# 第5章　字符串和其他数据类型

在MATLAB中可能会遇到对字符和字符串的操作。字符串能够显示在屏幕上，也可以用来构成一些命令，这些命令在其他的命令中用于求值或者被执行。细胞矩阵或者细胞数组是无类型矩阵，它们中的元素可以是任何类型。 MATLAB中还有大量适合于位运算的函数和一些常用的整数函数。还有可能把向量作为集合来看待。

## 5.1　字符串

一个字符串是存储在一个行向量中的文本，这个行向量中的每一个元素代表一个字符。实际上，元素中存放的是字符的内部代码，也就是ASCII码。当在屏幕上显示字符变量的值时，显示出来的是文本，而不是ASCII数字。由于字符串是以向量的形式来存储的，所以可以通过它的下标对字符串中的任何一个元素进行访问。

字符矩阵也可以这样，但是它的每行字符数必须相同。

### 5.1.1　分配

MATLAB中的字符串用单引号来定义：

```
NameOfVariable='text'
```

这里的'text'可以是字母、数字和特殊字符。

■ **例**5.1

(a) 简单的分配方法，如name='John Smith'，在屏幕上就会有如下显示：

```
name =
John Smith
```

(b)如果刚输入'John Smith'，那么就将这个字符串赋给变量 *ans*：

```
ans =
John Smith
```

(c) 分配一个字符。如果(a)中变量**name**已存在，令name(3)='a'，则会给出：

```
name =
Joan Smith
```

(d) 将上例中的字符串**name**的元素前后互换位置，可以输入：

```
for i = length(name):-1:1
  eman(i) = name(length(name)+1-i);
end
eman
```

下面显示出字符串**eman**的值：

```
eman =
htimS naoJ
```

关于循环for-loops语句可参考12.2节。用eman=fliplr(name)也可得到同样的结果；见4.1节。

(e) 一个字符串的长度：namelen=size(name)，给出：

```
namelen =
     1    10
```

(f) 在字符串中用两个单引号来表示一个单引号：

```
whoscat='Joan"s cat'
```

显示结果为：

```
whoscat=
   Joan's cat
```

(g) 字符串的组成可以象数字矩阵一样：

```
name1 = 'Joan'; name2 = 'John'; heart = 'is in love with';...
sentence = [name1,' ',heart,' ',name2]
```

显示的结果为：

```
sentence =
Joan is in love with John
```

也可参见命令集47中的命令strcat和strvcat。

(h) 冒号表达式的使用和在数字矩阵中的使用情况一样：

```
name='Charles Johnson';firstname= name(1:7)
```

给出：

```
firstname=
Charles
```

(i) text1='John'；text2='Joan'；couple=[text1; text2]给出：

```
couple =
John
Joan
```

■

### 5.1.2 字符串命令

有一些命令可用于把字符串转换成其他的表示形式。

**命令集45 转换字符串**

| | |
|---|---|
| abs(str) | 返回一个向量，其元素是字符串**str**中字符的ASCII码值。 |
| char(x) | 根据指定的字符集将向量 **x**中的整数转换成字符。这个命令是命令abs的逆操作。在旧版MATLAB中命令setstr还可用，但将会被去掉。 |
| num2str(f) | 将数值*f*转换成浮点格式的字符串。如果需要，可包含四位数字和指数。这个命令经常和命令 disp、xlabel还有一些其他输出命令一起使用；见 13.3节例13.9。 |
| num2str(f,k) | 将数值*f*转换成带有*k*位数字的浮点格式的字符串。 |
| num2str(f,format) | 将数值*f*转换成由**format**设定格式的字符串， **format**用 |

