第 3 章 MATLAB 字符串和数组

CHAPTER 3

本章要点：

- 字符串；
- 多维数组；
- 结构数组；
- 元胞数组。

在 MATLAB 语言中，数组是存储和运算的基本单元，向量和矩阵是数组的特例。MATLAB 常用数组包括一维数组、二维数组、多维数组、结构数组和元胞数组。其中字符串就是一个典型的一维字符数组。二维数组与二维矩阵在形式上是一样的，在第 2 章已经详细介绍过。本章主要介绍字符串的创建方法、操作和转换常用函数，以及多维数组、结构数组和元胞数组的创建和使用方法。

3.1 字符串

在 MATLAB 语言中，字符串是 MATLAB 语言的一个重要组成部分，MATLAB 语言提供强大的字符串处理功能，本节主要介绍字符串的创建方法、操作和转换常用函数等内容。

3.1.1 字符串的创建

在 MATLAB 语言中，字符串一般以 ASCII 码形式存储，以行向量形式存在，并且每个字符占用两字节的内存。在 MATLAB 语言中，创建一个字符串可以用下面几种方法。

(1) 直接将字符内容用单引号(' ')括起来，例如：

```
>> str = 'Teacher_name'
str =
    'Teacher_name'
```

字符串的存储空间如下所示，所定义的字符串有 12 个字符，每个字符占用两字节的内存。

```
>> whos
  Name       Size              Bytes  Class     Attributes
  str        1x12                 24  char
```

若要显示单引号(')字符,需要使用两个单引号,例如:

```
>> str = 'I''m a teacher'
str =
    'I'm a teacher'
```

(2) 用方括号连接多个字符串组成一个长字符串,例如:

```
>> str = ['I''m' 'a' 'teacher']
str =
    'I'm a teacher'
```

(3) 用函数 strcat 把多个字符串水平连接合并成一个长字符串,strcat 函数语法格式如下:

```
str = strcat(str1,str2,…)
```

例如:

```
>> str1 = 'I''m a student';
>> str2 = ' of';
>> str3 = ' Guangdong Ocean University';
>> str = strcat(str1,str2,str3)
str =
    'I'm a student of Guangdong Ocean University'
```

(4) 用函数 strvcat 把多个字符串连接成多行字符串,strvcat 函数语法格式如下:

```
str = strvcat(str1,str2,…)
```

例如:

```
>> str1 = 'good';
>> str2 = 'very good';
>> str3 = 'very very good';
>> strvcat(str1,str2,str3)
ans =
  3×14 char 数组
    'good           '
    'very good      '
    'very very good '
```

MATLAB 语言可以用 abs 或者 double 函数获取字符串所对应的 ASCII 码数值矩阵。相反,可以用 char 函数把 ASCII 码转换为字符串。例如:

```
>> str1 = 'I''m a teacher'
str1 =
    'I'm a teacher'
>> A = abs(str1)                  % 把字符串转换为对应的 ASCII 码数值矩阵
A =
    73    39   109    32    97    32   116   101    97    99   104   101   114
>> str = char(A)                  % 把 ASCII 码数值矩阵转换为字符串
str =
    'I'm a teacher'
```


微课视频

【例 3-1】 已知一个字符串向量 str='It is a Yellow Dog',完成以下任务:

(1) 计算字符串向量字符个数;

(2) 显示'a Yellow Dog';

(3) 将字符串倒序重排;

(4) 将字符串中的大写字母变成相应小写字母,其余字符不变。

将下列程序代码存为脚本文件 fexam_3_1.m,并放于当前工作目录下。

```
str = 'It is a Yellow Dog'                    % 创建字符串向量
n = length(str)                               % 计算字符串向量字符个数
str1 = str(7:18)                              % 显示'a Yellow Dog'
str2 = str(end: - 1:1)                        % 将字符串倒序重排
k = find(str > = 'A'&str < = 'Z')             % 查找大写字母的位置
str(k) = str(k) + ('a' - 'A')                 % 将大写字符变成相应小写字母
```

然后在命令行窗口中运行下列指令:

```
>> fexam_3_1
str =
    'It is a Yellow Dog'
n =
    18
str1 =
    'a Yellow Dog'
str2 =
    'goD wolleY a si tI'
k =
     1     9    16
str =
    'it is a yellow dog'
```

3.1.2 字符串的操作

1. 字符串比较

MATLAB 语言比较两个字符串是否相同的常用函数有 strcmp、strncmp、strcmpi 和 strncmpi 这 4 个,字符串比较函数的调用格式及功能说明如表 3-1 所示。

