第 2 章　Matlab 数据处理

司守奎
烟台市，海军航空大学
Email：sishoukui@163.com
微信：sishoukui

2.1　数据组织

我们在组织数据时大多使用标准矩阵，这些矩阵是能有效存储大量数据对象的数据结构，如数值矩阵和字符矩阵。然而，这样的矩阵不适用于数据对象中既有数字也有字符串的情况。Matlab 软件提供的元胞数组和结构数组等数据结构能解决这个问题。

2.1.1　元胞数组

元胞数组是一种包含名为元胞的索引数据容器的数据类型，其中的每个元胞都可以包含任意类型的数据。元胞数组通常包含文本列表、文本和数字的组合或者不同大小的数值数组。通过将索引括在圆括号 () 中可以引用元胞集。使用花括号 {} 进行索引来访问元胞的内容。

当要将数据放入一个元胞数组中时，请使用元胞数组构造运算符 {} 创建该数组。

1.创建元胞数组

（1）预先分配空间

```
>>A=cell(2,3)
A =
  2×3 cell 数组
    {0×0 double}    {0×0 double}    {0×0 double}
    {0×0 double}    {0×0 double}    {0×0 double}
```

（2）直接赋值

```
>> B={[1,2,3,4],'Zhang San'}
B =
  1×2 cell 数组
    {[1 2 3 4]}    {'Zhang San'}
```

（3）单独赋值使用花括号{}

```
>> C=cell(2,3)
C =
  2×3 cell 数组
    {0×0 double}    {0×0 double}    {0×0 double}
    {0×0 double}    {0×0 double}    {0×0 double}
>> C{1}='Zhang San'
C =
  2×3 cell 数组
    {'Zhang San'}    {0×0 double}    {0×0 double}
    {0×0 double }    {0×0 double}    {0×0 double}
```

（4）整体赋值适用圆括号()

```
>> D=cell(2,3)
D =
  2×3 cell 数组
    {0×0 double}    {0×0 double}    {0×0 double}
    {0×0 double}    {0×0 double}    {0×0 double}
>> D(1,2:3)={[1,2,3,4],'Zhang San'}
D =
  2×3 cell 数组
    {0×0 double}    {[ 1 2 3 4]}    {'Zhang San'}
    {0×0 double}    {0×0 double}    {0×0 double }
```

2.索引元胞数组

（1）圆括号索引得到元胞集

```
B =
  1×2 cell 数组
    {[1 2 3 4]}    {'Zhang San'}
>> B(1,2)
ans =
  1×1 cell 数组
    {'Zhang San'}
```

（3）花括号索引得到元胞内容

```
>> B{1,2}
ans =
    'Zhang San'
```

例 2.1　元胞数组存取示例。

```
clc, clear
a={'张三','李四','王五';12,15,16}
b1=[a{2,:}]            %提出其中的数值矩阵
b2=cell2mat(a(2,:))  %提出其中的数值矩阵
[r1c1,r2c1,~,~,r1c3,r2c3]=a{:}  %逐列展开并赋值给其他变量
c=[a;{'M','M','F'}]  %元胞数组垂直方向串联
```

2.1.2　结构体数组

结构体数组（structure array）和元胞数组非常类似，因为它们都能将不同数据类型的数据组织在单一变量中。和元胞数组的不同之处在于，结构体数组的数据是由称作字段（field）的名称指定的，而不是由数字索引指定的，每一个字段都能包含任意类型和大小的数据。它使用圆点表示法而不是用花括号{}索引来访问其中的数据。

MATLAB 提供两种方法建立结构数组，用户可以直接给结构体数组字段赋值建立结构体数组，也可以利用函数 struct 建立结构体数组。

例 2.2　利用赋值建立结构体数组。

```
clc, clear
stu(1).name='LiMing'; stu(1).number='0101';
stu(1).sex='f'; stu(1).score=[90,80];
stu(2).name='LiHong'; stu(2).number='0102';
stu(2).sex='m'; stu(2).score=[88,80];
stu      %显示结构体数组的结构
stu(1)  %显示结构体数组第 1 个元素
stu(2)  %显示结构体数组第 2 个元素
```