| | |
|---|---|
| | 在函数sprintf中，见下面的命令。 |
| int2str(n) | 将整数*n*转换成整数字符串表达式。 |
| rats(x,strlen) | 将浮点小数*x*转换成含有对*x*的有理逼近的字符串，整数*strlen*是每个元素的字符串长度，缺省值为13。 |
| hex2num(hstr) | 将字符串**hstr**中的十六进制数转换成相应浮点数(IEEE双精度)。 |
| hex2dec(hstr) | 将字符串**hstr**中的十六进制数转换成相应整数。 |
| dec2hex(n) | 将整数*n*转换成相应的十六进制数字符串。 |
| base2dec(str,base) | 将字符串**str**中从基数*base*开始的元素转换成十进制数。 |
| dec2base(n,base) | 将整数*n*转换成基数*base*。 |
| bin2dec(str) | 将字符串中的二进制数转换成十进制数。 |
| dec2bin(n) | 将整数*n*转换成二进制数。 |
| mat2str(A,n) | 将矩阵**A**转换成字符串。如果给出了*n*值，它就代表正确数字的个数。 |
| str2num(str) | 返回字符串**str**的数字形式，字符串可以包含数字、小数点，开始的符号表示10的幂的*e*，还有复数虚部的*i*。 |
| str2rng(str) | 将一个电子表区域**str**转换成[*R*1 *C*1 *R*2 *C*2]形式的向量，它给出了字符串**str**中指定区域的开始和最后的行和列来规定电子表中的区域。 |
| strjust(str) | 返回一个和**str**一样内容的字符串，但是调整了右边。 |
| sprintf(formatstr,A) | 将矩阵**A**中元素返回到一个格式化字符串中，这个格式是由格式字符串**formatstr**来定义的，和C语言中的格式控制相似。这个命令和命令fprintf的作用是一样的，只是返回的结果是一个字符串而不是一个文件，见15.4节。 |
| [Str,E]= sprintf(.....) | 返回一个上述的字符串和矩阵**E**。如果发生错误，E中就会得到一个错误信息字符串；如果变换正确，就返回空矩阵**E**。 |
| sscanf(str, formatstr,mn) | 返回一个矩阵，它是根据字符串**formatstr**从字符串**str**中读出数据而构成的矩阵。读出元素的最大个数为*mn*，但是这个参数是可选的。这个命令和命令 **fscanf**的作用是一样的，区别在于前者是对一个字符串而不是对文件进行操作，见15.4节。 |
| [A, nm, E, next]= sscanf(str, formatstr,mn) | 得到一个象通过命令sscanf返回的一样的矩阵**A**，也返回正确转换的元素*mn*的个数以及矩阵**E**中的错误的个数。当所有的元素都没有读取时，通过数值 *next*来指定下个元素。 |

■ **例**5.2

假设这些变量已被定义为：str='ABC'；float=1.25；

(a) 执行命令x=abs(str)，将返回：

```
x =
    65    66    67
```

这是字符A、B、C的ASCII码。

(b) 如果输入number=hex2dec(str)，MATLAB将会给出：

```
number =
        2748
```

(c) 命令：

```
numstr = num2str(float)
disp(['Number as a string = ',numstr,'!']);
```

结果为：

```
numstr =
1.25

Number as a string = 1.25!
```

为了显示变量 **numstr**，实际上它是一个字符串，可以输入 char=numstr(4)，MATLAB就会给出：

```
char=
5
```

通过命令ischar(numstr)和whos，也可知道**numstr**是一个字符串。

(d) 命令numinfo=sprintf('The number%5.2e', float)给出：

```
numinfo =
The number = 1.25e+00
```

(e) 命令rational=rats(0.979796)将会给出一个包含浮点小数0.979796有理逼近的字符串：

```
rational =
   4995/5098
```

在命令littleRat=rats(0.979796,5)中的可选数字5将严格限定字符串的长度为5。

```
littleRat =
 48/49
```

■

命令rat在本书2.4节中定义。

■ **例**5.3

假设矩阵A被定义为：

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 3 \\ 2 & 4 & 7 \end{pmatrix}$$