表 3-1 字符串比较函数格式及功能

函 数 名	调 用 格 式	功 能 说 明
strcmp	strcmp(str1,str2)	比较两个字符串是否相等,相等为 1,不相等为 0
strncmp	strncmp(str1,str2,n)	比较两个字符串前 n 个字符是否相等,相等为 1,不相等为 0
strcmpi	strcmpi(str1,str2)	忽略大小写,比较两个字符串是否相等,相等为 1,不相等为 0
strncmpi	strncmpi(str1,str2,n)	忽略大小写,比较两个字符串前 n 个字符是否相等,相等为 1,不相等为 0

例如:

```
>> str1 = {'one','two','three','four'};      % 定义字符串元胞数组
>> str2 = 'two';
```

```
>> strcmp(str1,str2)                    % 比较两个字符串 str1 和 str2 是否相等
ans =
  1×4 logical 数组
   0   1   0   0
>> str1 = 'I am a handsome boy';
>> str2 = 'I am a pretty girl';
>> strncmp(str1,str2,7)                 % 比较两个字符串 str1 和 str2 前 7 个字符是否相等
ans =
logical
   1
>> strncmp(str1,str2,8)
ans =
logical
   0
>> str1 = 'MATLAB 2018a';
>> str2 = 'MATLAB 2018A';
>> strcmp(str1,str2)
ans =
logical
   0
>> strcmpi(str1,str2)                   % 忽略大小写,比较两个字符串是否相等
ans =
logical
   1
>> str1 = 'I am a handsome boy';
>> str2 = 'I am A pretty girl';
>> strncmpi(str1,str2,7)                % 忽略大小写,比较两个字符串前 7 个字符是否相等
ans =
logical
   1
>> strncmp(str1,str2,7)
ans =
logical
   0
```

2. 字符串查找和替换

MATLAB语言查找与替换字符串的常用函数有 5 个：strfind、findstr、strmatch、strtok 和 strrep。字符串查找函数的调用格式及功能说明如表 3-2 所示。

表 3-2 字符串查找函数

函 数 名	功 能 说 明
strfind(str, 'strl')	在字符串 str 中查找另一个字符串 strl 出现的位置
findstr(str, 'strl')	在一个较长字符串 str 中查找较短字符串 strl 出现的位置
strmatch('strl',str)	在 str 字符串数组中,查找匹配以字符 strl 为开头的字符串所在的行数
strtok(str)	从字符串 str 中截取第一个分隔符(包括空格、Tab 键和回车键)前面的字符串
strrep(str, 'oldstr', 'newstr')	在原来字符串 str 中,用新的字符串 newstr 替换旧的字符串 oldstr

例如：

```
>> str = 'sqrt(X) is the square root of the elements of X. Complex'   % 构建一个长字符串 str
str =
    'sqrt(X) is the square root of the elements of X. Complex'
>> findstr(str,'of')              % 在一个较长字符串 str 中查找较短字符串'of'出现的位置
ans =
    28    44
>> strfind(str,'of')              % 在字符串 str 中查找字符串'of'出现的位置
ans =
    28    44
>> strrep(str,'X','Y')            % 在原来字符串 str 中,用新的字符串'Y'替换旧的字符串'X'
ans =
    'sqrt(Y) is the square root of the elements of Y. Complex'
>> strtok(str)                    % 从字符串 str 中截取第一个分隔符前面的字符串 sqrt(X)
ans =
    'sqrt(X)'
>> str = strvcat('good','very good','very very good')          % 构建字符串数组 str
str =
  3×14 char 数组
    'good           '
    'very good      '
    'very very good'
>> strmatch('very',str)           % 在字符串数组 str 中,查找匹配以字符'very'为开头的字符串
                                  % 所在的行数
ans =
     2
     3
```

3. 字符串的其他操作

在 MATLAB 语言中，除了常用的字符串创建、比较、查找和替换操作外，还有许多其他字符串操作，如表 3-3 所示。

表 3-3　字符串其他操作函数

函　数　名	函数功能及说明
upper(str)	将字符串 str 中的字符转换为大写
lower(str)	将字符串 str 中的字符转换为小写
strjust(str,'right') strjust(str)	将字符串 str 右对齐
strjust(str,'left')	将字符串 str 左对齐
strjust(str,'center')	将字符串 str 中间对齐
strtrim(str)	删除字符串开头和结束的空格符
eval(str)	执行字符常量 str 运算