例 2.3　利用函数 struct 建立结构体数组。

```
clc, clear
field1='Name'; value1={'张三','李四','王五'};
field2='Amount'; value2={15000,25000,50000};
field3='Data'; value3={[25,65;150,100],[2,8;78,90],[1,2;6,8]};
customers=struct(field1,value1,field2,value2,field3,value3)
customers(4).Name='刘六'    %增加一条记录
```

MATLAB 命令

```
f=dir('*.m')
```

可以显示当前目录下所有后缀名为 m 的文件信息，返回值 f 是一个结构数组，包括 5 个域：name，date，bytes，isdir，datenum；通过结构数组的元素个数就可以知道当前目录下 m 文件的个数，通过 name 域可以知道当前目录下所有 m 文件的名称.

dir 命令可以读出所有类型文件的信息。

例 2.4　读入附件 1 目录下所有的后缀名为 bmp 的图片文件，并把数据保存在元胞数组中。

```
clc, clear
f=dir('附件 1\*.bmp'); %读入所有 bmp 图像文件的信息，保存在结构数组 f 中
n=length(f)      %计算 bmp 文件的个数
for i=1:n
      a{i}=imread(['附件 1\',f(i).name]); %把第 i 个图像数据保存到元胞数组
end
```

2.1.3　table 类型

Matlab 中 table 数据类型适合表达表格数据，这些数据通常以列形式存储在文本文件或电子表格中。表中的每个变量可以具有不同的数据类型和大小，但每个变量必须具有相同的行数。table 数据类型的一个典型用法是存储试验数据，使用行表示不同的观测结果，使用列表示不同的测量变量。

硬盘上的文件与工作空间之间通过 readtable 和 writetable 两个函数进行交流。

table 数据类型的引用有三点要注意：

1）花括号{}，圆括号()，圆点.对 table 类型数据的作用都是引用数据，但是存在较为明显使用上的差异。

2）花括号{}的作用是{Rows,Columns}模式提取变量，形成数组。这里有一个基本要求，所有按照行列提取的数据要求是相互兼容的类型，不能一列是浮点数，另一列是字符串。

3）圆括号()的作用是(Rows,Columns)模式生成新的 table。注意 table 是一种新的数据类型，Matlab 定义的数学运算都是在数组的层面上进行的，所以如果要进行运算，需要用{}引用数据，但是要生成新表，需要用()。

4）圆点.引用数据每次只能引用一列。

例 2.5　table 类型数据操作示例。

```
clc, clear
load hospital    %加载 Matlab 内置的数据文件 hospital.mat
Age=hospital.Age;
Weight=hospital.Weight;
TP=table(Age,Weight)      %构造新的表
summary(TP)               %对表中数据进行统计
```

2.1.4　分类数组

分类数组（categorical array）旨在存储具有以下特征的数据：此类数据的值只能来自一个包含离散类别的有限集合。如果分类数组中记录的属性是离散的，有不能排序的值，就说它是无序分类的（categorical unordered）。这些属性被赋值，除了能够区分类别，没有其他意义（如男人、女人这两个类别）。相应地，如果分类数组中记录的属性是离散的，能够进行排序，就说它是有序分类的（categorical ordered）。这些属性被赋值且能够比较它们之间的大小关系（如 1、2、3/4）。

可以使用函数 categorical 从一个数值数组、逻辑数组、元胞数组、字符数组或是一个已经存在的分类数组中创建分类数组。

例 2.6　指定整数的类别名称。

```
A = [1 3 2; 2 1 3; 3 1 2];
%将 A 转换为分类数组 B 并指定类别名称
```

```
B = categorical(A,[1 2 3],{'red' 'green' 'blue'})
categories(B)  %显示 B 的类别
```

例 2.7　创建有序分类数组。

```
A = [3 2;3 3;3 2;2 1;3 2]
%将 A 转换为有序分类数组，其中 1、2 和 3 分别表示类别 child、adult 和 senior
valueset = [1:3];
catnames = {'child' 'adult' 'senior'};
B = categorical(A,valueset,catnames,'Ordinal',true)
```

