Matlab 基础知识

李志明

天津大学应用数学中心 & 天津大学数学系

2014 年 11 月 19 日

目 录

1 Matlab 简介 **1**

1.1 Desktop 操作桌面 1

1.2 MATLAB 语言的特点 1

2 变量、数值数组及其运算 **3**

2.1 变量命名规则 3

2.2 基本运算及优先级 3

2.3 数值数组的创建和寻访 4

2.3.1 一维数组的创建 4

2.3.2 二维数组的创建 5

2.3.3 二维数组元素的编址和寻访 6

2.3.4 数组构做技法综合 6

2.3.5 高维数组 7

2.4 数值数组运算 8

3 字符串、元胞和构架数组 **8**

3.1 MATLAB 的数据类型 9

3.2 字符串数组 9

3.2.1 数组的属性和标识 10

3.2.2 复杂串数组的创建 10

3.2.3 串操作函数 10

3.3 元胞数组 11

3.3.1 元胞数组的创建和显示 11

3.3.2 元胞数组的扩充、收缩和重组 12

3.3.3 元胞数组内容的获取和配置 12

3.4 构架数组 12

3.4.1 构架数组的创建和显示 13

3.4.2 构架数组域中内容的调取和设置 14

4 MATLAB 控制流 **14**

4.1 for 循环和 while 循环结构 14

4.2 if-else-end 条件分支结构 15

4.3 switch-case 切换分支结构 15

4.4 try-catch 容错结构 16

4.5 编程用的其他指令 16

5 M 函数文件 **16**
5.1 M 脚本文件和 M 函数文件 16
5.2 函数文件的一般结构 17
5.3 主函数和子函数 18
5.3.1 主函数 18
5.3.2 子函数 18
5.4 函数的变量 19
5.4.1 局部变量与全局变量 19
5.4.2 变量的检测 19

6 图像相关操作 **19**
6.1 图像读、写及显示 21
6.1.1 基本图像操作函数 21
6.1.2 帧图像显示 22
6.2 图像格式转换 22
6.3 图像的标注 22

7 其他常用指令 **22**
7.1 几类常用指令 22
7.2 常见问题 23

1 Matlab 简介

MATLAB（MATrix LABoratory，即矩阵实验室）是 MathWork 公司推出的一套高效率的数值计算和可视化软件。MATLAB 是当今科学界最具影响力、也是最具活力的软件之一，它起源于矩阵运算，并已经发展成一种高度集成的计算机语言。它提供了强大的科学运算、灵活的程序设计流程、高质量的图形可视化与界面设计、便捷的与其他程序和语言接口的功能。

1.1 Desktop 操作桌面

MATLAB 的 Desktop 操作桌面是一个高度集成的 MATLAB 工作界面。R2014 支持简体中文，其工作界面的默认形式如图 1-1 所示。该桌面的上层铺放着五个最常用的界面：命令窗口（Command Window）、编辑器（Editor）、当前目录（Current Folder）浏览器、工作空间（Workspace）、历史命令窗口（Command History）。

在以前的版本中，如 R2010a，R2011a 等，编辑器和工作界面分离，一个 M 文件占用一个编辑器窗口。现在的版本编辑器融入主界面，且多个 M 文件以标签形式排列在窗口中，当然也可以分离出来。工作空间中列出所有的变量名、大小、类型和最值等；也可以对变量进行观察、图示、编辑、提取和保存。

1.2 MATLAB 语言的特点

(1) 编程简单使用方便

MATLAB 的基本数据单元既不需要指定维数、也不需要说明数据类型的矩阵，而且数学表达式和运算规则与通常的习惯相同。因此，在 MATLAB 环境下，数组的操作与数的操作一样简单。

MATLAB 的矩阵和向量操作功能是其他语言无法比拟的。在编程实践时常会看到提示（如对一个矩阵循环赋值时），要求指定变量的维数以加快运行速度。

(2) 函数库可任意扩充

由于 MATLAB 语言库函数与用户文件的形式相同，用户文件可以像库函数一样随意调用。所以用户可根据自己的需要任意扩充函数库。

(3) 语言简单内涵丰富

MATLAB 语言中最重要的成分是函数，其一般形式为：

$$\text{function} \quad [a,b,c,\ldots] = \text{functionname}\,(d,e,f,\ldots) \tag{1}$$

其中，functionname 是自定义的函数名，只要不与库函数名相重，并且符合字符串的书写规则即可。这里的函数既可以是数学上的函数，也可以是程序块或子程序，内涵