例如：

```
>> str = 'Matlab 2020a'
str =
    'Matlab 2020a'
```

```
>> upper(str)                                          % 将字符转换为大写
ans =
    'MATLAB 2020A'
>> lower(str)                                          % 将字符转换为小写
ans =
    'matlab 2020a'
>> str = strvcat('good','very good','very very good')  % 创建一个字符串 str 序列
str =
  3×14 char 数组
    'good     '
    'very good      '
    'very very good'
>> strjust(str)                                        % 将字符串 str 右对齐
ans =
  3×14 char 数组
    '          good'
    '     very good'
    'very very good'
>> strjust(str,'right')                                % 将字符串 str 右对齐
ans =
  3×14 char 数组
    '          good'
    '     very good'
    'very very good'
>> strjust(str,'left')                                 % 将字符串 str 左对齐
ans =
  3×14 char 数组
    'good          '
    'very good     '
    'very very good'
>> strjust(str,'center')                               % 将字符串 str 中间对齐
ans =
  3×14 char 数组
    '       good   '
    '    very good  '
    'very very good'
>> str = '  matlab 2020a   '
str =
    '  matlab 2020a   '
>> strtrim(str)                                        % 删除字符串开头和结束的空格符
ans =
    'matlab 2020a'
>> str = '2 * 3 + 6'                                   % 创建一个字符串常量
str =
    '2 * 3 + 6'
>> eval(str)                                           % 执行字符常量 str 运算
ans =
    12
```

3.1.3 字符串转换

在MATLAB语言中,字符串进行算术运算会自动转换为数值型。MATLAB还提供了许多字符串与数值之间转换函数,如表3-4所示。

表3-4 字符串与数值转换函数

函数名	格式及例子	功能与说明
abs	abs('a')=97	将字符串转换为ASCII码数值
double	double('a')=97	将字符串转换为ASCII码数值的双精度类型数据
char	char(97)=a	将数值整数部分转换为ASCII码等值的字符
str2num	str2num('23')=23	将字符串转换为数值
num2str	num2str(63)= '63'	将数值转换为字符串
str2double	str2double('97')=97	将字符串转换为双精度类型数据
mat2str	mat2str([32,64;97,101])= '[32 64;97 101]'	将矩阵转换为字符串
dec2hex	dec2hex(64)= '40'	将十进制整数转换为十六进制整数字符串
hex2dec	hex2dec ('40')=64	将十六进制字符串转换为十进制整数
dec2bin	dec2bin(16)= '10000'	将十进制整数转换为二进制整数字符串
bin2dec	bin2dec('10000')=16	将二进制字符串转换为十进制整数
dec2base	dec2base(16,8)= '20'	将十进制整数转换为指定进制整数字符串
base2dec	base2dec('20',8)=16	将指定进制字符串转换为十进制整数

例如,可以利用字符串与数值之间的转换,对一串字符明文进行加密处理。MATLAB代码如下:

```
>> str = 'welcome to MATLAB 2020a'      % 创建待加密的字符串
str =
    'welcome to MATLAB 2020a'
>> str1 = str - 2;                      % 将每个字符的 ASCII 码值减去 2
>> str2 = char(str1)                    % 将移位后的每个 ASCII 码转换为字符,完成加密
str2 =
    'ucjamkc - rm - K?RJ?@ - 0.0.'
>> str3 = str2 + 2;                     % 解密与加密相反的过程
>> str4 = char(str3)
str4 =
    'welcome to MATLAB 2020a'
```

3.2 多维数组

多维数组(Multidimensional Arrays)是三维及以上的数组。三维数组是二维数组的扩展,二维数组可以看成行和列构成的面,三维数组可以看成行、列和页构成的“长方体”,实际中三维数组用得比较多。

三维数组用3个下标表示,数组的元素存放遵循的规则是:首先存放第一页第一列,接着是该页的第二列、第三列,以此类推;第一页最后一列接第二页第一列,直到最后一页最

后一列结束。

四维数组和三维数组有些类似，使用 4 个下标表示，更高维的数组是在后面添加维度来确定页。

3.2.1 多维数组的创建

多维数组的创建一般有 4 种方法：直接赋值法、二维数组扩展法、使用 cat 函数创建法和使用特殊数组函数法。

1. 直接赋值法

例如，创建三维数组 A。

```
>> A(:,:,1) = [1 2;3 4]          % 赋值第一页
A =
     1     2
     3     4
>> A(:,:,2) = [5 6;7 8]          % 赋值第二页
A(:,:,1) =
     1     2
     3     4
A(:,:,2) =
     5     6
     7     8
>> whos A                        % 查看三维数组 A 的属性
  Name       Size              Bytes  Class     Attributes
  A          2x2x2                64  double
```

2. 二维数组扩展法

MATLAB 可以利用二维数组扩展到三维数组，例如：

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

如果第一页不赋值，直接赋值第二页，那么也能产生三维数组，第一页值全默认为 0，例如：

```
>> C(:,:,2) = [5 6;7 8]
C(:,:,1) =
     0     0
     0     0
C(:,:,2) =
     5     6
     7     8
```

3. 使用 cat 函数创建法

MATLAB 语言可以使用 cat 函数，把几个原先赋值好的数组按照某一维连接起来，创建一个多维数组。函数调用格式如下：

```
A = cat(n,A1,A2,…)              % 将 A1、A2 等数组连接成 n 维数组
```

