在执行一次函数式 M 文件之后，会将其编为机器码，下一次运行时就直接调用，故其运算速度很快．函数式 M 文件中可以调用其他一般 M 文件（或称为脚本文件），这时脚本文件中的变量都只在函数式 M 文件中有效，在 MATLAB 的工作空间中无效．

MATLAB 中的 M 文件的功能非常强大．它可以自由编写复杂的程序，通过调用各种内部函数、其他 M 文件等，完成复杂的数值、逻辑和符号运算，是一个非常有用的工具．

例 4　用一个简单的 M 文件来计算 $y=2(\sqrt{x^2+10}+z_1-z_2)-5$，其中 z_1, z_2 为

全局变量.

解　首先将如下程序用 f1.m 文件名保存为 M 文件：

```
global z1 z2
x=1:5
z1=1:-0.1:0.6
z2=0:0.5:2
y=f1(x)
```

为了运行上面的 M 文件，我们再建立一个如下程序的 M 函数文件 f2.m：

```
function[p]=f2(x)
global z1 z2

n=length(x)；

for i=1:n

        p1(i)=sqrt(x(i)^2+10)+z1(i)-z2(i)；

end

p=p1*2-5；
```

最后在 MATLAB 的工作空间输入命令：

```
>>f1
    x=
      1   2   3   4   5
    z1=
        1.00        0.9000   0.8000   0.7000   0.6000
    z2=
        0           0.5000   1.0000   1.5000   2.0000
    y=
        3.6332   3.2833   3.3178   3.5980   4.0322
```

3）函数变量及变量作用域

M 文件的变量主要有输入变量、输出变量及函数内部变量．输入变量相当于函数的入口数据，是一个函数操作的主要对象．函数的作用就是对输入变量进行加工以实现一定的功能．函数的输入变量为形式参数，即只传递变量的值而非变量的地址，函数对输入变量的一切的操作和修改如果不依靠输出变量传出的话，将不会影响工作空间中该变量的值．

MATLAB 语言提供了函数 nargin 来控制输入变量的个数，即可以实现不定参数输入的操作．例如，在函数 test1 中，如果调用过程时只提供一个输入变量，则求该输入变量的模；如果是两个输入变量，则求两个输入变量的和．

如果调用过程时只提供一个输入变量，则求该输入变量的模；如果是两个输入变量，则求两个输入变量的和．

```
function c=test1(a,b)
  if nargin==1
    c=norm(a)；
  elseif nargin==2
    c=a+b；
  end
```

在工作窗口调用：

```
>> a=[2 3 4]
  a =
    2     3     4
>> test1(a)
```

```
    ans =
        5.3852
>> b=3；
>> test1(a,b)
    ans =
        5        6        7
```

同时，MATLAB 语言还提供了另一个针对输入变量的函数 varargin，该函数可以实现不定数目输入变量的函数的程序设计．此时，将函数的一切输入变量均存储在以 varargin 命名的单元型数组中．

例 5　在函数 test2 中，实现如下功能：通过使用函数 varargin，用户可以输入任意多个学生的数学、英语及语文的成绩，然后求各科目的平均值．

解

```
function [mathavg,englishavg,chineseavg]=test2(varargin)
    s=length(varargin)；
    mathsum=0；
    englishsum=0；
    chinesesum=0；
    for j=1:s
        mathsum=mathsum+varargin{j}(1)；
        englishsum= englishsum +varargin{j}(2)；
```

```
        chinesesum= chinesesum +varargin{j}(3)；
      end
    mathavg=mathsum/s；
    englishavg=englishsum/s；
    chineseavg=chinesesum/s；
```

与输入变量相对应，MATLAB 语言对输出变量也提供了相应的函数，如 nargout、varargout 等．具体的使用与函数 nargin 和 varargin 相似．例如：在函数 test3 中，综合使用了 nargin、nargout、narargin、narargout 等函数，其目的是求各学生（总数不确定）的个人平均成绩以及指定科目的平均成绩等．函数 test2 的调用：[90,89,60]分别表示数学、英语、语文的成绩．

```
>> [a,b,c]=test2([90,89,60],[79,89,66],[99,98,100])
    a =
        89.3333
    b =
        32.6667
    c =
        33.3333
function [vararout]=test3(lessons,varargin)
 inputnum=nargin；
 lessonnum=length(lessons)；
```

```
outputnum=nargout；
for i=1:lessonnum
    switch lessons(i)
    case 'math'
        varargout{1}=sum(varargin{1:inputnum}(1))；
    case 'english'
        varargout{2}=sum(varargin{1:inputnum}(2))；
    case 'chinese'
        varargout{3}=sum(varargin{1:inputnum}(3))；
    end
  end
  for i=1:inputnum
  varargout{i+3}=sum(varargin{i}(:))；
end
```