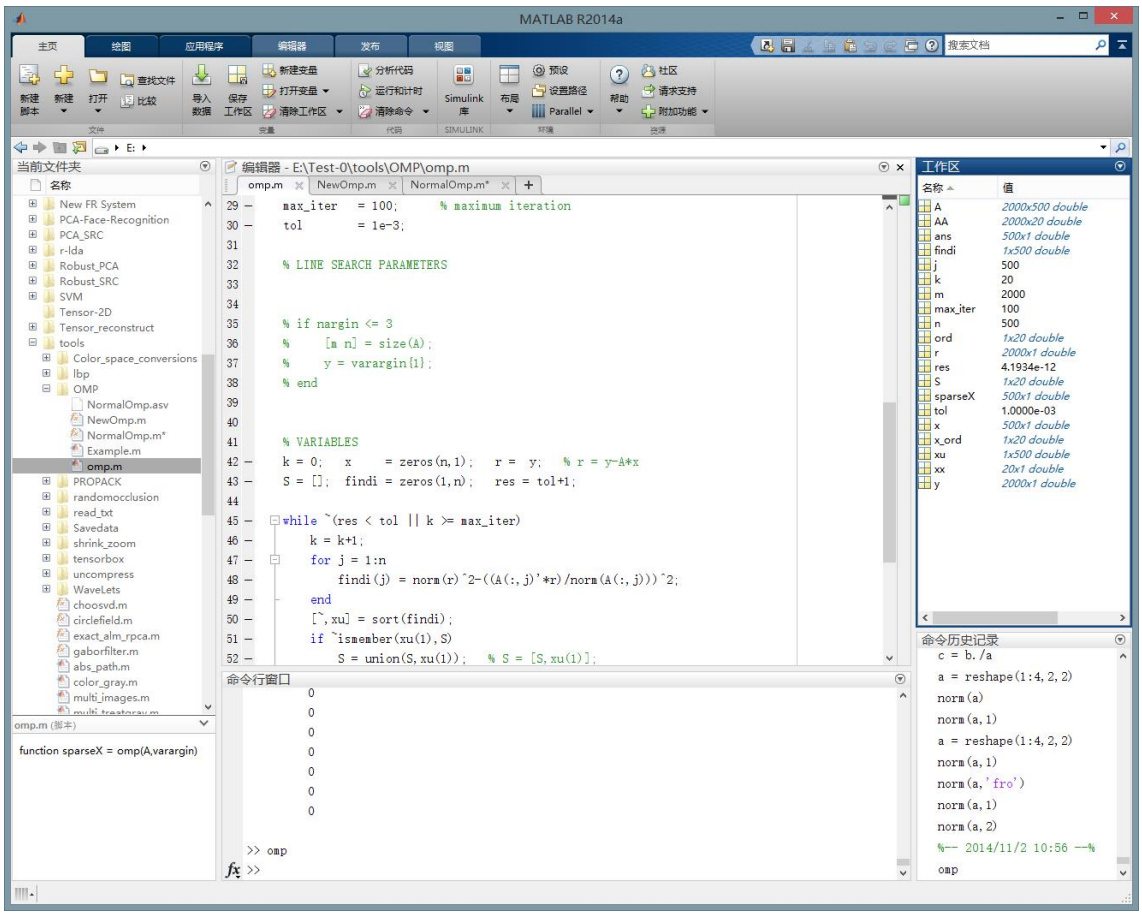


图 1-1: **Desktop 操作桌面**

十分丰富。每个函数建立一个同名的 M 文件，如上述函数的文件名为 functionname.m。这种文件简单、短小、高效，并且便于调试。

(4) 简便的绘图功能

MATLAB 具有二维和三维绘图功能，使用方法十分简便。而且用户可以根据需要在坐标图上加标题。坐标轴标记。文本注释及网格等，也可以指定图线形式（如实线、虚线，带 *、△ 等）和颜色，也可以在同一张图上画不同函数的曲线，对于曲面图还可以画出等高线。

(5) 丰富的工具箱

由于 MATLAB 的开放性，许多领域的专家都为 MATLAB 编写了各种程序工具箱，如 tensor_toolbox，dipum_toolbox（数字图像处理(matlab 版)-冈萨雷斯一书提供 dipum_toolbox_pcode.zip）。

各种工具箱提供了用户在特别应用领域所需的许多函数，这使得用户不必花大量的时间编写程序就可以直接调用这些函数，达到事半功倍的效果。

2 变量、数值数组及其运算

2.1 变量命名规则

△ Matlab 的变量名、函数名区分大小写字母，如 TrainSample 和 trainsample 表示了两个不同变量，imshow 是 Matlab 定义的显示图片的函数名，但 IMshow、ImShow 等都不是；

△ 变量名的首字符必须是英文字母，最多可包含 63 个字符（英文、数字和下连符）；

△ 变量名不得包含空格、标点、运算符，如 tr_num 是合法的，但 tr num 就不是一个变量名；

△ 定义变量名的禁忌：不与 MATLAB 关键词（如 for，if，else，end 等）同名；不与自带变量名（如，eps，pi 等）、函数名（如 imshow，eig等）、文件夹名（如 toolbox 等）相同；为此有判断指令：

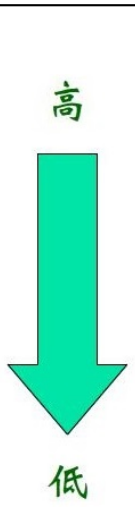
◊ iskeyword MyImage 若运行结果为 0，则不是关键词，可用；

◊ exist MyImage 若运行结果为 0，则不同于自带变量名、函数名、文件夹名，可用。

2.2 基本运算及优先级

MATLAB 表达式基本运算符及其优先级见表 2-1 。

表 2-1: 基本运算符及优先级

名称	数学表达	运算符	优先级
括号	( )	( )	
幂，点幂	a^b	a ^ b	
正号，负号，逻辑非	+, −, ~		
乘，除，点乘，点除	×, ÷	*, .*, \, .\ 或 /, ./	
加，减	+, −	+, −	
冒号运算	:	:	
关系运算	<, <=, >, >=, ==, ~=	<, <=, >, >=, ==, ~=	
逻辑运算		&	
		|	
		&&	
		||	

2.3 数值数组的创建和寻访

鉴于数值数组在 MATLAB 中的基础地位，本节专门讲述数组（包括矩阵）的创建、编址和寻访。

2.3.1 一维数组的创建

就所创建一维数组的用途而言，大致分为两类：自变量数组和通用变量数组，算例演示参见 myvector.m。

1、递增/递减型一维数组的创建

这类数组的特点：数值元素的值的大小按递增或递减的次序排列；数值的元素值之间的差是确定的，即“等步长”的。这类数组主要用作函数的自变量和 for 循环中循环变量等。

(1) 冒号生成法

$x = a : inc : b$

说明

- ♡ a 为左端点，inc 为步长。若 $(b-a)$ 是 inc 的整数倍，则右端点为 b，否则，右端点的绝对值小于 b。
- ♡ 冒号必须用英文状态输入。
- ♡ inc 缺省时，默认 $inc = 1$。
- ♡ inc 可正可负，产生相应增/减数组。

(2) 线性/对数定点法

$x = \text{linspace}(a, b, n)$

说明 以 a, b 为左右端点，线性等间隔 $(1 \times n)$ 数组，等价于 $x = a : (b-a)/(n-1) : b$。

$x = \text{logspace}(a, b, n)$

说明 以 a, b 为左右端点，对数等间隔 $(1 \times n)$ 数组，即产生的数组每一个元素取对数后是 a, b 间线性等间隔数组，等价于 $x = 10.\hat{}\ \text{linspace}(a, b, n)$。

2、其他类型一维数组的创建

(1) 逐个元素输入法

这是最简单，但又最常用的构造法。如 $x = [0.2, pi, -5]$。

(2) 运用 *MATLAB* 函数生成法

MATLAB 中有许多用来生成特殊形式数组的函数，如均匀分布随机数组的 $\text{rand}(1, n)$，全 1 数组 $\text{ones}(1, n)$ 等，更多函数参见表 2-2 。