例如，使用 cat 函数创建多维数组：

```
>> A1 = [1 2;3 4];              % 创建三个二维数组
>> A2 = [5 6;7 8];
>> A3 = [9 10;11 12];
>> A = cat(1,A1,A2,A3)          % 用函数 cat 按垂直方向连接 A1、A2 和 A3 成一个二维数组
A =
     1     2
     3     4
     5     6
     7     8
     9    10
    11    12
>> A = cat(2,A1,A2,A3)          % 用函数 cat 按水平方向连接 A1、A2 和 A3 成一个二维数组
A =
     1     2     5     6     9    10
     3     4     7     8    11    12
>> A = cat(3,A1,A2,A3)          % 用函数 cat 创建一个三维数组
A(:,:,1) =
     1     2
     3     4
A(:,:,2) =
     5     6
     7     8
A(:,:,3) =
     9    10
    11    12
```

4. 使用特殊数组函数法

MATLAB 语言提供了许多创建特殊多维矩阵的函数，例如，rand、randn、ones 和 zeros 等，这些函数都可以创建多维特殊矩阵。函数的功能和使用方法与二维特殊矩阵类似。

例如：

```
>> A = rand(2,2,2)              % 创建 0～1 均匀分布的三维随机矩阵
A(:,:,1) =
    0.0975    0.5469
    0.2785    0.9575
A(:,:,2) =
    0.9649    0.9706
    0.1576    0.9572
>> B = randn(2,2,2)             % 创建正态分布的三维随机矩阵
B(:,:,1) =
   -0.0631   -0.2050
    0.7147   -0.1241
```

```
B(:,:,2) =
     1.4897    1.4172
     1.4090    0.6715
>> C = ones(2,2,2)              % 创建三维 1 矩阵
C(:,:,1) =
     1     1
     1     1
C(:,:,2) =
     1     1
     1     1
>> D = zeros(2,2,2)             % 创建三维 0 矩阵
D(:,:,1) =
     0     0
     0     0
D(:,:,2) =
     0     0
     0     0
>> E = rand(2,2,2,2)            % 创建 0～1 均匀分布的四维随机矩阵
E(:,:,1,1) =
     0.0357    0.9340
     0.8491    0.6787
E(:,:,2,1) =
     0.7577    0.3922
     0.7431    0.6555
E(:,:,1,2) =
     0.1712    0.0318
     0.7060    0.2769
E(:,:,2,2) =
     0.0462    0.8235
     0.0971    0.6948
```

3.2.2 多维数组的操作

MATLAB多维数组操作主要有数组元素的提取、多维数组形状的重排和维度重新排序。

1. 多维数组元素的提取

提取多维数组元素的方法有两种：全下标方式和单下标方式。

1）全下标法

例如：

```
>> A = [1 2;3 4];
>> A(:,:,2) = [5 6;7 8]          % 创建一个三维数组
A(:,:,1) =
     1     2
     3     4
A(:,:,2) =
     5     6
     7     8
>> a = A(1,1,2)                 % 用全下标法提取第 2 页第 1 行第 1 列的元素
a =
     5
```

2）单下标法

MATLAB单下标法提取多维数组的元素遵循的规则：先是第一页第一列，然后是第一页第二列，以此类推；直到第一页最后一列，然后是第二页第一列，直到最后一页最后一列。

例如：

```
>> A = [1 2;3 4];
>> A(:,:,2) = [5 6;7 8]          % 创建一个三维数组
A(:,:,1) =
     1     2
     3     4
A(:,:,2) =
     5     6
     7     8
>> a = A(7)                      % 单下标法提取第 7 个元素
a =
     6
```

2. 多维数组形状的重排

MATLAB语言可以利用函数 reshape 改变多维数组的形状，函数的调用格式如下：

```
A = reshape(A1,[m,n,p])
```

其中 m、n 和 p 分别表示行、列和页，A1 是重排的多维数组。数组还是按照单下标方式存储顺序重排，重排前后元素数据大小没变，位置和形状会改变。

例如：

```
>> A1 = rand(3,3);               % 创建三维数组
>> A1(:,:,2) = randn(3,3)
A1(:,:,1) =
     0.4456     0.7547     0.6551
     0.6463     0.2760     0.1626
     0.7094     0.6797     0.1190
A1(:,:,2) =
    -0.0068     0.3714    -1.0891
     1.5326    -0.2256     0.0326
    -0.7697     1.1174     0.5525
>> A = reshape(A1,[2,3,3])      % 重排第 2 行、第 3 列和第 3 页的三维数组
A(:,:,1) =
     0.4456     0.7094     0.2760
     0.6463     0.7547     0.6797
A(:,:,2) =
     0.6551     0.1190     1.5326
     0.1626    -0.0068    -0.7697
A(:,:,3) =
     0.3714     1.1174     0.0326
    -0.2256    -1.0891     0.5525
```

