北京郵電大學

第5讲 M脚本与M函数

北京郵電大學

1 使用M文件编程

1.1 M文件的结构

1．脚本M文件

脚本文件也叫命令文件，是独立执行的文件，它**不接受输入参数，不返回任何值**，而是代码的结合，该方法允许用户将一系列MATLAB命令输入到一个简单的脚本“.m”文件中，只要在MATLAB命令窗口中执行该文件，则会依次执行该文件中的命令。

用户只需在MATLAB 的提示符“>>”下键入该M文件的文件名，这样MATLAB就会自动执行该M文件中的各条语句，并将结果直接返回到MATLAB 的工作空间。

2．函数M文件

一个函数M文件与脚本文件类似，它们都是一个扩展名为“.m”的文本文件。与M脚本不同，函数M文件不是**独立执行的文件，它接受输入参数、提供输出参数，只能被程序调用**。一个 函数M文件通常包含以下部分：

- 函数定义语句；
- H1帮助行；
- 帮助文本；
- 函数体或者脚本文件语句；
- 注释语句。

M文本编辑器

北京郵電大學

一个完整的函数M文件的结构如下：

```
function f = fact(n)                          函数定义语句
% Compute a factorial value.                 H1行
% FACT(N) returns the factorial of N,        帮助文本
% usually denoted by N!
% Put simply, FACT(N) is PROD(1:N).          注释语句
    f = prod(1:n);                            函数体
```

正文的第一行是帮助行，**称为H1行，H1行是紧随函数定义语句后面的一行注释语句，是由lookfor命令所搜索的行**，lookfor()函数只检索和显示H1行。

H1行后面连续的注释行是帮助文本，**当在命令窗口中通过help命令查询该函数的说明信息时，则在窗口中显示这些内容**。

3．块注释

在 MATLAB 5以前的版本中，注释是逐行进行的，采用百分号(%)进行标记。逐行注释不利于用户增加和修改注释内容。在MATLAB 5及以后的版本中，用户可以使用**“%{”和“%}”符号进行块注释**，“%{”和“%}”分别代表注释块的起始和结束。

北京郵電大學

4．代码元胞

一个代码元胞指用户在M文件中指定的一段代码，以一个代码元胞符号：两个百分号加空格，即**“%%”**为开始标志，到另一个代码元胞符号结束。


```
%% Example 1
% 求解符号向量的范德蒙矩阵
syms a b c d e;
v = [a,b,c,d,e];
n = length(v)
v = v(:)
A = sym(ones(n))
for j = n-1:-1:1
    A(:,j) = v.*A(:,j+1)
end

%% Example 2
```

5. 代码折叠

可在“预设值”中“代码”折叠部分勾选可折叠项。

1.2 M文件的命名规则

M文件的命名规则如下：

(1) 文件名命名要用**英文字符，第一个字符必须是字母而不能是数字，其中间不能有非法字符**。

(2) 文件名不要取为MATLAB的固有函数，尽量不要是简单的英文单词，最好是由**大小写英文、数字、下划线**等组合而成的。原因是简单的单词命名容易与内部函数名同名，结果会出现一些莫名其妙的错误。