2.3.2 二维数组的创建

二维数组是最常用的数组，如一张人脸灰度图就是一个二维数组。本节介绍二维数组的几种创建方法。

1、直接输入法

对于较小数组，从键盘上直接输入最简便。二维数组直接输入要注意以下几点：

♡ 输入数组收尾是方括号“[ ]”；

♡ 数组行间用分号“;”或回车键隔开；

♡ 数组元素之间用逗号“,”或空格隔开；

♡ 分号“;”在“[ ]”中时，表行间分隔符，在在指令末尾时，屏幕上不显示该指令执行结果。

2、数组编辑器创建法

在工作空间新建变量直接输入。

3、利用 M 文件创建法

对于经常需要调用的数组，尤其是较大而复杂的数组，建立一个专门的 M 文件，当需要时直接运行文件即可。最好在 M 文件首行编写简短说明以及命名，以便查阅。算例参见 myMatrix.m。

4、利用 MATLAB 函数创建数组

实际中，往往需要产生一些特殊形式的数组/矩阵。MATLAB 提供了许多生成特殊数组的函数。表 2-2 列出了最常用的函数。

表 2-2: 标准数组生成函数

指 令	含 义	指 令	含 义
diag	对角数组（仅对二维适用）	rand	均匀分布随机数组
eye	单位数组（仅对二维适用）	randi	均匀分布整数
magic	魔方数组（仅对二维适用）	randn	正态分布随机数组
ones	全 1 数组	random	各种分布随机数组
zeros	全 0 数组	randsrc	在指定字符集上生成均布随机数组
gallery	各种用途的测试数组/矩阵	randperm	1 ~ n 随机排列的整数

2.3.3 二维数组元素的编址和寻访

二维数组是使用最多，其编址寻访的概念和方法不难推广到高维数组，本节内容将以二维数组展开。

1、二维数组元素的编址

二维数组中元素的位置有两种表达方式：全下标编址（Subscripts）和单序号编址（SingleIndex）。

(1) *全下标编址* 类似于矩阵元素的表示，如 $X(2,1)$ 表示二维数组第 2 行第 1 列上的元素。这种编址形式最明了直接，使用最多。

(2) *单序号编址* 用单个序号唯一地确定元素在数组中的位置。MATLAB 中单序号产生规则是按列依次编号，即第一列编完接着编第二列，指到最后一列。

(3) *全下标编址和单序号编址的转换指令* MATLAB 提供了数组的全下标编址和单序号编址之间的相互转换指令如下：

[rowSub, colSub] = ind2sub(ArraySize, IND)

IND = sub2ind(ArraySize, rowSub, colSub)

说明 ArraySize 表示数组规模，如二元数组的行、列数组成的二维数组；IND 是 K 个感兴趣元素的单序号，即是一个 K 元数组。rowsub 和 colsub 是两个 K 元数组，分别表示感兴趣元素的行指标和列指标。见实例 ArrayBianzhi.m。

2、二维数组元素的寻访

如何借助全下标或单序号编址寻访数组的元素、子数组，如何从数组中寻找满足某种条件的元素是本小节讲述的内容。对二维数组元素的寻访有三种方式：

(1) *全下标寻访*

(2) *单序号寻访*

(3) *逻辑寻访*

下面用实例演示，二维数组单序列编址规则；数组元素的三种寻访方法的基本操作；比较三种寻访方式的应用差异。见 M 文件 ArrayXunfang.m。

2.3.4 数组构做技法综合

为了生成比较复杂的数组，或为了对已生成数组进行修改、扩展，MATLAB 提供了诸如反转、插入、提取、收缩、重组等操作。理解和掌握本节内容，对灵活使用 MATLAB 是重要的。最常见的操作函数将见表 2-3，ArrayCaozuo.m 给出了其中部分操作函数的实例。

表 2-3: 数组操作函数

指 令	功 用
reshape	在总元素不变的前提下，改变数组的“行数、列数”
flipud	以数组“水平中线”为对称轴，交换上下对称位置上的数组元素
fliplr	以数组“垂直中线”为对称轴，交换左右对称位置上的数组元素
flipdim	按指定维度反转，对高维数组更有用
rot90	把数组逆时针旋转 90°（不适用于二维以上）
sort	按升序或降序排列
sortrows	按升序排矩阵的行
circshift	循环移动数组
shiftdim	数组维度移动，或前导悬垂维度的删除
permute	按指定的次序，对矩阵各维的次序进行重组（适用于任何维度）
ipermute	permute 的逆操作
repmat	按指定的“行数、列数”铺放模块数组，以形成更大的数组
cat	把规模相同的若干数组，沿“指定维”方向，串接成高维数组
horzcat	水平排放数组
vertcat	垂直排放矩阵
diag	提取对角元素，或生成对角阵
blkdiag	据输入构造块对角阵
squeeze	撤销长度为 1 的“孤维”，使数组降维
end	数组最后一个下标

2.3.5 高维数组

创建高维数组最常用的方法有：

♡ 直接通过“全下标”元素赋值方式创建高维数组；

♡ 由若干同样大小的低维数组组合成高维数组；

♡ 由函数 ones，zeros，rand，randn 直接创建标准数组；

♡ 借助 cat（catenate），repmat，reshape 等函数构作高维数组。

高维数组的创建演示实例见 ArrayND.m。其他操作函数及操作技法请查阅相关专著。

2.4 数值数组运算

从外观形状和数据结构上看，二维数组和数学中的矩阵没有区别。但是，矩阵作为一种变换或映射算子的体现，矩阵运算有着明确而严格的数学规则。而数组运算是 MATLAB 软件所定义的规则，其目的是为了数据管理方便、操作简单、指令形式自然和执行计算的有效。虽然数组运算尚缺乏严谨的数学推理，但它的作用和影响还是值得我们关注的。对比两种运算指令形式和实质内涵的异同，列于表 2-4 ，服从数组运算规则的函数及其他算符列于表 2-5 。

表 2-4: 数组、矩阵运算符及其数学含义

数组运算		矩阵运算	
数学含义	指　令	数学含义	指　令
A 的非共轭转置	$A.'$	A 的共轭转置	A'
$a_{ij}+b_{ij}$	$A+B$	$A+B$	$A+B$
$a_{ij}-b_{ij}$	$A-B$	$A-B$	$A-B$
$a_{ij}\times b_{ij}$	$A.*B$	AB	$A*B$
a_{ij}/b_{ij} 或 $b_{ij}\backslash a_{ij}$	$A./B$ 或 $B.\backslash A$	AB^{-1} 或 $AB^{\dagger}$	A/B
		$B^{-1}A$ 或 $B^{\dagger}A$	$B\backslash A$
$a_{ij}\hat{}b_{ij}$	$A.\hat{}B$		
$a+b_{ij}$	$a+B$ 或 $a.+B$	$a+b_{ij}$	$a+B$
$a-b_{ij}$	$a-B$ 或 $a.-B$	$a-b_{ij}$	$a-B$
$a\times b_{ij}$	$a.*B$	aB	$a*B$
a/b_{ij} 或 $b_{ij}\backslash a$	$a./B$ 或 $B.\backslash a$		
$a\backslash b_{ij}$ 或 b_{ij}/a	$a.\backslash B$ 或 $B./a$	$\frac{1}{a}B$	B/a 或 $a\backslash B$
$a^{b_{ij}}$	$a.\hat{}B$	(B方阵时) a^B	$a\hat{}B$
b_{ij}^{a}	$B.\hat{}a$	(B方阵时) B^a	$B\hat{}a$