3. 多维数组维度的重新排序

MATLAB语言可以利用函数 permute 重新定义多维数组的维度顺序，按照新的行、列和页重新排序数组，permute 改变了线性存储的方式，函数的调用格式如下：

```
A = permute(A1,[m,n,p])
```

其中 m、n 和 p 分别是列、行和页，A1 是重定义的多维数组，要求定义后的维度不少于原数组的维度，而且各维度数不能相同。

例如：

```
>> A1 = rand(3,3);                          % 创建一个三维数组
>> A1(:,:,2) = randn(3,3)
A1(:,:,1) =
     0.8909     0.1386     0.8407
     0.9593     0.1493     0.2543
     0.5472     0.2575     0.8143
A1(:,:,2) =
   - 0.1924   - 1.4023   - 0.1774
     0.8886   - 1.4224   - 0.1961
   - 0.7648     0.4882     1.4193
>> B = permute(A1,[3,2,1])                  % 重新定义三维数组，存储顺序改变
B(:,:,1) =
     0.8909     0.1386     0.8407
   - 0.1924   - 1.4023   - 0.1774
B(:,:,2) =
     0.9593     0.1493     0.2543
     0.8886   - 1.4224   - 0.1961
B(:,:,3) =
     0.5472     0.2575     0.8143
   - 0.7648     0.4882     1.4193
```

彩色图像被读入 MATLAB 中，RGB 三种颜色分量一般被存为三维数组。对彩色图像进行处理，实际上是对三维数组进行提取和操作，所以用 MATLAB 语言处理彩色图像比较方便。下面通过一个例子说明三维数组在彩色图像处理中的应用。

【例 3-2】 用 MATLAB 语言对一幅彩色图像分别提取红色分量、绿色分量和蓝色分量，并在同一个图形窗口不同区域显示，利用 cat 函数把三个分量连接成一个三维数组，并显示合成后的图像。


微课视频

程序代码如下：

```
clear
close all;
% ------ 读入图片 flower.jpg 存入 A 中 ------ %
A = imread('D:\work\flower.jpg');
subplot(2,2,1)
imshow(A);                    % 将三维数组 A 显示为彩色图像
title('原始图像')
[r c d] = size(A); % 计算图像大小，r 为行，c 为列，d 为页，1、2 和 3 分别代表红、绿和蓝分量
% ------ 提取红色分量并显示分解图 ------ %
red(:,:,1) = A(:,:,1);
red(:,:,2) = zeros(r,c);  % 蓝色和绿色分量用 0 矩阵填充
red(:,:,3) = zeros(r,c);
red = uint8(red);
```

```
subplot(2,2,2)
imshow(red)
title('红色分量');
% -------- 提取绿色分量并显示分解图 -------- %
green(:,:,2) = A(:,:,2);
green(:,:,1) = zeros(r,c);
green(:,:,3) = zeros(r,c);
green = uint8(green);
subplot(2,2,3)
imshow(green)
title('绿色分量');
% --------- 提取蓝色分量并显示分解图 -------- %
blue(:,:,3) = A(:,:,3);
blue(:,:,1) = zeros(r,c);
blue(:,:,2) = zeros(r,c);
blue = uint8(blue);
subplot(2,2,4)
imshow(blue)
title('蓝色分量');
% ------------- 合成彩色图像 ------------ %
ci = cat(3,red(:,:,1),green(:,:,2),blue(:,:,3));
figure;
subplot(1,2,1)
imshow(A);
title('原始图像')
subplot(1,2,2)
imshow(ci);
title('合成图像');
```

由程序代码可知，彩色图像读入 MATLAB 中，被存为三维数组，红色分量存为第一页，绿色分量存为第二页，蓝色分量存为第三页。用三维数组提取和连接方法就能实现三种颜色分量的提取以及合成彩色图像，程序结果如图 3-1 和图 3-2 所示。

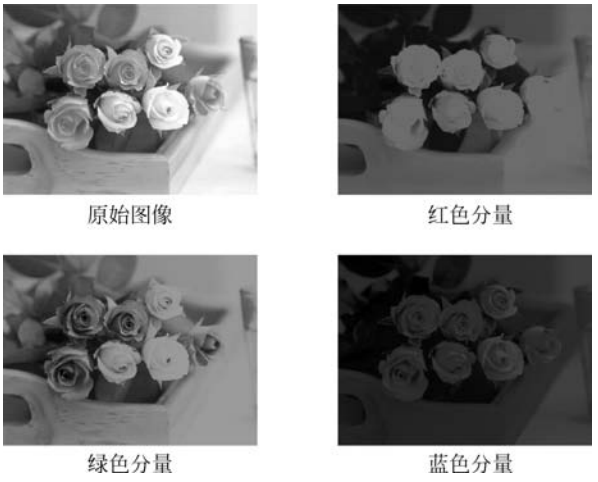

图 3-1　提取彩色图像各个分量

原始图像　　合成图像

图 3-2　原始图像和合成图像比较

3.3　结构数组

在 MATLAB 语言中，有两种复杂的数据类型，分别是结构数组(Structure Array)和元胞数组(Cell Array)，这两种类型都能在一个数组里存放不同类型的数据。