例 2.8　转换字符串数组。

从 Matlab2017a 开始，可以使用双引号来创建字符串数组。而且，字符串可以包含缺失值，显示为<missing>，不带双引号。

```
str = ["plane","jet","plane","helicopter",missing,"jet"]
C = categorical(str)
```

例 2.9　将数值数据分组为类别。

使用 discretize 函数（而不是 categorical）将 100 个随机数划分为三个类别。

```
rng(2)   %进行一致性比较
x = rand(100,1);
y = discretize(x,[0 0.25 0.75 1],'categorical',{'small','medium','large'});
summary(y)
```

2.1.5　稀疏矩阵

所谓的稀疏矩阵就是不存储矩阵中的零元素，而指对非零元素进行操作，这样就大大减少了存储空间和计算时间。

1.稀疏矩阵的建立

在 Matlab 中，稀疏矩阵是用特殊的命令创建的，在运算中，Matlab 对稀疏矩阵采取不通于满矩阵的算法进行各种计算。

与稀疏矩阵有关的函数如表 2.1 所示。

表 2.1　与稀疏矩阵有关的函数

函数名	功能
sparse	构造稀疏矩阵
spdiags	提取非零对角线并创建稀疏带状对角矩阵
spconvert	从外部稀疏矩阵形式输入
speye	稀疏单位矩阵
sprand	稀疏均匀分布随机矩阵
sprandn	稀疏正态分布随机矩阵
sprandsym	稀疏对称随机矩阵
full	变换稀疏矩阵为满矩阵

下面对几个常用的函数做重点介绍。

（1）sparse

函数 sparse 的调用格式为：

```
S = sparse(A)   %把满矩阵转换为稀疏矩阵 S
S = sparse(i,j,v,m,n)   %生成 m×n 的稀疏矩阵 S，使得 S(i(k),j(k)) = v(k)
```

例 2.10　构造如下形式的稀疏矩阵（以 $n=5$ 为例）。

$$\boldsymbol{A}=\begin{bmatrix}4 & 1 & & \\ 1 & 4 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 4\end{bmatrix}_{n\times n}.$$

```
clc, clear, n=5;
a1 = sparse(1:n,1:n,4*ones(1,n),n,n);
a2 = sparse(2:n,1:n-1,ones(1,n-1),n,n);
a = a1 + a2 + a2'
b = full(a)
```

（2）spdiags

函数 spdiags 的调用格式为：

```
B = spdiags(A)              %提取由 d 指定的对角矩阵。
S = spdiags(Bin,d,A)        %把 Bin 中的列代替由 d 指定的 A 中对角线位置。
S = spdiags(Bin,d,m,n)      %把 Bin 中的列放在由 d 指定的对角线位置生成 m×n 稀疏矩阵 S。
```

例 2.11　利用 spdiags 函数生成一下矩阵的稀疏矩阵（以 $n=5$ 为例）。

$$\boldsymbol{A}=\begin{bmatrix}4 & 1 & & \\ 1 & 4 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 4\end{bmatrix}_{n\times n}.$$

```
clc, clear, n=5;
e = ones(n,1);
a = spdiags([e,4*e,e],-1:1,n,n)
b = full(a)
```

2.稀疏矩阵的运算

同满矩阵比较起来，稀疏矩阵在算法上有很大的不同。具体表现在存储空间减少，计算时间减少。

例 2.12　比较求解下面方程组 $n=1000$ 时的两种方法的差别。

$$\begin{bmatrix}4 & 1 & & \\ 1 & 4 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 4\end{bmatrix}_{n\times n}\begin{bmatrix}x_1\\ x_2\\ \vdots\\ x_n\end{bmatrix}=\begin{bmatrix}1\\ 1\\ \vdots\\ 1\end{bmatrix}.$$

```
clc, clear, n=1000;
e = ones(n,1); c = ones(n,1);
a = spdiags([e,4*e,e],-1:1,n,n)
b = full(a)
tic, x1 = a\c; toc
tic, x2 = b\c; toc
```