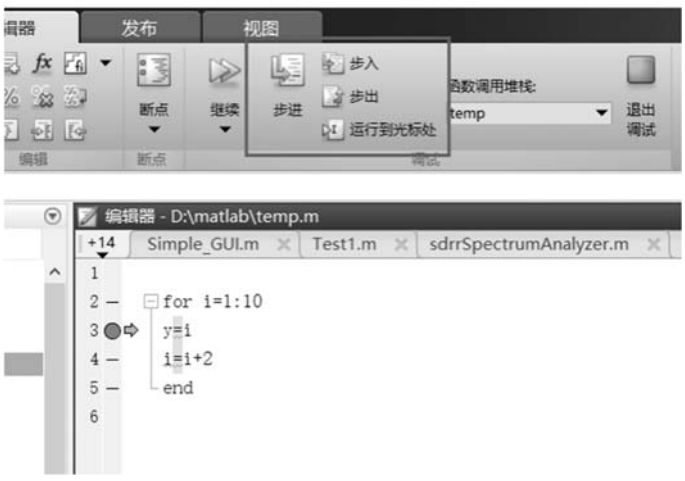
(3) 文件**存储路径一定要为英文**。

(4) 文件名不能为两个单词，如random walk，应该写成random_walk，即中间不能有空格等非法字符。

北京郵電大學

1.3 程序的调试

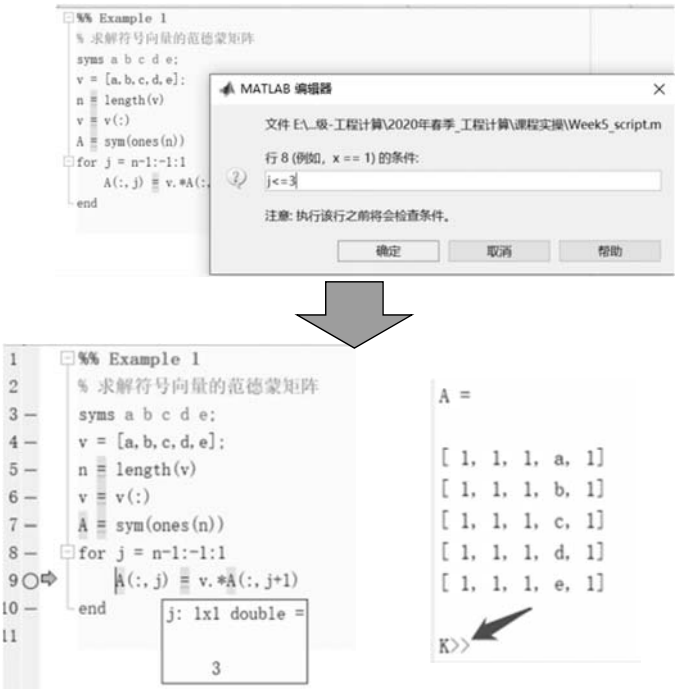
程序的错误分语法错误和逻辑错误。语法错误会直接在命令行以**“红色波浪线”**给出错误信息。逻辑错误因为从语法上是正确的，因此MATLAB不会有错误提示，需要逐条调试程序，修正设计逻辑。