结构数组又称结构体，能将一组具有不同属性的数据放到同一个变量名下进行管理。结构体的基本组成是结构，每个结构可以有多个字段，可以存放多种不同类型的数据。

3.3.1　结构数组的创建

结构数组的创建方法有两种：直接创建和用 struct 函数创建。

(1) 直接创建。可以直接使用赋值语句，对结构数组的元素赋值不同类型的数据。具体格式：

```
结构数组名.成员名 = 表达式
```

例如，构建一个班级学生信息结构数组 dz1143，有三个元素 dz1143(1)、dz1143(2)和 dz1143(3)，每个元素有四个字段 Name、Sex、Nationality 和 Score，分别存放学生姓名、性别、国籍和成绩等信息。

程序代码如下：

```
>> dz1143(1).Name = 'Zhang san';
>> dz1143(1).Sex = 'Male';
>> dz1143(1).Nationality = 'China';
>> dz1143(1).Score = [98 95 90 99 87];
>> dz1143(2).Name = 'Li si';
>> dz1143(2).Sex = 'Male';
>> dz1143(2).Nationality = 'Japan';
>> dz1143(2).Score = [88 95 91 90 97];
>> dz1143(3).Name = 'Wang wu';
>> dz1143(3).Sex = 'Female';
>> dz1143(3).Nationality = 'USA';
>> dz1143(3).Score = [81 75 61 80 87];
>> dz1143
dz1143 =
包含以下字段的 1×3 struct 数组:
    Name
    Sex
    Nationality
    Score
```

其中，dz1143 是结构数组名，dz1143(1)、dz1143(2)和 dz1143(3)分别是结构数组的元素，Name、Sex、Nationality 和 Score 分别是字段。

(2) 利用 struct 函数创建结构数组。函数具体格式如下：

```
struct('field1','值 1', 'field2','值 2', 'field3','值 3',...)
```

例如：

```
>> dz1144(1) = struct('Name','Like','Sex','Male','Nationality','China','Score',[98,95,91,89])
dz1144 =
包含以下字段的 struct:
            Name: 'Li ke'
             Sex: 'Male'
    Nationality: 'China'
           Score: [98 95 91 89]
>> dz1144(2) = struct('Name','Xubo','Sex','Male','Nationality','Canada','Score',[99,97,95,92])
dz1144 =
包含以下字段的 1×2 struct 数组:
    Name
    Sex
    Nationality
    Score
```

3.3.2 结构体内部数据的获取

(1) 使用“.”符号获取结构体内部数据，对于 3.3.1 节例题中的 dz1143 结构体，用下面命令获得结构体的各个字段的内部数据：

```
>> str1 = dz1143(1).Name
str1 =
    'Zhang san'
>> str2 = dz1143(1).Sex
str2 =
    'Male'
>> str3 = dz1143(1).Nationality
str3 =
    'China'
>> S = dz1143(1).Score
S =
    98    95    90    99    87
```

(2) 使用函数 getfield 获取结构体内部数据，函数的格式如下：

```
str = getfield(S,{S_index},'fieldname',{field_index})
```

其中，S 是结构体名称，S_index 是结构体的元素，fieldname 是结构体的字段，field_index 是字段中数组元素的下标。

例如：

```
>> str1 = getfield(dz1143,{1},'Name')      % 获取 dz1143 结构体中第一个元素,
                                           % 字段为 Name 的内容
```

```
str1 =
    'Zhang san'
>> S1 = getfield(dz1143,{1},'Score',{2})   % 获取 dz1143 结构体中第一个
                                           % 元素,字段 Score 中第 2 门课成绩
S1 =
    95
```

(3) 使用函数 fieldnames 获取结构体所有字段,函数的格式如下:

```
x = fieldnames(S)
```

例如:

```
>> x = fieldnames(dz1143)                  % 获取结构体 dz1143 所有字段信息
x =
  4×1 cell 数组
    {'Name'        }
    {'Sex'         }
    {'Nationality'}
    {'Score'       }
>> whos dz1143 x                           % 查看结构体 dz1143 和变量 x 的属性信息
  Name         Size            Bytes  Class     Attributes
  dz1143       1x3             1720   struct
   x           4x1             462    cell
```