2.2　数据导入和导出

数据分析的第一步是将收集的数据导入 Matlab 中，根据数据的不同格式，有相对应的不同方法在 Matlab 中导入数据。

2.2.1　导入向导

我们既可以交互式地向 Matlab 中导入数据，也可以通过程序语句导入数据。交互式地导入数据时，数据可以来源于硬盘上的文件或者系统剪贴板。

从硬盘的文件中导入数据可以使用如下方式的任意一种。

（1）在“主页”（Home）选项卡的“变量”（Variable）功能区中，单击“导入数据”（Import Data）。

（2）在命令窗口中调用函数 uiimport。

对初学者而言，导入向导非常有用，它能根据数据的性质提供不同的导入方式，在数据导入过程中给予各种帮助。我们能够从多种类型文件（包括图片、音频、视频数据等）中进行导入。导入向导能够显示文件内容，可供用户选择导入的数据，去掉不需要的数据。

在导入向导中使用的命令可以被存入一个 Matlab 函数或脚本文件中。用户可以保存这个文件，在导入相似的文件时，可以重复使用它。

csv(Comma-Separated Values，CSV)文件，也称为逗号分隔值文件，其文件以文本形式存储表格数据（数字和文本），文件的每一行都是一个数据记录。每个记录由一个或多个字段组成，用逗号分隔。使用逗号作为字段分隔符是此文件格式的名称的来源，因为分隔字符也可以不是逗号，有时也称为字符分隔值。

例 2.13　导入“共享汽车定位数据.csv”中的数据。

通过导入工具导入 CSV 文本文件的步骤如下：

（1）通过单击“导入数据”按钮，打开“导入数据”对话框，然后双击要导入的文件“共享汽车定位数据.csv”，打开如图 2.1 所示的界面。

（2）设置列分隔符，选定导入工作空间的数据范围，最后点击“导入所选内容”选项卡，把数据导入工作空间或生成导入数据的脚本。

确保变量被正确导入 Matlab 后，用户就能使用这些变量进行运算或操作了。

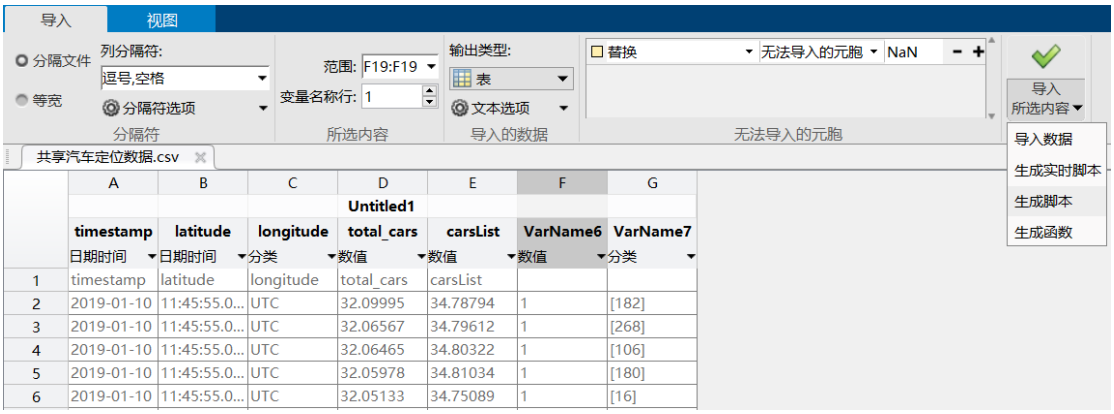

图 2.1　导入工具界面

2.2.2　通过程序语句导入数据

根据需要导入的数据类型，常使用表 2.2 中的函数读取文件中的数据。

表 2.2　Matlab 读入文件数据的函数

输出的数据类型	函数
时间表格（Timetable）	readtimetable
数值矩阵（Numeric Matrix）	readmatrix
元胞数组（cell Array）	readcell
分离的列向量（Separate Column Vectors）	readvars