3 字符串、元胞和构架数组

本节将介绍另外三种数据类型：字符串数组（Character String Array）、元胞数组（Cell Array）和构架数组（Structure Array）。字符串数组在编程中是不可回避的，而且熟练运用相关函数及操作能大大提升编程效率。元胞数组和构架数组对初学者来说不易运用，但正是得益于这两类数据才使得 MATLAB 提供出功能齐全、表达简洁、操作方便的 M 码函数指令，才使得图形表示，人机交互界面顺利实现。对我们来说图像的读取存储及相关操作都会用到这三种数据类型。

表 2-5: 服从数组运算规则的函数及其他算符

函数类别		举　例
服从数组运算规则的函数	三角、反三角	sin，cos，tan，…，asin，acos，atan，…
	双曲、反双曲	sinh，cosh，…，asinh，acosh，…
	指数、对数	exp，sqrt，pow2，log，log10，log2
	圆整、求余	ceil，floor，fix，round，mod，rem，idivide
	模、角、虚实部	abs，angle，real，image，conj
	符号函数	sign
关系运算符		==，~=，>，<，>=，<=
逻辑运算符		&，|，~

3.1 MATLAB 的数据类型

MATLAB 涉及到的所有数据类型归纳到图 3-2 中。

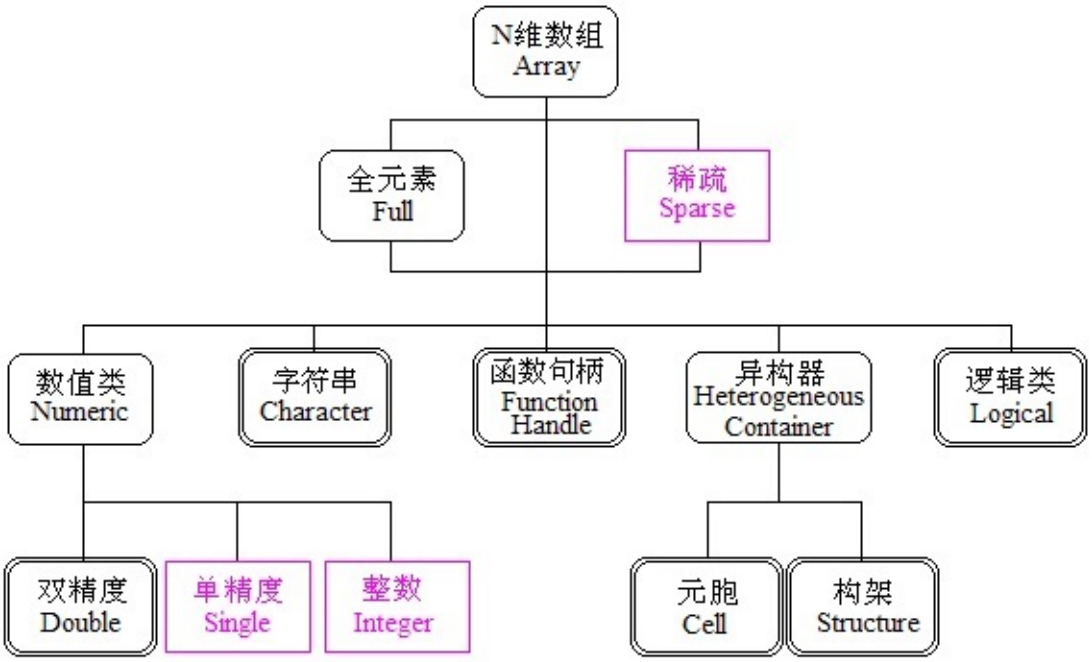


图 3-2: MATLAB 的数据类型

3.2 字符串数组

本小节内容集中于字符串数组（以下简称串数组）。与数值数组相比，串数组在 MATLAB 中的重要性较小，但它不可缺少。假如没有串数组及相应的操作，那么数据可视化

将会遇到困难，构作 MATLAB 的宏指令也将会遇到困难。

字符变量的创建方式是：在指令窗中，先把待建的字符放在“单引号对”中，再按 [Enter] 键。注意，这“单引号对”必须在英文状态下输入。这“单引号对”是 MATLAB 识别送来内容“身份”（是变量名、数字，还是字符串）所必需的。串函数和串操作，为 MATLAB 的文字表达、复杂字符的组织、宏功能的发挥提供了有力的支持。

3.2.1 数组的属性和标识

本小节以算例（见 char_1.m）的形式演示串数组的基本属性、标识和简单操作。主要包括：

(1) 创建串数组

(2) 串数组的大小

(3) 串数组的元素标识　在一维串数组中，自左至右依次编号，采用单下标索引。

(4) 创建带单引号的字符串　每个单引号符用“连续的两个单引号符”表示。

(5) 由小串构成长串

3.2.2 复杂串数组的创建

1、多行串数组的创建

(1) 补空等长直接输入法　在直接创建多行数组是，关键是要保证同一串数组的各行字符数要相等，即保证各行等长。为此，多数情况下须通过空格来调节各行长度。见算例 char_2.m。