3.3.3 结构体的操作函数

(1) 可以使用函数 setfield 对结构体的数据进行修改,函数的格式如下:

```
S = setfiled(S,{S_index},'fieldname',{field_index},值)
```

例如,修改结构体 dz1143(1)中的 Sex 字段的内容:

```
>> dz1143 = setfield(dz1143,{1},'Sex','Female')       % 修改字段 Sex 内容
dz1143 =
包含以下字段的 1×3 struct 数组:
    Name
    Sex
    Nationality
    Score
```

(2) 可以使用 rmfield 函数删除结构体的字段,函数格式如下:

```
S = rmfield(S,'fieldname')
```

例如,删除结构体 dz1143 中的 Nationality 字段。

```
>> dz1143 = rmfield(dz1143,'Nationality')   % 删除字段 Nationality
dz1143 =
包含以下字段的 1×3 struct 数组:
    Name
    Sex
    Score
```

3.4 元胞数组

元胞数组是常规矩阵的扩展，其基本元素是元胞，每个元胞可以存放各种不同类型的数据，如数值矩阵、字符串、元胞数组和结构数组等。

3.4.1 元胞数组的创建

创建元胞数组的方法和一般数值矩阵方法相似，用大括号将所有元胞括起来。创建元胞数组方法有两种：直接创建和使用函数创建。

(1) 直接创建元胞数组。可以一次性输入所有元胞值，也可以每次赋值一个元胞值。

```
>> A = {[1 + 2i],'MATLAB 2020A';1:6,{[1 2;3 4],'cell'}}    % 一次性输入所有元胞值
A =
  2×2 cell 数组
    {[1.0000 + 2.0000i]}    {'MATLAB 2020A'}
    {1×6 double        }    {1×2 cell     }
>> B(1,1) = {[1 + 2i]};                                    % 每次输入一个元胞值
>> B(1,2) = {'MATLAB 2020A'};
>> B(2,1) = {1:6};
>> B(2,2) = {{[1 2;3 4],'cell'}}
B =
  2×2 cell 数组
    {[1.0000 + 2.0000i]}    {'MATLAB 2020A'}
    {1×6 double        }    {1×2 cell     }
```

另外，还可以根据各元胞内容创建元胞数组，例如：

```
>> C{1,1} = [1 + 2i];
>> C{1,2} = 'MATLAB 2020A';
>> C{2,1} = 1:6;
>> C{2,2} = {[1 2;3 4],'cell'}
C =
  2×2 cell 数组
    {[1.0000 + 2.0000i]}    {'MATLAB 2020A'}
    {1×6 double        }    {1×2 cell     }
```

由上面结果可知，用三种不同的直接输入法创建的元胞数组 A、B 和 C 结果是一样的。注意()和{ }的区别，创建元胞数组无论用哪种方法，等式的左边或者右边一般都需要使用一次{ }，除了元胞，因它是由元胞数组构成的，需要用两次{ }。

(2) MATLAB 语言可以使用 cell 函数创建元胞数组。函数格式如下：

```
A = cell(m,n)
```

cell 函数可以创建一个 m×n 空的元胞数组，对于每个元胞的数据还需要单独赋值。例如：

```
>> A = cell(2)
A =
```

```
  2×2 cell 数组
    {0×0 double}    {0×0 double}
    {0×0 double}    {0×0 double}
>> A{1,1} = [1 + 2i];
>> A{1,2} = 'MATLAB 2020A';
>> A{2,1} = 1:6;
>> A{2,2} = {[1 2;3 4],'cell'}
A =
  2×2 cell 数组
    {[1.0000 + 2.0000i]}    {'MATLAB 2020A'}
    {1×6 double         }    {1×2 cell     }
```

3.4.2 元胞数组的操作

在 MATLAB 中，创建元胞数组后，可以通过下面几种方法，引用和提取元胞数组元素数据。

(1) 用{ }提取元胞数组的元素数据。

例如：

```
>> A = {1 + 2i,'MATLAB 2020A';1:6,{[1 2;3 4],'cell'}}   % 创建 2×2 的元胞数组
A =
  2×2 cell 数组
    {[1.0000 + 2.0000i]}    {'MATLAB 2020A'}
    {1×6 double         }    {1×2 cell     }
>> a = A{2,1}                                           % 全下标提取元素
a =
     1     2     3     4     5     6
>> a = A{1,2}
a =
    'MATLAB 2020A'
>> a = A{4}                                             % 单下标提取元素
a =
  1×2 cell 数组
    {2×2 double}    {'cell'}
```

(2) 使用()只能定位元胞的位置，返回的仍然是元胞类型的数组，不能得到详细元胞元素数据，例如：

```
>> b = A(2,1)                                           % 全下标定位
b =
  1×1 cell 数组
    {1×6 double}
>> b = A(4)                                             % 半下标定位
b =
  1×1 cell 数组
    {1×2 cell}                                         % 元胞类型
```