例 2.14　读入如图 2.2 所示格式的 csv 文件。

	A	B	C	D	
1	timestamp	latitude	longitude	total_cars	carsList
2	2019-01-10 11:45:55.070781 UTC	32.09995	34.78794	1	[182]
3	2019-01-10 11:45:55.070781 UTC	32.06567	34.79612	1	[268]
4	2019-01-10 11:45:55.070781 UTC	32.06465	34.80322	1	[106]
5	2019-01-10 11:45:55.070781 UTC	32.05978	34.81034	1	[180]
6	2019-01-10 11:45:55.070781 UTC	32.05133	34.75089	1	[16]
7	2019-01-10 11:45:55.070781 UTC	32.04223	34.7742	1	[72]
8	2019-01-10 11:45:55.070781 UTC	32.04156	34.77128	1	[160]
9	2019-01-10 11:45:55.070781 UTC	32.12373	34.81346	1	[210]
10	2019-01-10 11:45:55.070781 UTC	32.11874	34.83406	1	[136]

图 2.2　共享汽车定位数据.csv 中的数据格式

```
%程序文件 ex2_14_1.m
clc, clear
opts = delimitedTextImportOptions('VariableNames',{'timestamp',...
    'latitude','longitude','total_cars','carsList'},...
    'VariableTypes',{'char','double','double','int32','categorical'},...
    'Delimiter',',','DataLines', [2 Inf],'ExtraColumnsRule','ignore');
[t,latitude,longitude,total,List]=readvars('共享汽车定位数据.csv',opts);

%程序文件 ex2_14_2.m
clc, clear
[a,b0]=xlsread('共享汽车定位数据.csv','A2:E1048576');
b1=b0(:,1);
for i=1:length(b1)
    b2(i)=datenum(b1{i}(1:end-4));   %删除时间后面的字符并转换为数值型
end
save data a b2

程序文件 ex2_14_3.m
clc, clear
opts = delimitedTextImportOptions('VariableNames',{'timestamp',...
    'latitude','longitude','total_cars','carsList'},...
    'VariableTypes',{'char','double','double','int32','categorical'},...
    'Delimiter',',','DataLines', [2 Inf],'ExtraColumnsRule','ignore');
a=readtable('共享汽车定位数据.csv',opts);
```

例 2.15　CSV 文件 outages.csv（Matlab 内置文件）中存放如下格式的数据：

```
Region,OutageTime,Loss,Customers,RestorationTime,Cause
SouthWest,2002-01-20 11:49,672,2902379,2002-01-24 21:58,winter storm
SouthEast,2002-01-30 01:18,796,336436,2002-02-04 11:20,winter storm
SouthEast,2004-02-03 21:17,264.9,107083,2004-02-20 03:37,winter storm
West,2002-06-19 13:39,391.4,378990,2002-06-19 14:27,equipment fault
```

```
%程序文件 ex2_15_1.m
formatSpec = '%C%{yyyy-MM-dd HH:mm}D%f%f%{yyyy-MM-dd HH:mm}D%C';
T = readtable('outages.csv','Format',formatSpec);
```

```
head(T,5)

%程序文件 ex2_15_2.m
opts = detectImportOptions('outages.csv');
varNames = opts.VariableNames
varTypes = {'categorical','datetime','double',...
'double','datetime','categorical'}
opts = setvartype(opts,varNames,varTypes);
T = readtable('outages.csv',opts);
head(T,5)
T.Duration = T.RestorationTime - T.OutageTime; %增加新字段
head(T,5)
```