(2) 利用串操作函数创建多行串数组　用专门函数 char，str2mat，strvcat 创建多行串数组。见例 char_2.m。

一个小小的应用，在给 MATLAB 图标添加双行标题时，非常方便有效，参见实例 chart_usecellarray.m。

2、利用元胞数组创建复杂字符串

(1) 直接输入法生成存放复杂字符串的元胞数组　见算例 char_3.m

(2) 借助 *cellstr* 指令生成存放复杂字符串的元胞数组　见算例 char_3.m

3.2.3 串操作函数

MATLAB 中常用的字符串操作函数及其功能介绍列在表 3-6 中。部分操作的算例演示参见 char_4.m　需补充算例。

表 3-6: 字符串操作函数

指 令	含 义	指 令	含 义
blanks(n)	创建 n 个空格	char(s1,s2)	把串 s1，s2 等逐个写成行，形成多行数组
deblank(s)	删去串尾部的空格符	str2mat(s1,s2,...)	把串 s1，s2 等逐个写成行，形成多行数组
eval(s)	把串 s 当作 MATLAB 指令运行	strcat(s1,s2,...)	把字符串 s1，s2 连接成长串
eval(s1,sc)	把串 s1 当作 MATLAB 指令运行；若 s1 运行发生错误，则运行 sc	strvcat(s1,s2,...)	把串 s1，s2 等逐个写成行，形成多行数组，并删除全空行
sort	按字符值的升序或降序排列元素	strjust(s)	字符串的对齐方式：或右对齐、或左对齐、或对中
feval(f,x,y,...)	对输入量 x，y 等计算函数 f	strmatch(s1,s2)	逐行搜索串 s2，给出以 s1 开头的那些行的行号
strfind(s1,s2)	在较长串中，找出短串的起始字符的下标	strncmp(s1,s2,n)	若串 s1，s2 的前 n 个字符想，则判断“真”给出逻辑 1
ischar(s)	s 是字符串，判断“真”给出逻辑 1	strrep(s1,s2,s3)	串 s1 中所以出现 s2 的位置替换为 s3
isletter(s）	以逻辑 1 指示 s 里字符的位置	strtok(s)	找出第一个间隔符（空格、制表位、回车符）前的内容
isspace(s)	以逻辑 1 指示 s 里空格的位置	strcmp(s1,s2)	若字符串 s1，s2 相同，判断“真”给出逻辑 1
lower(s)	使 s 里的英文字符全部小写	upper(s)	使 s 里的英文字符全部大写

3.3 元胞数组

元胞数组（cell array）是 MATLAB 的一种特殊数据类型，可以将元胞数组看做一种无所不包的通用矩阵，或者叫做广义矩阵。组成元胞数组的元素可以是任何一种数据类型的常数或者常量，每一个元素也可以具有不同的尺寸和内存占用空间，每一个元素的内容也可以完全不同，所以元胞数组的元素叫做元胞（cell）。形象地说，元胞数组就像一个仓库，里边有若干个房间（元胞），每个房间里可以存放任意类型的物品（元胞内容）。

3.3.1 元胞数组的创建和显示

1、元胞标识寻访和内容编址寻访的不同

在元胞数组中，元胞和元胞里的内容是两个不同范畴的东西。因此，寻访元胞和寻访元胞内容是两种不同的操作。对元胞数组中元胞元素的寻访，也就是“外标识的元胞元素”用“圆括号”，如 A(2,3) 是指 A 元胞数组中第 2 行第 3 列元胞元素；对元胞元素的内容寻访，也就是“编址元胞元素内容”用“花括号”，如 A{2,3} 是指 A 元胞数组中第 2 行第 3 列元胞元素中存储的内容。

2、元胞数组的创建和显示

(1) 直接创建法之一：“外标识元胞元素赋值法”；

(2) 直接创建法之二：“编址元胞元素内容的直接赋值法”；

(3) 显示元胞数组全部或部分内容的指令是 celldisp，若直接在指令窗输入元胞数组的名，除单元素元胞外，一般只能得知元胞所存内容的属性，而不显示元胞数组内容。

本小节的算例见 cell1.m。

3.3.2 元胞数组的扩充、收缩和重组

元胞数组的扩充、收缩和重组的方法大致与数值数组情况相同。本小节以算例形式分别表述，参见 cell2.m。

3.3.3 元胞数组内容的获取和配置

本节给出调取元胞数组内容的详细方法（参见 cell3.m），包括：

(1) 调取一个元胞

(2) 调取一个元胞的内容

(3) 调取元胞内的子数组

(4) 同时调取多个元胞的内容

3.4 构架数组

与元胞数组一样，构架数组（structure array）也能在一个数组里存放各类数据。从一定意义上讲，构建数组组织数据能力比元胞数组更强、更富于变化。

构架数组的基本组分（Element）是构架（structure）。数组中的每个构架师平等的，它们以下标区分。构架必须在划分“域”后才能使用。数据不能直接存放于构架，而只能存放在域中。构架的域可以存放任何类型、任何大小的数组（如任意维数值数组，字符串数组，元胞数组，符号对象等）。而且不同构架的同名域中存放的内容可以不同。

构架数组维数不受限制，可以使一维、二维或更高维，不过一维构架数组用得最多。为便于比较，表 3-7 给出了构架数组与元胞数组的异同。

表 3-7: 构架数组与元胞数组的异同比较

	元胞数组（Cell Array）	构架数组（Structure Array）
举　　例	(3×4) 元胞数组 A	(3×4) 构架数组 B， 它有名为 f1，f2 的两个域
基本组分（Element）	元胞（Cell）	构架（Structure）
可存放的数据类型	任何类型（数值、字符、元胞、构架等及其他对象）	任何类型（数值、字符、元胞、构架等及其他对象）
直接存放数据的场所	元胞本身， 如：A(1,2)	域（Field）， 如构架域 B(1,2).f1
基本组分的寻访方式	被标识的元胞名，如：A(1,2)	被标识的构架名，如：B(1,2)
具体内容的寻访方式	“花括号”标识的元胞名， 如：A{1,2}	带“域名”的标识构架名， 如：B(1,2).f1
实现元胞数组与构架数组之间转换的指令	cell2strct 把元胞数组转换为构架数组	struct2cell 把构架数组转换为元胞数组

3.4.1　构架数组的创建和显示

1、直接创建法及显示

构架数组的结构形式与常见的一般数组不同，为便于理解，本小节通过实际算例/归纳形式展开。主要介绍：① 单构架的创建和显示；② 构架数组的创建和显示。具体算例参见 struct1.m。