命令mat2str(A)给出：

```
ans =
[1 1 3;2 4 7]
```

假设x=4.12345，命令mat2str(A)给出：

```
ans =
4.12345

num2str(x,2)

ans =
4.1
```

■

还有对字符串操作的逻辑函数和生成子串的函数。在下面的命令集46中假设**str**是一个字符串。

**命令集46 字符串函数(一)**

| | |
|---|---|
| `blanks(n)` | 返回有*n*个空格的字符串。 |
| `deblank(str)` | 返回没有后续空格的字符串**str**。 |
| `lower(str)` | 将**str**中所有字母转换为小写字母。 |
| `upper(str)` | 将**str**中所有字母转换为大写字母。 |
| `ischar(s)` | 如果*s*是字符数据类型，则返回 1；否则返回 0。在旧版中该命令还可以使用，但是在将来的高版本中将会被去掉。 |
| `isletter(str(i))` | 如果**str**中的第*i*个字符是字母，则返回1。 |
| `isspace(str)` | 返回一个和 **str**大小相同的向量。如果在 **str**中的字符是空格、制表符或者换行符，则向量的相应位置的元素为1；否则为0。 |
| `strcmp(str1,str2)` | 比较串**str1**和串**str2**，如果相等返回1；否则返回0。 |
| `stricmp(str1,str2)` | 和**strcmp**一样，但是在比较时不区分大小写。 |
| `str2mat(str1,str2,...)` | 用**str1**、**str2**等创建字符串矩阵。如果字符串$\mathbf{str}_i$的大小不同，MATLAB自动在较短的字符串后添加空格。函数最多可以带11个参数，但是它们本身也可以是字符串矩阵。 |
| `findstr(str1,str2)` | 返回一个向量，它包含**str1**中子串**str2**的起始位置。 |
| `strrep(str1,str2,str3)` | 在字符串**str1**中含有**str2**的所有位置用**str3**来代替。 |
| `strtok(str1,str2)` | 返回在**str1**中含有**str2**的第一个标记前所有的**str1**的部分。如果str2没有指定，MATLAB使用空格，那么就找出**str1**中不含有空格的第一个序列。 |
| `[outstr,rstr]=strtok` | 字符串outstr等于`strtok(str1,str2)`返回值，字符串**rstr** |
| `(str1,str2)` | 等于字符串**str1**余下的部分。 |
| `lasterr` | 返回上一个错误信息的字符串。 |
| `lastwarn` | 返回上一个警告信息的字符串。 |

■ **例**5.4

(a) `name=upper('matlab')`给出：

```
name=
MATLAB
```

(b) fun=strrep('hahaha','a','i')

```
fun=
hihihi
```

(c) 假设字符串变量定义为：

```
greet = 'Welcome', where = 'to Joan''s', party = ...
'birthday party!'
```

那么执行命令str2mat(greet,where,party)的结果为：

```
ans =
Welcome
to Joan's
birthday party!
```

(d) 可使用命令strtok从用逗号分隔内容的字符串中抽取出信息来。字符串 **text**定义为：

```
text = ...
'Monday,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday';

[day,rest] = strtok(text,',')

day =
Monday
rest =
,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday
```

通过字符串**rest**来调用strtok可以找到下一天等：执行命令[day2,rest]=strtok(rest,','), 结果为：

```
day2 =
Tuesday
rest =
,Wednesday,Thursday,Friday,Saturday,Sunday
```

要注意的是此时调用该命令时会读取到第 2个逗号处，是因为第 1个逗号在开始位置，它并不分隔字符串的任何部分。

■

有两个用于连接字符串的命令： strcat用于连接字符串， strvcat用于把字符串连接成一个列向量。分析字符串的内容可以使用命令： strmatch和strcmp。

**命令集47 字符串函数(二)**

| | |
|---|---|
| strcat(str1,str2,...) | 将字符串**str1**和**str2**连接起来，它不同于对细胞矩阵操作的cat命令。 |
| strvcat(str1,str2,...) | 将**str1**和**str2**连接成一个列向量。**str1**和**str2**必须要有相同的字符个数，行数可以不相等，最后结果的总行数是它们的行数之和。 |
| strmatch(key,strs) | 检查**strs**中的各行，返回一个向量，它包含了行以字符串**key**开头的行号。 |

strncmp(str1,str2,n) 比较**str1**和**str2**中前*n*个字符，如果相等返回1；否则返回0。
strncmpi(str1,str2,n) 和命令strncmp一样，但是在比较时不区分大小写。

■ **例**5.5

(a) 假设**A**和在例5.3中定义的一样，那么：

```
text1 = 'ResultMatrix = ';
text2 = mat2str(A);
restext = strcat(text1,text2)
```