MATLAB 语言也可以定义子函数，用来扩充函数的功能．在函数文件中题头定义的函数为主函数，而在函数体内定义的其他函数均被视为子函数．子函数只能被主函数或同一主函数下其他的子函数所调用．局部函数与子函数的区别是：局部函数可以被其父目录下的所有函数所调用，而子函数则只能被其所在 M 文件的主函数所调用；在函数编辑的结构上，局部函数与一般的函数文件的编辑相同，而子函数则只能在主函数文件中编辑．

例如：

```
function c=test(a,b)
```

```
c=test1(a,b)*test2(a,b)；
function c=test1(a,b)
c=a+b；
function c=test2(a,b)
c=a-b；
```

MATLAB 中的用户交互函数主要指键盘输入函数，它包括函数 input 及 keyboard．

函数 input 用于提示用户输入指定参数的值，其调用格式为：

var=input(‘提示性语句’)，

其中：“提示性语句”将给出相应的提示信息以告知用户输入的对象，在这种调用过程中，用户可以由键盘输入任何的可计算的表达式或已赋值的当前工作空间中的变量名，而且返回至变量 var 中的值也是数值型，所有的输入以回车键予以确认．

函数 input 的另一种调用格式为：

var=input(‘提示性语句’,‘参数’),

以该格式调用时，input 函数将视用户键盘输入的一切字符为字符型赋予变量 var，而不对其进行任何计算．在提示性语句中可以用“\n”来控制显示时的换行，‘参数’定义输入数据的格式．例如:

第一种调用格式：

```
>> a=input('input example\n')
   input example
   1+4
   a =
      5
>> ischar(a)
   ans =
        0
```

第二种调用格式：

```
>> b=input('input example\n','s')
     input example
     1+4
     b =
        1+4
     >> ischar(b)
        ans =
             1
```

keybroad 函数出现在 M 文件中，将终止程序的运行，此时用户可以查看运算过程中各变量的值，必要时也可进行适当的干涉及编辑．同时该函数在 MATLAB 语言中均可应用，直到键入 return 并回车，这时程序返回中断处，继续执行其余代码．当然 MATLAB 也提供了另一种中断函数 pause，其调用格式为：

pause (n),

该命令的作用是中断程序运行并等待 n 秒．如果不带参数，则程序无限期中断，直到用户在键盘上键入任意键．

1.1.4　流程控制语句

MATLAB 语言的流程控制语句主要有 for、while、if-else-end 和 switch-case 这四种语句．

1）for 语句

for 循环语句是流程控制语句的基础，使用该循环语句可以以指定的次数重复执行循环体内的语句．for 循环语句的调用形式为：

for 循环控制变量=<循环次数设定>

循环体

end

例如：

```
for i = 1:4
    for j = 4:-1:1
        H(i, j) = 1/(i+j-1)；
    end
end
```

结果:

```
>> H
    H =
        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
```

2）while 语句

while 循环语句与 for 循环语句不同的是，前者是以条件的满足与否来判断循环是否结束的，而后者则是以执行次数是否达到指定值来判断的．while

循环语句的一般形式为：

while <循环判断语句>

循环体

end

其中，循环判断语句为某种形式的逻辑判断表达式．当表达式的值为真时，执行循环体内的语句，否则退出．当循环判断语句为矩阵时，当且仅当所有的矩阵元素非零时，逻辑表达式的值为真．

例 6 求满足不等式 $2^n < 100$ 的最大 n.

解

```
n = 0 ;
while 2^n < 100
        n = n + 1 ;
 end
 N=n-1
```

例 7 计算级数的前 n 项和 $S = 1+2+2^2+2^3+\cdots+2^{63}$.

解

方法一：

```
S=0 ;
for i=0:63
    S=S+2^i ;
end
```

方法二：

```
S=0 ;
i=0 ;
    while i<64
        S=S+2^i ;
        i=i+1 ;
    end
```

方法三：

```
n=0:1:63 ;
S=sum(2.^n)
```

从例 7 中可以看出，while 循环语句与 for 循环语句有着异曲同工之妙．但