(3) 用 deal 函数提取多个元胞元素的数据。

例如：

```
>> [c1,c2,c3] = deal(A{[1:3]})                    % 提取元胞数组 A 中第一到第三个元素
                                                  % 分别赋值给 c1、c2 和 c3
c1 =
   1.0000 + 2.0000i
c2 =
     1     2     3     4     5     6
c3 =
    'MATLAB 2020A'
>> [c1,c2,c3,c4] = deal(A{:,:})
c1 =
   1.0000 + 2.0000i
c2 =
     1     2     3     4     5     6
c3 =
    'MATLAB 2020A'
c4 =
  1×2 cell 数组
    {2×2 double}    {'cell'}
```

(4) 用 celldisp 函数显示元胞数组中详细数据内容。

在 MATLAB 命令窗口中,输入元胞数组名称,只显示元胞数组的各元素的数据类型和尺寸,不直接显示各元素的详细内容。可以用 celldisp 函数显示元胞数组中各元素的详细数据内容。

例如:

```
>> A = {1 + 2i,'MATLAB 2020A';1:6,{[1 2;3 4],'cell'}}     % 创建 2×2 的元胞数组
A =
2×2 cell 数组
    {[1.0000 + 2.0000i]}    {'MATLAB 2020A'}
    {1×6 double        }    {1×2 cell     }
>> A                                                      % 在命令窗口直接输入元胞数组名称
A =                                                       % 只显示各元胞的数据类型和尺寸
2×2 cell 数组
    {[1.0000 + 2.0000i]}    {'MATLAB 2020A'}
    {1×6 double        }    {1×2 cell     }
>> celldisp(A)                                            % 显示元胞数组各元胞的具体数据
A{1,1} =
    1.0000 + 2.0000i
A{2,1} =
     1     2     3     4     5     6
A{1,2} =
MATLAB 2020A
A{2,2}{1} =
     1     2
     3     4
A{2,2}{2} =
cell
```

(5) 用 cellplot 函数以图形方式显示元胞数组的结构。

在 MATLAB 中,可以用 cellplot 函数以图形方式显示元胞数组的结构。

例如，创建一个元胞数组，并用图形方式显示。

代码如下：

```
>> A = {1 + 2i,'MATLAB 2020A';1:6,{[1 2;3 4],'cell'}}      % 创建 2×2 的元胞数组
A =
  2×2 cell 数组
    {[1.0000 + 2.0000i]}    {'MATLAB 2020A'}
    {1×6 double        }    {1×2 cell     }
>> cellplot(A)
```

用 cellplot 函数显示元胞数组 A，结果如图 3-3 所示。用不同的颜色和形状表示元胞数组各元素的内容。

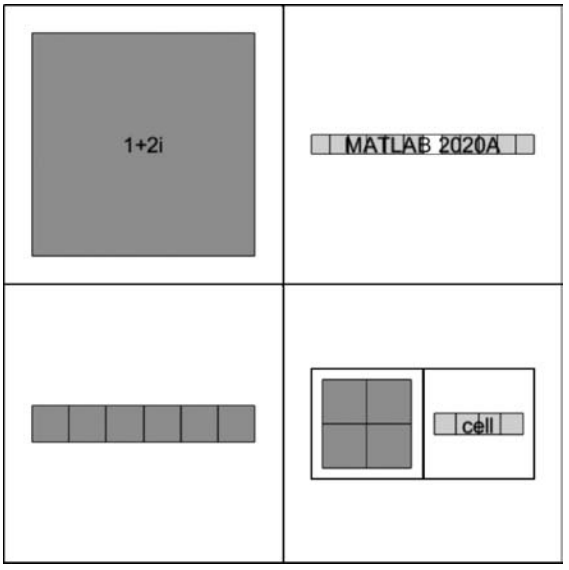

图 3-3　元胞数组显示图

习题

1. 定义两个字符串 str1＝'MATLAB　R2018a'和 str2＝'MATLAB　R2018A'，试用字符串比较函数 strcmp、strncmp、strcmpi 和 strncmpi 比较 str1 和 str2 两个字符串。

2. 在 MATLAB 语言中，建立下面的多维数组。

```
A(:,:,1) =
     0     0     0
     0     0     0
     0     0     0
A(:,:,2) =
     1     1     1
     1     1     1
     1     1     1
A(:,:,3) =
     1     0     0
```

```
     0     1     0
     0     0     1
```

3. 在 MATLAB 语言中,建立下面的结构数组。

```
dz1161 =
        Name: 'Li ke'
  Sex: 'Male'
  Province: 'Guangdong'
  Tel: '13800000000'
>> whos dz1161
  Name         Size             Bytes  Class     Attributes
  dz1161       1x1              762    struct
```