则：

```
restext =
ResultMatrix =[1 1 3; 2 4 7]
```

注意在等号和左括号之间没有空格，这是因为strcat去掉了所有的前置空格，如果想保留这些空格就应该使用命令cat。

(b) 有下列语句：

```
head = ['First name' '  ' 'Last name'];
boss = ['John' '              ' 'Smith'];
workers = ['Arthur' '         ' 'Moore ';
           'Joseph' '         ' 'Jonson';
           'Daniel' '         ' 'Smart '];
```

在每行中间的字符串包含两个tab符号，它是用来做表格对齐的。

```
Table = strvcat(head,boss,workers)
```

```
Table =
First name      Last name
John            Smith
Arthur          Moore
Joseph          Jonson
Daniel          Smart
```

对**Table**来说，利用命令whos可知：

```
whos Table
```

```
  Name        Size        Bytes      Class

  Table       5x20          200      char array

Grand total is 100 elements using 200 bytes
```

■

它的大小取决于最长的行，所以它更适合于存储这种分开形式的列表。

■ **例**5.6

语句如下：

```
carVocabulary = strvcat('car','carpool','police car')
```

```
carVocabulary =
car
carpool
police car
strmatch('car',carVocabulary)

ans =
     1
     2
```

这个结果是字符串中以'cat'开头的行号。

■

### 5.1.3 显示和输入

可以通过简单地输入变量的名字来显示数字矩阵或者字符串向量的内容，见 2.3节。结果将显示出变量的名字和内容。

另一种显示变量的值就是使用命令 disp。使用它只显示出变量的内容，这是有用的，特别是在字符串的应用中。

**命令集48 显示命令**

| | |
|---|---|
| disp(A) | 显示矩阵**A**的内容，如果**A**是字符串，则显示出它的文本。 |

通过input命令来接收从终端输入的内容，它也可以显示文本和提示，见附录 A.6中的例子。从例13.9可知命令disp可以和num2str和int2str结合使用。

**命令集49 输入命令**

| | |
|---|---|
| input(out,*in*) | 在屏幕上显示出字符串**out**的文本并等待终端的输入。如果变量**in**是's'，则输入的内容以字符串的形式进行保存，通常MATLAB在保存前要尽可能地求出表达式的值。如果使用格式控制符号如'\n'，字符串**out**可以是若干行。 |

■ **例**5.7

(a) 读入一个实数$x$：

```
x=input('Give a number x:')
```

显示的结果为：

```
Give a number x: 2.0944
x =
    2.0944
```

(b) 读入矩阵**A**。input命令之后的分号用来限定输出结果。

```
A=input('Give the matrix A row by row:');
```

显示并输入：

```
Give the matrix A row by row: [1  2 ; 3  5]
```

(c) 输入时还允许使用数学表达式和函数：

```
A=input('Please give me a matrix: ');
```

显示并输入：

```
Please give me a matrix: rand(4)+hilb(4)
```

(d) 输入命令还可带多个参数：

```
[m, n]=input('Give the size of A:')
```

显示并输入：

```
Give the size of:Asize(A)
```

(e) 逻辑表达式的计算：

```
s = input('MATLAB input\nWrite an expression: ')
MATLAB input
Write an expression: log2(8) + 7 < 10
s =
     0
```

(f) 读入一个字符串：

```
name=input('What is your last name? ');
```

显示并输入：

```
What is your last name? 'Smith'
```

注意，在这种情况下输入的字符串要用’’引出。如果命令是这样：

```
strname=input('What is your last name? ', 's');
```