说明：

(1) 在指令窗直接键入单构架名，通常指能得到该构架的结构信息，而不显示该构架域中的具体内容；除非内容是极为简单的数值变量或单行字符串。

(2) 在指令窗键入不带子域的构架域名时，可显示该域中的内容；但当带有子域时，需键入完整的构架名、域名、子域名，才能显示域中的内容。

(3) 构架数组中每个构架元素的域名是相同的，但子域名不必相同。

2、利用构造函数创建构架数组

除了上一小节讲的直接复制法外，MATLAB 还提供了一个专门的构造函数 struct 可以创建构架数组。这里通过算例介绍其使用方法，参见 struct2.m。

说明：

(1) struct 函数指令中偶序号输入宗量必须是元胞数组，用来向不同域赋值，除允许时（1×1）元胞数组（集单个元胞）外，必须有完全相同的维数。

(2) 在任何情况下，空数组“[ ]”总可以用来创建新的空域。

(3) struct 不能直接创建带子域的构架数组，子域需另行创建。

3.4.2 构架数组域中内容的调取和设置

构架数组的域是存放数据的场所，因此调取和设置构架数组中的数据的前提是事先知道域名。上节已经知道，直接在命令窗键入构架名就会显示其域名，但这不便于进一步的操作，MATLAB 里有专门的指令 fieldnames 解决此问题。下面简列出几个常用的构架数组操作指令，为简单起见，约定 Sn 表示构架名 structname，Fn 表示构架的所有域名，fn 表示构架的某一个域名，Fc 表示构架域中的内容。

Fn = fieldnames(Sn)	获得构架域名，输出一维元胞数组
Fc = getfield(Sn,{S-index},‘fn’,{f-index})	获得具体构架域中的内容
Sn = setfield(Sn,{S-index},‘fn’,{f-index},value)	设置具体构架域中的内容
Sn = rmfield(Sn,‘fn’)	删除构架的域，删除域影响整个构架数组，删除子域只影响当前操作的那个子构架

上述命令中，S-index 用来指定“元构架”的下标，f-index 用来指定域中数组的下标，两者均须是元胞数组形式。更多构架数组的其他操作，如构架数组的扩缩、域的增删、域名的重排、构架数组和元胞数组之间的转换等，请参阅相关的 MATLAB 教程获取。

4 MATLAB 控制流

计算机编程语言允许程序员根据某些判断结构来控制程序流的执行次序。MATLAB 提供了 5 种控制程序流的结构：for 循环结构，while 循环结构，if-else-end 条件分支结构，switch-case 切换分支结构，try-catch 容错结构。本节对于各组关键词的用法描述比较简明，且大多通过算例进行。

4.1 for 循环和 while 循环结构

♠ for 循环结构的语法形式为：

```
for i = 1:n
    (commands)
end
```

说明：1、i 为循环变量；2、commands 为循环体，每次循环被执行；3、for 循环的次数是确定的。

♠ while 循环结构的语法形式为：

```
while expression
    (commands)
end
```

说明：1、当控制表达式 expression 为“真”，则执行循环体 commands；否则，结束循环；2、while 循环的次数是不确定的。

4.2 if-else-end 条件分支结构

if-else-end 指令为程序流提供了一种分支控制，常见的有单分支、双分支和多分支。

♠ 单分支结构语法形式为：	♠ 双分支结构语法形式为：	♠ 多分支结构语法形式为：
if expression (commands) end	if expression (commands1) elseif (commands2) end	if expression1 (commands) elseif expression2 (commands) elseif (commandsend) end
说明：当 expression 给出“逻辑 1”时，(commands) 指令才被执行。	说明：当 expression 给出“逻辑 1”时，(commands1) 指令被执行；否则，(commands2) 指令被执行。	说明：expression1，expression2，⋯ 依次被判断，若有一个给出“逻辑 1”，则执行紧跟的(commands)；否则，(commandsend) 指令被执行。if 多分支结构常用 switch-case 代替。

算例 M 文件 creat_folder.m 和 creat_folder_new.m 是最近项目组任务（查找图片）中用于批量创建文件夹，其中使用了 for 循环、if 结构。第二个为函数版本，使用了子函数，也对字符串的条件有所放松，仅供参考。

4.3 switch-case 切换分支结构

♠ switch-case 指令的语法结构形式如下：

```
switch ex                 ex 为一标量或字符串
    case test1
        (commands 1)      当 ex 等于 test1 是，执行组命令 1，然后跳出该结构。
    case test2
        (commands 2)
    ......
    case testk
        (commands k)      当 ex 等于 testk 是，执行组命令 k，然后跳出该结构。
    otherwise             otherwise 指令可以不存在
        (commands)        表达式不等于前面所有检测值时，执行该组命令。
end
```

switch-case 切换分支结构的实例见 switch_case.m，该例中涉及到元胞数组、构架数组的创建，字符串的转换，打印输出等。

4.4 try-catch 容错结构

♠ try-catch 结构具有容错能力，若执行“try 块程序”发生错误，就跳转去执行“catch 块程序”，以作一种后补应对措施；当然，若“try 块程序”执行正常，那么“catch 块程序”是不被执行的；若“try 块程序”也发生错误，则该程序终止。try-catch 容错控制的语法结构如下：

```
try
    (commands1)        组命令 1 总被执行。若正确，则跳出此结构。
catch
    (commands2)        仅当组命令 1 出现执行错误，组命令 2 才被执行。
end
```

4.5 编程用的其他指令

1、辅助控制指令 continue 和 break

Continue 和 break 指令为用户编写循环控制提供了更大的自由度。它们的具体含义如下：

continue	在 for 或 while 循环中遇到该指令，执行下一次迭代，不管其后指令如何。
break	在 for 或 while 循环中遇到该指令，跳出循环，不管其后指令如何。

2、其他常用指令

控制程序流的其他常用指令及其使用说明列于表 4-8 中。

5 M 函数文件

除极简单的问题外，大多数实际问题不可能仅仅依靠 MATLAB 指令窗中逐条输入解决，而需要编程。本节就讲述如何编写解决实际问题的程序文件。

5.1 M 脚本文件和 M 函数文件

M 文件根据调用方式的不同可分为两类：

♡ Script: 脚本文件/命令文件　直接输入文件名即可运行

♡ Function: 函数文件　供其它 M 文件调用，通常带输入参数和输出参数

M 脚本文件与 M 函数文件的区别见表 5-9 ：

表 4-8: 控制程序流的其他常用指令

指令及使用格式	功 能 说 明
v=input('message')	用于键入数值、字符串、元胞数组等数据，待输入结束，按下 Enter 键，控制权交换 MATLAB。
v=input('message','s')	以字符串形式赋给变量 v，待输入结束，按下 Enter 键，控制权交换 MATLAB。
keyboard	将控制权交给键盘，使用户可以从键盘输入各种指令。仅当用户输入 指令后，控制权才交还给程序。它与 input 的区别是：它允许输入任意多个 MATLAB 指令，而 input 只能输入“赋给变量的值”。
return	把控制权由 return 所在的被调函数交还主调函数或键盘。
pause	使程序暂停执行，等待用户按任意键继续。
pause(n)	使程序暂停 n 秒后，在继续执行。