例 2.16　文本文件 bigfile.txt（Matlab 内置文件）存放如下格式的数据：

```
## A ID = 02476
## YKZ Timestamp Temp Humidity Wind Weather
06-Sep-2013 01:00:00 6.6 89 4 clear
06-Sep-2013 05:00:00 5.9 95 1 clear
06-Sep-2013 09:00:00 15.6 51 5 mainly clear
06-Sep-2013 13:00:00 19.6 37 10 mainly clear
06-Sep-2013 17:00:00 22.4 41 9 mostly cloudy
06-Sep-2013 21:00:00 17.3 67 7 mainly clear
## B ID = 02477
## YVR Timestamp Temp Humidity Wind Weather
09-Sep-2013 01:00:00 15.2 91 8 clear
09-Sep-2013 05:00:00 19.1 94 7 n/a
09-Sep-2013 09:00:00 18.5 94 4 fog
09-Sep-2013 13:00:00 20.1 81 15 mainly clear
09-Sep-2013 17:00:00 20.1 77 17 n/a
09-Sep-2013 18:00:00 20.0 75 17 n/a
09-Sep-2013 21:00:00 16.8 90 25 mainly clear
## C ID = 02478
## YYZ Timestamp Temp Humidity Wind Weather
```

```
clc, clear, N = 12;
formatSpec = '%D %f %f %f %c';
fileID = fopen('bigfile.txt');
C = textscan(fileID,formatSpec,N,'CommentStyle','##',...
    'Delimiter','\t','DateLocale','en_US')
```

注 2.1　上述程序有问题，相当于行与行之间加入了空行。

例 2.17　读入附件 2 下所有文本文件数据。

```
clc, clear
fi=dir('附件 2\*.xyz'); n=length(fi);
```

```
D1=[]; D2=[];
for i=1:n
    f=fopen(['附件 2\',fi(1).name]);
    d1=textscan(f,'%d',1);
    D1=[D1,d1{1}];
    d2=textscan(f,'%s%s%f',1);
    D2=[D2,d2{3}];
    d=textscan(f,'%s%f%f%f','CollectOutput',1);
    D3{i}=d{2};
    fclose(f);
end
```

2.2.3 把数据写入文本文件

1.把 table 类型数据写入文本文件

例 2.18 把 table 类型写入文本文件示例。

```
clc, clear
Pitch = [0.7;0.8;1;1.25;1.5];
Shape = {'Pan';'Round';'Button';'Pan';'Round'};
Price = [10.0;13.59;10.50;12.00;16.69];
Stock = [376;502;465;1091;562];
T = table(Pitch,Shape,Price,Stock)
writetable(T,'table2_18_1.csv')
rowNames = {'M4';'M5';'M6';'M8';'M10'};
T2 = table(Pitch,Shape,Price,Stock,'RowNames',rowNames)
writetable(T2,'table2_18_2.csv','WriteRowNames',true)
```

2.把元胞数组数据写入文本文件

例 2.19 把元胞数组数据写入文本文件示例。

```
clc, clear
C = {'Atkins',32,77.3,'M';'Cheng',30,99.8,'F';'Lam',31,80.2,'M'};
writecell(C,'data2_19.csv')
```

3.把数值数组数据写入文本文件

例 2.20 把数值数组数据写入文本文件示例。

```
clc, clear
A = magic(5)/10
writematrix(A,'data2_20.csv')
```

2.2.4 Excel 文件存取

读入 Excel 文件也可以使用函数 readtable, readmatrix,readcell。把数据写入 Excel 文件也同样使用函数 writetable，writematrix，writecell。

例 2.21 **生成随机数矩阵，然后写入 Excel 文件。**

```
clc, clear, rng(1)      %进行一致性比较
A = randi([1,100],8,5) %生成 8×5 的随机整数矩阵
```

```
B = array2table(A)      %把矩阵转换为 table 数据
C = normrnd(0,1,6,3)    %生成 6×3 的标准正态分布随机数矩阵
writematrix(A,'data2_21_1.xlsx')
writetable(B,'data2_21_2.xlsx')
warning('off')                                  %关闭新建表单的警告信息
writematrix(C,'data2_21_2.xlsx','Sheet',2)  %写入第 2 个表单中
```

例 2.22　读入 Excel 文件 data2_21_2.xlsx 中的所有数据。

```
clc, clear
a = readtable('data2_21_2.xlsx');  %读入第 1 个表单的数据
b1 = a{:,:}                                %table 数据转换为矩阵数据
b2 = readmatrix('data2_21_2.xlsx','NumHeaderLines',1) %再读入第 1 个表单
c = readmatrix('data2_21_2.xlsx','Sheet',2)            %读入第 2 个表单
```