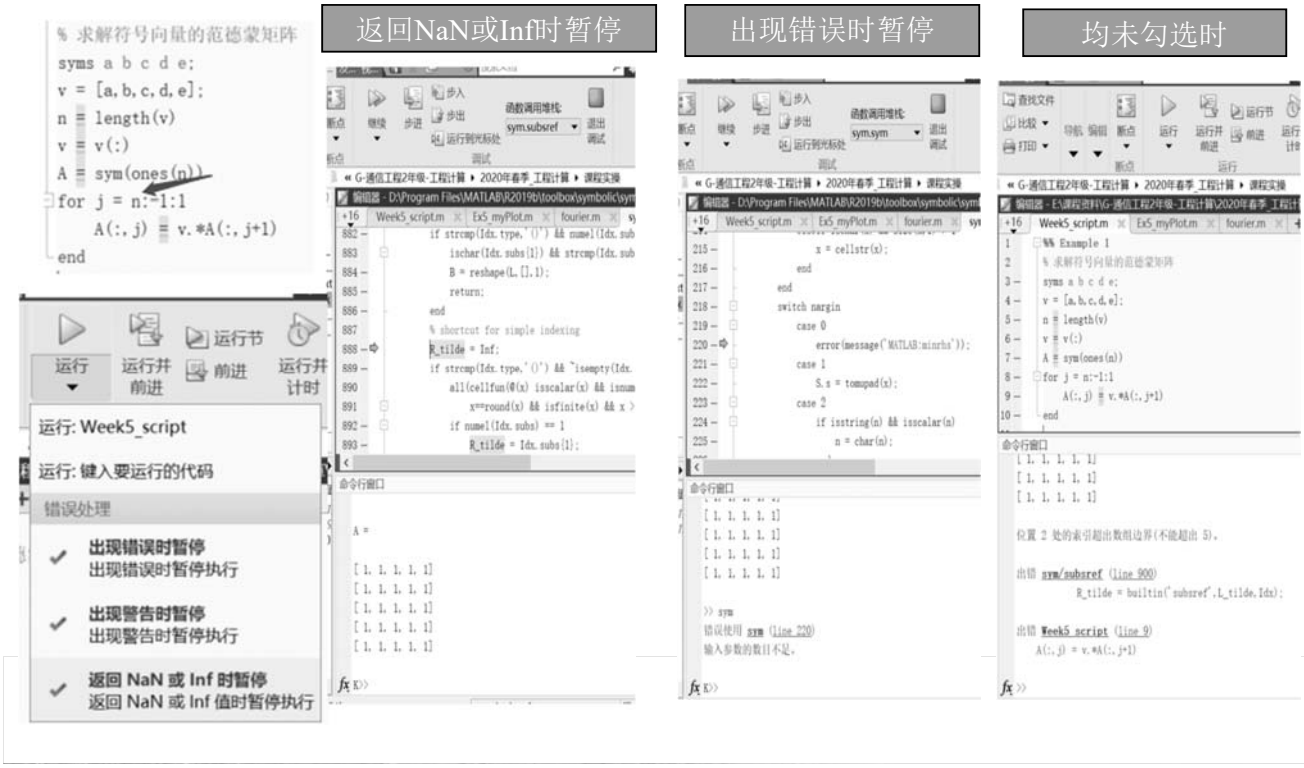
调 试 菜 单 项

菜 单 项	功　　能	快捷键
Open M-files when Debbuging	选择该选项则在调试时打开 M 文件	无
Step	单步执行，下一步	F10
Step In	进入被调用函数内部	F11
Step Out	跳出当前函数	Shift+F11
Continue	继续执行，直至下一断点	F5
Go until Cursor	执行至当前光标处	无
Set/Clear Breakpoint	设置或删除断点	F12
Set/Modify Conditional Breakpoint…	设置或修改条件断点	无
Enable/Disable Breakpoint	开启或关闭光标行的断点	无
Clear Breakpoints in All Files	删除所有文件中的断点	无
Stop if Errors/Warings	遇到错误或者警告时停止	无

条件断点设置与调试示例



运行--出现错误时暂停、出现告警时暂停、返回NaN或Inf时暂停



在程序调试中，变量的值是查找错误的重要线索，在MATLAB中有三种查看变量值的方法：

(1) 在编辑器中将鼠标放置在待查看的变量处停留，则在此处显示该变量的值；

(2) 在工作区浏览器中查看该变量的值；

(3) 在命令窗口中输入该变量的变量名，则显示该变量的值。

2 变量空间

2.1 M函数

函数M文件与脚本文件M文件在通信方面是不同的。函数是一个黑箱，使用函数时，可以看见的就是输入数据和输出数据。函数与MATLAB工作空间之间的通信，只通过传递给它的变量和通过它所创建的输出变量。在函数内局部变量与MATLAB工作空间不交互。

```
function [ a, b ] = exch(a,b)
%EXCH exchange the value of two input parameters
%   Detailed explanation goes here
temp=a;
a=b;
b=temp;
end
```

北京郵電大學

2.2 M脚本

M脚本共用MATLAB的工作空间。

```
a=input('请输入第一个数值：');
b=input('请输入第二个数值：');
fprintf('您输入的数值为[ %f %f] \n',a,b);
temp=a;
a=b;
b=temp;
fprintf('两个数值交换顺序后为 [%f %f]\n',a,b);
```

2.3 函数变量

MATLAB的变量有输入变量、输出变量和函数内使用的变量之分，还可分为**局部变量、全局变量、永久变量**。

输入变量相当于函数的入口数据，也是一个函数操作的主要对象，从某种意义上来说，函数的功能就是对输入变量进行一定的操作，从而实现一定的功能。函数的输入变量为局部变量，函数对输入变量的一切操作和修改如果不依靠输出变量的话，将不会影响工作区间中该变量的值。