表 5-9: M 脚本文件与 M 函数文件的区别

M 脚本文件	M 函数文件
♡ 需要执行指令的集合	♡ function 引导的指令集合
♡ 没有输入、输出宗量	♡ 一般有输入、输出宗量，且没有数目限制；允许使用比标称数目较少的输入输出宗量，实现对函数的调用。
♡ 运行产生的变量保存于工作空间，直到使用 clear 指令清除，或关闭 MATLAB 指令窗。	♡ 运行时开辟临时（函数）工作空间，到函数运行结束当即被清除。若运行函数文件时调用了脚本文件，则此脚本文件产生的变量都存于当前函数空间。

5.2 函数文件的一般结构

M 函数文件犹如一个“黑箱”，通过输入、输出变量实现与外界的联系，而内部的运作是苍耳不见的。它的一般结构如下：

```
function   [output] = myfun(input)      函数申明行，包括函数名，输入、输出宗量列表。
% 注释说明部分                          帮助文档，可选，但最好有，以便在调用时查阅。
(commands)                              函数体语句，必须有。
end                                     只有主函数时可以省略，若有子函数则必须有。
```

说明：

(1) 函数名的命名规则与变量名相同（必须以字母开头，可包括数字，下连符），字符数不

得超过 31 个；

(2) 函数文件名必须与函数名一致；

(3) 当输出参数多于一个时，用方括号括起来；也可以没有输出参数；

(4) 注释常包括：函数功能介绍，变量说明，调用格式，编写和修改记录（参考文献，作者，日期信息）等；

(5) 函数必须是一个单独的 M 文件。

5.3 主函数和子函数

在 MATLAB 中，函数 Function 又被细分为主函数、子函数、嵌套函数、私用函数、匿名函数等。本节只对主函数和子函数进行阐述。

5.3.1 主函数

主函数（primary function）的特点：

◊ 一般为“与保存文件同”的那个函数；

◊ 在当前目录、搜索路径上，列出文件名的函数；

◊ 在指令窗中或其他函数中，可以直接调用的函数；

◊ M 函数文件中，由第一个 function 引出的函数；

◊ 采用 help functionname 可获取函数所携带的帮助信息。

5.3.2 子函数

子函数（subfunction）的特点：

◊ 子函数不独立存在，只能寄生在主函数体内；

◊ 在函数文件中，由非第一个 function 引出的函数；

◊ 一个 M 函数文件可以包含多个子函数；

◊ 子函数只能被其所在的主函数和其他“同居”子函数调用；

◊ 子函数可以出现在主函数体的任何位置，其位置先后不影响调用次序；

◊ 在 M 函数文件中，任何指令通过“名字”对函数进行调用时，子函数的优先次序仅次于内装函数；

◊ 同一文件的主函数、子函数的工作空间都是彼此独立的，各函数间的信息，或通过输入输出宗量传递，或通过全局变量传递；

◊ 采用 help functionname/subfunctionname 可获取子函数所带的帮助信息。

主函数和子函数的实例见 creat_folder_new.m 和 p-s-function.m （补充实例）。另外，任意一个函数都可以被自己调用，参见算例 FilesAll.m。

5.4 函数的变量

5.4.1 局部变量与全局变量

局部（*Local*）变量 存在于函数空间内部的中间变量，产生于该函数的运行过程中，不会被其他函数文件或脚本文件使用。当函数调用完毕后，局部变量全部被清除。

全局（*Global*）变量 是允许几个不同的函数空间以及基本工作空间共享的同一个变量。每个希望共享全局变量的函数或 MATLAB 基本工作空间，必须逐个用 global 对具体变量加以专门定义。未申明的函数无权享用全局变量。全局变量的值在一处发生变化，则在其他函数空间以及基本工作空间中的同名变量也就随之变化。全局变量损害了函数的封装性，故不提倡使用全局变量。

5.4.2 变量的检测

本小节首先列出几个关于输入输出宗量的指令，见表 5-10 :

表 5-10: 函数变量的几个指令

指　令	功　能
nargin	在函数体内，用于获取实际输入宗量；
nargout	在函数体内，用于获取实际输出宗量；
nargin('fun')	获取'fun'指定函数的标称输入宗量数；
nargout('fun')	获取'fun'指定函数的标称输入宗量数；
inputname(n)	在函数体内使用，给出第 n 个输入宗量的实际调用变量名；
varargin	"变长度"输入宗量列表
varargout	"变长度"输入宗量列表

函数文件 EPC2A\Epcimage.m 是实现一个人脸识别算法是用到的，其中涉及到输入宗量的检测，EPC2A\pca.m 是经常使用的一个函数文件，将看到另一种输入宗量的检测，仅供参考。

6 图像相关操作

数字化图像按照记录方式可分为两种：矢量图像和位图图像。矢量图像用数字的矢量方式来记录图像等内容，以线条和色彩块为主。位图图像就是将图像的每一个像素点转换为一个数据。如果以 8 位来记录，便可以表现出 256 种颜色或色调（$2^8 = 256$），因此使用的位元素越多所能表现的色彩也就越丰富。一般所说的真彩色是指 24 位（$2^8 \times 2^8 \times 2^8$）的位图存储模式，适合于内容复杂的图像和真实照片。用数码相机和扫描仪获取的图像都属于位图。

在 MATLAB 中，一幅图像可能包含一个矩阵，也可能包含一个颜色映像表矩阵。MATLAB 中有 3 种基本的图像类型：索引图像（见表 6-11 ）、灰度（强度）图像和 RGB （真彩）图像。三种类型的图像的比较列于表 6-12 。

表 6-11: 索引图像存放的不同数据类型和显示指令

数据类型	双精度型（Double）	无符号整型（Uint8/Uint16）
图像矩阵	X 数组大小：$m \times n$ 元素取值：[1, p] 间的整数	X 数组大小：$m \times n$ 元素取值：[0, p-1] 间的整数
色图矩阵	map 数组大小：$p \times 3$ 元素取值范围：[0,1] 间的浮点数	map 数组大小：$p \times 3$ 元素取值范围： [0, 255]或[0, 65535] 间的浮点数
成图方式	像素 P_{ij} 的颜色为 map(X(i, j), :)	同左
显示代码	imageX; colormap(map); axis image off 在特殊场合，可以使用伴随色图 map 以外任何其他色图 mapk image(X); colormap(mapk); axis image off	