2.3　数据整理

收集好数据之后，还需要对数据进行整理，这是一个艰巨的过程，会耗费很长时间，有时候甚至直接占据整个数据分析时间的 80%。然而，这是进行接下来的数据分析过程的一个基本前提条件。

2.3.1　初步查看数据

在将数据进行分析处理之前，我们需要大致查看一下将什么样的数据导入到了 Matlab 中，是否存在一些问题。通常，原始数据是很杂乱的，并且格式也很不规范。

第一步是将数据排序，这会让数据清洗更加容易。但是，什么时候可以说数据是干净的？根据 Edgar F. Codd 的建议，如果一个数据集满足以下条件，那么就说它是干净的。

- 每行是观测结果。
- 每列是观测变量。
- 数据是在单一的数据集中。

除此之外，我们还能够从哪个原始数据中发现什么？在收集数据过程中，可能会出现如下所示的问题。

- 一个表格中拥有不合理的且观测现象中并不存在的数据类型。
- 单个观测现象被存储在多张表中。
- 列标题不包含变量名。
- 某一列中包含多个变量的内容。
- 变量的观测值有的是按行存储的，但有的是按列存储的。

例 2.23　查看数据文件 data2_23.xlsx。

数据文件 data2_23.xlsx 中的数据如表 2.3 所示。

表 2.3　data2_23.xlsx **中的数据**

name	gender	age	right	wrong
Emma	F	24	80	20
Liam	M	-19	.	47
Olivia		32	75	25
Noah	M	15	60	40
Ava	F	18	45	55
Mason	M	21	54	46
Isabella	F	28	-19	85
Lucas	M	30	13	87
Sophia	F	26	NaN	30

Elijah	M	100	98	2
Mia	F	22	5	95
Oliver	M	NA	NaN	21

```
clc, clear
a = readtable('data2_23.xlsx')
summary(a)
```

数据读入 Matlab 后的显示格式如下：

```
a =
  12×5 table
      name           gender         age     right     wrong
    ____________    __________    ___    _____    _____
    {'Emma'    }    {'F'     }     24       80       20
    {'Liam'    }    {'M'     }    -19      NaN       47
    {'Olivia'  }    {0×0 char}     32       75       25
    {'Noah'    }    {'M'     }     15       60       40
    {'Ava'     }    {'F'     }     18       45       55
    {'Mason'   }    {'M'     }     21       54       46
    {'Isabella'}    {'F'     }     28      -19       85
    {'Lucas'   }    {'M'     }     30       13       87
    {'Sophia'  }    {'F'     }     26      NaN       30
    {'Elijah'  }    {'M'     }    100       98        2
    {'Mia'     }    {'F'     }     22        5       95
    {'Oliver'  }    {'M'     }    NaN      NaN       21
```

可以看到数据中存在如下问题。

- 空的元胞。
- 包含非数的数据 NaN。
- 包含负数（-19）。

2.3.2 缺失值的处理

使用函数 ismissing 可以快速找到包含缺失值的观测结果。注意，函数 ismissing 对不同的数据类型有默认的缺失值指示符。

- 数值矩阵为 NaN。
- 字符矩阵为''。
- 分类数组为<undefined>。
- 日期数组为 NaT。

函数 standardizeMissing 可以将数组或表中由某个其他指示符指示的缺失值替换为标准缺失值。

函数 fillmissing 可以使用各种方法对缺失值进行填充。

函数 rmmissing 可以移除矩阵或表中包含缺失值的记录。

例 2.24　对数据文件 data2_23.xlsx 中的缺失值进行处理。

```
clc, clear
a = readtable('data2_23.xlsx')
id = {NaN,'',-19};
WrongPos = ismissing(a, id)
```

```
WrongData = a(any(WrongPos,2),:)
a = standardizeMissing(a,-19)
b = fillmissing(a,'previous')   %使用缺失值前一行的值来填充缺失值。
c = rmmissing(a)                %删除包含缺失值的记录
summary(c)
writetable(c, 'data2_24.xlsx') %保存到 Excel 文件供下面使用
```