就可以直接输入字符串：

```
What is your last name? Smith
```

■

MATLAB还有一些其他的方式来读入字符串，比如通过菜单；见 14.3节。

### 5.1.4 字符串求值

MATLAB命令可以以字符串的形式进行输入和存储。这些命令字符串通过eval命令来求值。

**命令集50 字符串求值**

| | |
|---|---|
| eval(str) | 执行**str**中包含的MATLAB命令并返回结果。 |
| eval(str1, str2) | 执行**str1**中的MATLAB命令，如果没有错误就和执行eval(str1)一样；如果在对**str1**求值中第一个字符串是一个错误，则对字符串**str2**进行求值，给出一个错误信息或者其他内容。 |
| [x1,x2,...]=evalin(aa, str) | 和eval一样，决定在什么样的工作区中如读/写变量来对**str**求值。**aa**字符串可以是‘caller’(在evalin调用的工作内进行求值)，也可以是‘base’(在MATLAB命令执行区来求值)。结果数据存储在**x1**，**x2**, …中。 |
| evalin(aa, str, alt) | 在工作区**aa**(见上)中试着对字符串**str**求值。如果不成功，则对字符串**alt**求值。 |
| assignin(aa, name, | 把值**val**赋给变量**name**，如果变量不存在，则新建。参 |

| | |
|---|---|
| val) | 量**aa**表示工作区，见命令evalin。 |
| g=inline(str,*arg1*, *arg2*,···) | 从字符串**str**中建立一个叫内联的函数**g**，如存储在工作内存中的函数，可以用 g(**val** 1,**val** 2···)来调用。函数中参数的名字可以在字符串 **arg**1，**arg**2，…中给出，如果没有给出，MATLAB将从str中找出小写字母作为参数的名字。 |
| g=inline(str,n) | 用*n*+1个参量*x*，*P1*，*P2*，…，*Pn*来建立一个内联函数。 |
| argnames(g) | 用内联函数**g**中的参量名字创建一个细胞矩阵。 |
| vectorize(g) | 为了满足元素操作，在 *、/和^前加一个点‘.’，建立一个向量化的内联函数**g**。 |
| formula(g) | 把内联函数**g**作为字符串返回。 |
| char(g) | 等同于formula(g)。 |

命令eval使得MATLAB成为一个可灵活编程的语言。比如这个命令用来调用 MATLAB中没有预定义的函数，这很有用处，见 12.4节。

■ **例**5.8

(a) 假设要编写一段简单的矩阵计算的程序来输入矩阵 **A**和**B**，并且可以让用户来决定是否进行加法或者乘法操作。程序代码如下：

```
disp('Matrix analysis program. Give two matrices:');
A = input('A = ');
B = input('B = ');

choice = input('Choose one: 1 = A+B, 2 = A*B: ');

switch choice
  case 1, eval('disp(''Addition: ''); A+B');
  case 2, eval('disp(''Multiplication: ''); A*B');
end;
```

为了不结束字符串，这里应该用两个分号。在 5.1.3节对命令disp和input进行了描述，命令switch-case可参见12.1节。

由于矩阵维数的不正确等原因，如矩阵 **AB**相乘，可能会导致一些错误，所以要求矩阵的维数要一致。用第 2个字符串作为命令 eval的参数可以得到一个错误信息。通过运行下列程序代码可以得到：

```
disp('Matrix analysis program. Give two matrices:');
A = input('A = ');
B = input('B = ');

choice = input('Choose one: 1 = A+B, 2 = A*B :');

switch choice
  case 1, eval('disp(''Addition: ''); A+B','catchInfo');
  case 2, eval('disp(''Multiplication: ''); A*B', 'catchInfo');
end;
```

用命令eval来调用保存在文件**catchInfo.m**中的用户自定义的函数**catchInfo**。

```
function catchInfo

errStr = lasterr;
dimStr = findstr(errStr,'dimensions must agree');

if ~isempty(dimStr)                % If wrong dimensions...

  if ~isempty( findstr(errStr,'Inner'))
    disp('Error!  A*B requires A:m*n, B:n*p.');
  else
    disp('Error! Addition requires A:m*n, B:m*n.');
  end
end
```

字符串变量**lasterr**包含了MATLAB中最新的错误信息，假设给出矩阵：

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

相加可得：

```
Addition:
Error! Addition requires A:m*n, B:m*n.
```

相乘可得：

```
Multiplication:
Error!  A*B requires A:m*n, B:n*p.
```

(b) 字符串变量可以和字符串常量结合使用。比如有一个变量名为 file='myfile.mat'，要想将当前的变量值保存到这个文件中，可以输入 eval['save',file]，这和输入save myfile.mat是一样的。在2.8节中定义了save命令。