北京郵電大學

1．局部变量

在M函数文件中，所有变量默认为局部变量。因此，在一个函数文件中的变量与 MATLAB工作区中的同名变量是完全不同的变量，它们存在内存的不同位置。

每个函数都有自己的局部变量，**这些变量存储在该函数独立的工作区中**，与其他函数的变量及主工作区中的变量分开存储。当**函数调用结束时，这些变量随之删除**，不保存在内存中。并且，除了函数返回值，该函数不改变工作区中其他变量的值。

脚本文件没有独立的工作区，当通过命令窗口调用脚本文件时，脚本文件分享主工作区，当函数调用脚本文件时，**脚本文件分享主调函数的工作区**。需要注意的是，如果脚本中改变了工作区中变量的值，则在脚本文件调用结束后，该变量的值发生改变。

局部变量是在函数内部使用的变量，其影响范围只能在本函数内，每个函数在运行时，都占有**独立的函数工作空间**，此工作空间和MATLAB的工作空间是相互独立的，局部变量仅存在于函数的工作空间内。当函数执行完毕之后，该变量即自行消失。

北京郵電大學

2．全局变量

在MATLAB中，函数内部定义的变量都是局部变量，它们不被加载到工作区间中。有时，用户需要使用全局变量，这时要使用**global()**函数来进行定义。

局部变量只在一个工作区内有效，无论是函数工作区还是MATLAB主工作区。与局部变量不同，**全局变量可以在定义该变量的全部工作区中有效**。当在一个工作区内改变该变量的值时，该变量在其他工作区中的变量同时改变。

任何函数要使用全局变量，必须首先声明，即使是在命令窗口也不例外。如果一个M文件中包含的子函数需要访问全局变量，则需在子函数中声明该变量，如果需要在命令行中访问该变量，则需在命令行中声明该变量。声明格式：

global 变量名1 变量名2

3．永久变量

除局部变量和全局变量外，MATLAB 中还有一种变量类型为永久变量。除了通过全局变量共享数据外，函数式M文件还可以通过**persistent**声明一个变量，来对函数中重复使用和递归调用的变量的访问进行限制，该变量就是永久变量。

(1) 永久变量的特点如下：

① 永久变量与全局变量类似，但是只能在 M 文件内部定义，它的范围被限制在声明这些变量的函数内部，不允许在其他的函数中对它们进行改变。

② 只有该变量从属的函数能够访问该变量。

③ 当函数运行结束时，该变量的值保留在内存中。只要M文件还在MATLAB的内存中，永久变量就存在。因此当该函数再次被调用时，可以再次利用这些变量。

(2) 永久变量的定义方法：

persistent 变量名1 变量名2

使用格式形如persistent (X Y Z)。

北京郵電大學

```
function myFun()
   persistent n
   if isempty(n)
      n = 0;
   end
   n = n+1
end
```

在命令提示符下，调用myFun 三次。

```
myFun
myFun
myFun

n =
     1
n =
     2
n =
     3
```

persistent 变量用法示例：

编写一个函数，该函数在距离上个日志项至少三秒后开始记录数据。将 logTime 定义为持久性变量，用于存储 logData 上次写入文件的时间。

在当前工作文件夹中的文件中，定义 logData 函数。

```
function logData(fname,n)
    persistent logTime
    currTime = datetime;

    if isempty(logTime)
        logTime = currTime;
        disp('Logging initial value.')
        dlmwrite(fname,n)
        return
    end
    dt = currTime - logTime;
    if dt > seconds(3)
        disp('Logging.')
        dlmwrite(fname,n,'-append')
        logTime = currTime;
    else
      disp(['Not logging. '
num2str(seconds(dt)) ' sec since last log.'])
    end
end
```

测试代码：

```
for n = 1:10
    pause(1)
    logData('myLog.txt',rand)
end
```