2.3.3 改变数据类型和表格排序

在例 2.24 的表格数据中，观测函数 summary 返回的摘要报告，可以看到 gender 返回的是字符细胞数组，而 gender 实际上是分类数组。

根据指定的列变量对表格中的元素进行排序，有利于观测一些数据特征。

例 2.25　改变数据类型和表格排序示例。

```
clc, clear
a = readtable('data2_24.xlsx')
a.gender = categorical(a.gender)      %gender 转换类型
summary(a)
b = sortrows(a,{'age'},{'descend'})   %age 数据降序排序
writetable(b,'data2_25.xlsx')         %数据保存到 Excel 文件供下面使用
```

2.3.4 找数据中的异常值

异常值是与其他值相比特别极端的值。异常值的存在会扭曲数据分析的结果，尤其是在描述性统计和相关性的分析中。异常值的识别可以放在之前的数据清洗阶段进行，也可以放在数据清洗之后的步骤中进行。当只研究一个变量的极端值时，异常值是一元异常值；当研究几个变量值的组合时，出现的异常组合被称作多元异常值。

在 Matlab 中，可以使用函数 isoutlier 识别数据中的异常值。

例 2.26　找数据中异常值示例。

```
clc, clear
a = readtable('data2_25.xlsx')
b = isoutlier(a(:,[3:5]))
ind = find(any(b,2))                  %找异常值所在的行数
```

2.3.5 将多个数据源合并成一个数据源

例 2.27　多个数据源合并示例。

我们有中国不同性别期望寿命的数据，将其存储在文件 data2_27.xlsx 中，数据如表 2.4 所示，与 data2_25.xlsx 中的数据合并。

表 2.4　中国不同性别期望寿命数据

state	Le	gender
China	80.91	M
China	86.58	F

```
clc, clear
a = readtable('data2_25.xlsx')
b = readtable('data2_27.xlsx')
c = join(a,b,'Keys','gender')    %合并两个数据源的数据
```

```
d = [c; c]    %两份同样类型的样本数据合并
e = unique(d,'rows')    %去掉重复数据
```

习题 2

2.1　数据文件“2017 第十届华中地区数学建模邀请赛_经典赛 B 题_附件_通话记录.xlsx”取自 2017 年第 10 届华中地区大学生数学建模邀请赛 B 题：基于通信数据的社群聚类。该文件包括某营业部近三个月的内部通信记录，内容涉及通话的起始时间、主叫、时长、被叫、漫游类型和通话地点等，共 10713 条记录，每条数据有 7 列，部分数据如表 2.5 所列。

表 2.5　某营业部近三个月的内部通信记录

序号	起始时间	主叫	时长（秒）	被叫	漫游类型	通话地点
1	2016/09/01 10:08:51	涂蕴知	431	孙翼茜	本地	武汉
2	2016/09/01 10:17:37	毕婕靖	351	潘立	本地	武汉
3	2016/09/01 10:18:29	张培芸	1021	梁茵	本地	武汉
4	2016/09/01 10:23:22	张培芸	983	文芝	本地	武汉
⋮	⋮	⋮	⋮	⋮	⋮	⋮
10713	2016/12/31 9:36:15	柳谓	327	张荆	本地	武汉

（1）主叫和被叫分别有多少人，主叫和被叫是否是同一组人。

（2）统计主叫和被叫之间的呼叫次数和总呼叫时间。

（3）将日期中“2016/09/01”视为第 1 天，“2016/09/02”视为第 2 天，依此类推，将所有日期按上述方法转化。

（4）已知 2016/09/01 为星期四，将日期编码为数字。编码规则为：星期日对应“0”，星期一对应“1”，…，星期六对应“6”。

（5）假设周六和周日不上班，不考虑法定节假日，周日上班时间为上午 8:00~12:00 和下午 14:00~18:00。计算任意两人在上班时间的通话次数。