(c) MATLAB还可以对向量函数进行求值。令**str1**='b.*sin(k.*x)'。**b, k, x**都是向量：

```
b = [1 2 3]; k = [2 2 2]; x = [1.2 1.5 1.2];
```

执行命令：

```
values=eval(str1)
```

结果为：

```
values =
    0.6755    0.2822    2.0264
```

(d) 创建一个字符串：

```
fcn='input(''Give a function","$")'
```

在5.1.3节中提到了input命令，现在可以对从终端选择的函数进行求值操作。执行命令：

```
fplot(eval(fcn), [0,4])
```

在屏幕上显示如下结果：

```
Give a function
```

如果输入sin，MATLAB就会打开一个图形窗口显示出正弦函数。

假设一个名为**sinx2.m**的M文件(见2.9节)包含下列代码：

```
function y = sinx2(x)
y = sin(x.^2);
```

有关M文件的更多信息可参见12.3节，如果在文本*Give a function*之后输入sinx2，就可

以画出函数sinx²的图形来，如图5-1所示。

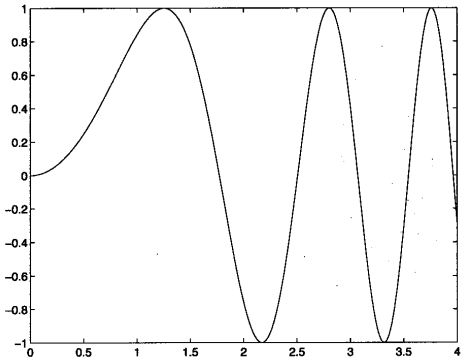

图5-1 函数sinx²在[0，4]区间上的图形

■

求值的字符串也能够包含在MATLAB结构语句中，如`if`、`while`、`for`等等；见12章。

■ **例**5.9

建立一个内联函数来计算表达式3sin(*x*)+5cos(*y*)，可以输入：

```
g = inline('3*sin(x)+5*cos(y)','x','y')

g =
     Inline function:
     g(x,y) = 3*sin(x)+5*cos(y)

argnames(g)

ans =
    'x'
    'y'

g(pi,2*pi)
ans =
     5
```

如果输入`whos`，可显示出：

```
  Name        Size              Bytes  Class

  ans         1x1                   8  double array
  g           1x1                 898  inline object

Grand total is 75 elements using 906 bytes
```

■

## 5.2 整数

MATLAB中有大量的关于整数操作的命令。在命令集 51中的*a*和*b*都是整数。

**命令集51 整数函数**

| | |
|---|---|
| mod(a,b) | 返回$a$，$b$相除后的余数。 |
| factor(a) | 返回$a$的素数因数。 |
| primes(a) | 返回一个由不大于$a$的素数组成的行向量。 |
| isprimes(a) | 如果$a$是一个素数，则返回1。 |
| nextpow2(a) | 返回最小的整数$n$，满足条件$2^n>a$。 |
| perms(c) | 返回向量c的所有可能的排列。 |
| nchoosek(A,k) | 返回一个每行有$k$个元素的矩阵，并列出了矩阵A中$k$个元素的可能每种组合。 |

函数perms和nchoosek随着输入返回的结果会变得很多，所以可能需要较长的时间来计算。

■ **例5.10**

判断4567是否是一个素数，执行命令isprime(4567)可得结果为：

```
ans=
     1
```

在4569中有哪些因数呢？可以输入命令factor(4569)并执行得到的结果为：

```
ans =
           3        1523
```

■

## 5.3 位操作

在MATLAB中可以对整数变量进行位操作，如果答案为真返回 1；为假返回0。

**命令集52 位操作**

| | |
|---|---|
| bitand(a,b) | 返回$a$和$b$按位与操作后的值。 |
| bitor(a,b) | 返回$a$和$b$按位或操作后的值。 |
| bitxor(a,b) | 返回$a$和$b$按位异或操作后的值。 |
| bitget(a,bit) | 返回在整数$a$中位置$bit$的位值。 |
| bitset(a,bit,newbit) | 将值$newbit$赋于整数$a$中在位置$bit$的位。 |
| bitshift(a,n) | 对$a$进行$n$步移位操作，如果$n$为整数则左移；否则右移。 |
| bitcmp(a,n) | 返回$n$位$a$的补码整数。 |
| bitmax | 返回机器的最大浮点整数。 |