表 6-12: 三种图像类型的比较

类型	索引图像	灰度图像	RGB 图像
描述	一种把像素值直接作为 RGB 调色板下标的图像	只有强度信息，而没有颜色信息的图像	用 R、G、B 分别表示红、绿、蓝 3 种不同的颜色，可以合成出任意颜色
图像数据	$m \times n$ 矩阵 X， 颜色映射矩阵 map	$m \times n$ 矩阵	$m \times n \times 3$ 张量
数据类型	uint8/uint16/double, double([0,1])	double([0,1]), uint8([0,255])	uint8/uint16, double([0,1])
显示代码	[X,map]=imread(‘canoe.tif’); image(X); colormap(map)	I=imread(‘moon.tif’); imagesc(I,[0,1]); colormap(gray)	RGB=imread(‘flowers.tif’); image(RGB)

二值图像 可以看成是一个仅包括黑与白的特殊灰度图像，或是仅有两种颜色的索引图像。它的二维 $m \times n$ 矩阵仅由 0 和 1 构成，可以保存为 double 型或 uint8 型的数组，显然使用 uint8 型更节省空间。显示一幅二值图像的代码为：

```
BW = imread(‘circles.png’);
```

imshow(BW)

上述几种类型的图像显示效果见 imagetype.m 代码还有问题。

6.1 图像读、写及显示

6.1.1 基本图像操作函数

本小节介绍几个常用的操作函数，实例演示见 imagedisplay.m。

♡ **imread 函数** 用于读取一幅图像，其调用格式为

I = imread(filename, fmt)

[X, map] = imread(filename,fmt)

说明：imread 适用于任何图像类型。① filename 是字符串，包含图像文件路径和文件名，图像在当前路径时，只保留文件名即可；② fmt 指定了文件格式，如：jpg，bmp，tif，png 等；③ I 是返回的图像数据，是二维矩阵或三维数组；④ map 用于存储与 I 相关的颜色映射表。

♡ **imwrite 函数** 用于实现图像文件的写入操作，其调用格式为

imwrite(I, filename, fmt)

imwrite(X, map, filename, fmt)

说明：① filename 是字符串，包含图像要存储到的路径和文件名，若要存储到当前路径，则只要文件名即可；② 有时 filename 和 fmt 可合二为一，MATLAB 会根据扩展名自动推断文格式。

♡ **imshow 函数** 用于显示一幅图像，其调用格式为

imshow(I) ⟨1⟩

imshow(I,[low high]) ⟨2⟩

inshow(X, map) ⟨3⟩

imshow(filename) ⟨4⟩

说明：① ；指令 ⟨1⟩ 中的 I 可以是灰度图像、RGB 图像或者二值图像；② 指令 ⟨2⟩ 中的 I 是灰度图像，[low high] 指定了图像 I 的数据范围，低于 low 的像素显示为黑色，高于 high 的像素显示为白色，之间的按原值显示。③ 指令 ⟨4⟩ 用于直接显示 filename 指定的图像文件；④ 对于真彩图像也可以用 image 来显示。

♡ **imfinfo 函数** 用于显示或获取图像文件的特征数据，其调用格式为

imfinfo(filename,fmt)

info = imfinfo(filename,fmt)

说明：① filename 和 fmt 的用法与前几个指令中的相同；② 返回的 info 是一个构架数组，它的域就是图像的各种属性，如：Filename，FileModDate，FileSize，BitDepth 等等。

6.1.2 帧图像显示

多帧图像是一个包含多个图像的图像文件。MATLAB 支持 HDF 和 TIFF 两种类型的多帧图像文件格式。文件一旦被读入 MATLAB 平台中，帧数即由矩阵（数组）的第四维决定。多帧图像可以使用多帧显示方式进行，其中包括：

- ♡ 使用 **imshow** 函数单独显示每一个图像帧；
- ♡ 使用 **montage** 函数同时显示所有图像帧；
- ♡ 使用 **immovie** 函数将图像帧转换为电影。

帧图像显示实例演示参加 multipleimage.m (还有问题需调试)。

6.2 图像格式转换

6.3 图像的标注

7 其他常用指令

7.1 几类常用指令

1、新建、删除文件夹

- ♡ mkdir('folderName') 新建文件夹。
- ♡ mkdir('parentFolder', 'folderName') 在父目录下新建文件夹，若父目录不存在，则一同建立。
- ♡ rmdir(folderName) 删除空文件夹，当前路径下或完全路径。
- ♡ rmdir(folderName,'s') 删除文件夹及其内容。

2、复制、移动文件或文件夹

- ♡ copyfile('source', 'destination') 复制文件或文件夹，同时复制多个文件可使用通配符“∗”。也可复制并重命名，如 copyfile('myfile.txt','myfile2.txt')。
- ♡ copyfile('source', 'destination', 'f') 当目的地只读是仍可复制但信息为空。
- ♡ movefile('source') 移动文件或文件夹到当前路径，移动多个文件可用通配符“∗”。
- ♡ movefile('source', 'destination') 移动文件或文件夹到目标路径。

3、获取文件夹内容

- ♡ dir 列出 MATLAB 当前路径的文件和文件夹，结果显示在命令窗。
- ♡ dir name 在命令窗列出 name 指定的文件或文件夹，当 name 是一个路径是显示其下内容。可用通配符。
- ♡ listing = dir(name) 返回构架数组，其域名包括：name，date，bytes，isdir 和 datenum。常用来获取文件夹下的图片信息。
- ♡ ls 列出 MATLAB 当前路径的文件和文件夹，结果显示在命令窗。与 dir 相同。
- ♡ ls name 同 dir name。
- ♡ list = ls(name) 返回字符串数组，匹配路径下的文件名和文件夹名，即只显示了 dir 返回值 listing 的域 name 的内容。

4、重命名文件、文件夹

- ♡ !ren E:\ 1.txt 2.txt 重命名文件或文件夹
- ♡ eval('!ren E:\ 1.txt 2.txt') 当需要批量处理时使用，其中文件名采用字符串变量以满足批量自动处理。

5、改变当前路径

- ♡ cd(newFolder) 改变当前路径为 newFolder。
- ♡ oldFolder = cd(newFolder) 返回当前路径作为字符串赋给 oldFolder，然后改变当前路径到 newFolder。
- ♡ cd 显示当前路径。

6、其他

7.2 常见问题