■ **例5.11**

令a=7, b=3, c=4，它们的二进制码为a=111, b=011, c=100。执行命令bitand(a,b)，结果为：

```
ans =
     3
```

```
bitor(b,c)
```

```
ans =
     7

bitset(a,2,0)

ans =
     5
```

■

## 5.4 集合

MATLAB 5中有大量的将向量和细胞矩阵解释为集合及其操作的函数。在命令集 53中变量a和b都是向量。注意这些命令都可以用添加字符串‘rows’作为最后一个参数来对矩阵操作。在这种情况下，矩阵的每一行就看成是一个单一集合，同时函数对一次一个集合作用。返回的集合中的数是有序的，并且每个数只出现一次。

**命令集53 集合**

| | |
|---|---|
| intersect(a,b) | 返回集合**a**和**b**的交集。 |
| ismember(a,s) | 返回一个和 **a**一样大小的 0-1向量。当 **a**的元素包含在 **s**中时，该向量中相应位置为1；否则为0。 |
| setdiff(a,b) | 返回存在集合**a**中但不存在集合**b**中的值。 |
| setxor(a,b) | 返回存在集合 **a**中但不存在集合 **b**中和存在集合 **b**中但不存在集合**a**中的所有的值。 |
| union(a,b) | 返回集合**a**和**b**的并集。 |
| unique(a) | 去掉集合**a**中重复的值，保证每个值在集合中只出现一次。 |

■ **例**5.12

令a=[12 24 42 24 12]，b=[96 42 64]

执行命令union(a,b)，得到的结果为：

```
ans =
    12    24    42    64    96

unique(a)

ans =
    12    24    42

intersect(a,b)

ans =
    42
```

■

## 5.5 细胞矩阵

细胞矩阵(或细胞数组)中不同位置(细胞)可有不同数据类型，这就使得它不同于只能由数

字组成的数字矩阵和只能由字符串组成的文本矩阵。所以，一个细胞矩阵可以包含比如一个字符串、两个数字和一个细胞矩阵。细胞矩阵也可象数字矩阵一样有多维的。对于一维细胞矩阵，也类似于数字矩阵，称为细胞向量。

有三种方法来创建一个细胞矩阵。第 1种方法是使用大括号，‘ {} ’，就象用中括号‘ [] ’来创建数字矩阵一样；第 2种方法是对细胞进行逐一赋值，称为细胞赋值；第 3种方法是创建一个大小合适的空矩阵。矩阵中所有的行必须要有相同的细胞数。

■ **例**5.13

可以象以下的方法用不同数据类型直接分配：

```
A = {'John' 'Smith' 38 11.21; 'Paul' 'Anderson' 41 23.12}
```

或者是：

```
A = {'John','Smith',38,11.21; 'Paul','Anderson',41,23.12}

A =
    'John'    'Smith'        [38]    [11.2100]
    'Paul'    'Anderson'     [41]    [23.1200]
```

如果一个细胞中包含一个细胞矩阵，就要求这样来分配：

```
B = { {2 2; 1 3} 22.3; 42 21 }

B =
    {2x2 cell}    [22.3000]
    [      42]    [     21]
```

可以对某个细胞进行赋值来创建一个完整的细胞矩阵。

```
C{2,1} = 12.2

C =
            []
    [12.2000]
```

要创建一个2×4的空细胞矩阵，可以使用命令：

```
D = cell(2,4)

D =
     []     []     []     []
     []     []     []     []
```

■

有些函数只能作用于细胞矩阵，还有一些其他的函数如cat和strcat也可以作用于细胞矩阵。

**命令集54 细胞矩阵**

| | |
|---|---|
| cell(m,n) | 创建一个$m\times n$的空细胞矩阵。 |
| cell2struct(cell, posts,dim) | 用细胞矩阵**cell**的元素和域**posts**创建一个结构，根据*dim*来决定挑选元素。*dim*=1是就是将**cell**中的第1列作为结构数1；要取出元素，*dim*应该为2。 |

| | |
|---|---|
| `celldisp(cell)` | 逐个显示**cell**的每个元素的值。还可以用该命令来显示 **cell** 中的细胞矩阵的元素。 |
| `cellplot(cell)` | 显示出**cell**的结构图。 |
| `cellstr(s)` | 用**s**中每一行作为一个细胞来创建一个细胞向量，命令char是这个函数的逆操作。 |
| `iscellstr(cell)` | 如果**cell**中只含有字符串，则返回 1；如果**cell**中既有细胞矩阵又有字符串，则返回0。 |
| `num2cell(A,dim)` | 返回一个和矩阵 **A**一样大小的细胞矩阵。如果给出了参量 *dim*，在它自己的细胞中将 *dim*维作为一个向量，这样得到的矩阵就和**A**的大小不一样。 |
| `[out1,out2,···]=deal(in1,in2,···)` | 将输入拷贝到输出中，如`out1=in1`，`out2=in2`等；参见`helpdesk`中有关细胞矩阵和结构的例程。 |

■ **例** 5.14

执行如下命令可以得到例5.13中细胞矩阵B的每个元素结构图：

```
cellplot(B)
```

结果如图5-2所示。

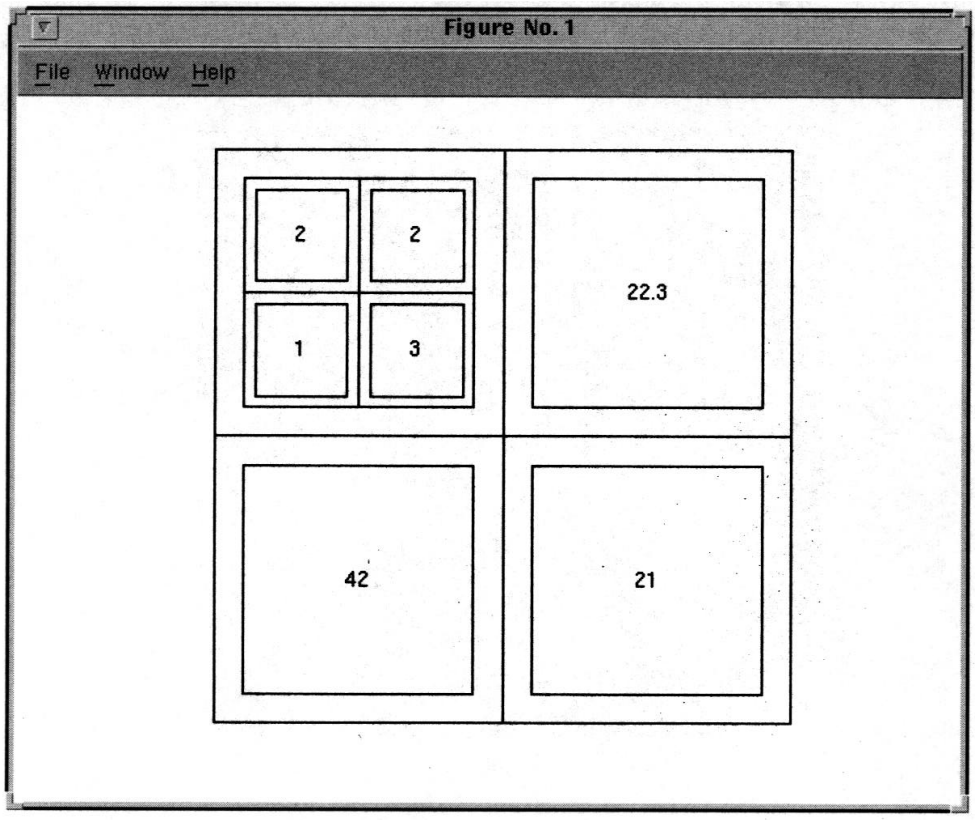


图5-2 细胞矩阵的图形表示

用命令celldisp(B)来显示细胞矩阵的每个元素的值：

```
celldisp(B)

B{1,1}{1,1} =
     2
B{1,1}{2,1} =
     1
B{1,1}{1,2} =
     2
B{1,1}{2,2} =
     3
B{2,1} =
    42
B{1,2} =
   22.3000
B{2,2} =
    21
```

■