第 5 章

CHAPTER 5

M 文件程序设计

MATLAB 作为一种高级语言，不但可以以命令行的方式完成操作，还可以像多数高级编程语言一样具有控制流、输入、输出和面向对象编程的能力，适用于各种应用程序设计。与其他高级语言相比，MATLAB 语言具有语法简单、使用方便和调试容易等优点。

5.1 M 文件

5.1.1 M 文件的建立与编辑

M 文件是由 MATLAB 命令或函数构成的文本文件，以.m 为扩展名，故称为 M 文件。

M 文件是一个文本文件，它可以用任何编辑器来建立和编辑，常用且最为方便的是使用 MATLAB 提供的文本编辑器。

1. 建立新的 M 文件

为建立新的 M 文件，常用两种方法启动 MATLAB 文本编辑器。

(1) 菜单操作。

执行 File→New→blank M-file 菜单命令，会弹出 Editor-Untitled 窗口(如图 5-1 所示)。Editor-Untitled 是一个集编辑与调试两种功能于一体的工具环境。利用它不仅可以完成基本的文本编辑操作，还可以对 M 文件进行调试。

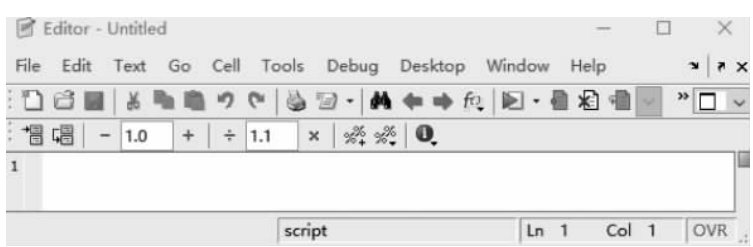

图 5-1 M 文件编辑器窗口

启动 MATLAB 文本编辑器后，在文档窗口中输入 M 文件的内容，输入完毕后，选择 File→Save 或 Save As 命令存盘。注意，M 文件存放的位置一般是 MATLAB 默认的用户工作目录“\matalb\work”，当然也可以存到其他的目录。如果是其他的目录，则应该将该目录设定为当前目录或将其加到搜索路径中。

(2) 命令按钮操作

单击 MATLAB 命令窗口工具栏上的新建命令按钮,启动 MATLAB 文本编辑器后,输入要编制的文件程序并存盘,此方法简单快捷。

M 文件有两种形式:命令文件(Script)和函数文件(Function)。

命令文件是命令和函数的组合,执行命令文件不需要输入参数,也没有输出参数。MATLAB 自动按顺序执行命令文件中的函数命令,命令文件的变量保存在工作空间中。

函数文件是以 Function 语句为引导的 M 文件,可以接收输入参数和返回输出参数,在默认情况下,函数文件的内部变量是临时的局部变量;函数运行结束后,这些局部变量被释放,不再占用内存空间。用户可以根据自己的需要编制函数文件以扩充已有的 MATLAB 功能。可以理解为命令文件相当于主程序,函数文件相当于子程序。子程序与主程序之间的数据是通过参数进行传递的,子程序应用主程序传递过来的参数进行计算之后,将结果返回主程序。

两种形式的 M 文件比较如表 5-1 所示。

表 5-1　命令文件与函数文件比较

	命令文件	函数文件
形式	为一系列命令和函数语句的组合,不需要任何说明和定义	文件中的第一行用 function 说明,然后再编写函数内容
参数	没有输入参数,也不用返回参数	接收输入参数,也可以返回参数
数据	处理的变量为工作空间变量	处理的变量为函数内部的局部变量,也可以处理全局变量
应用	自动完成一系列命令和函数,并可以多次运行。作为普通的运行程序,便于调试和修改	常用于需要反复调用并不断改变参数的场合,可用于扩充 MATLAB 函数库和一些特殊的应用
运行形式	在命令窗口中或编辑器内可直接运行	需要由其他语句调用

5.1.2　命令文件

命令文件没有输入输出参数,是最简单的 M 文件。

新建编辑器,在里面输入如下程序:

```
x = 1:100;
plot(x)
```

然后保存文件,如命名 ncy,可直接单击菜单中运行键 run,则可观察到运行结果。命令文件可调用工作空间中已有的变量或创建新的变量。命令文件运行结束后,所有变量仍然保存在工作空间中,直到被清除(clear)为止。

【例 5-1】 绘制曲线(如图 5-2 所示)。

解:只要给定自变量数值范围,写出变量的相关表达式,即可绘制出曲线。程序如下:

```
t = 0:0.1:50;
x = cos(t) + t. * sin(t);
y = sin(t) - t. * cos(t);
plot(x,y)
```

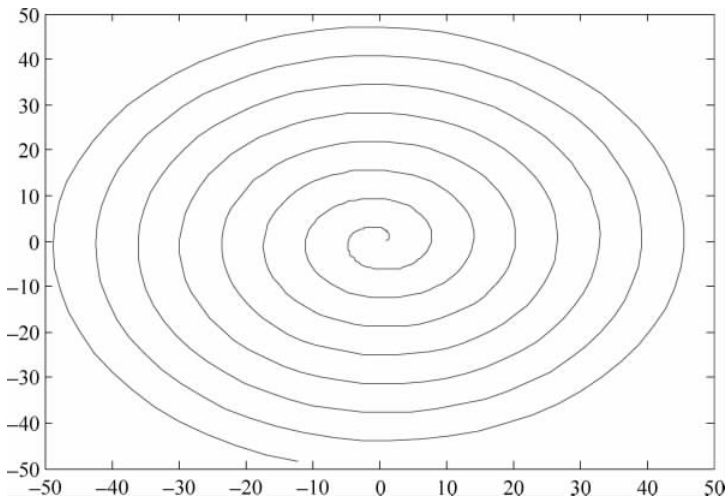

图 5-2　二维曲线

5.1.3　函数文件

函数文件的第一句以 function 开始。每一个函数文件定义一个函数。实际上，MATLAB 提供的函数命令大部分都是由函数文件定义的，这足以说明函数文件的重要性。从使用角度看，函数文件是一个“黑箱”，把一些数据送进去，经加工处理，把结果再送出来。从形式上看，函数文件区别于命令文件在于命令文件的变量在文件执行完成后保留在工作空间中，而函数文件内定义的变量只在函数文件内部起作用，当函数文件执行完后，这些内部变量将被清除。

通常，函数文件由以下基本部分组成。

（1）函数定义行

function 是函数定义的关键字。另外，当函数具有多个输出变量时，则以方括号括起；当函数具有多个输入变量时，则直接用圆括号括起来，例如：

```
function [w,y] = result(m,n)      % result 是函数文件名
```

（2）函数体

指由 MATLAB 命令语句提供的函数和用户自己设计的函数共同构成的语句实体，也是输入和输出之间的关系表达式。

（3）注释

注释行由“%”开始，可以出现在程序中的任意位置，目的是增强程序的可读性，方便程序的调试。

【例 5-2】　建立函数文件，计算矢量中元素的平均值。

解：在新建编辑器中输入如下程序：

```
function y = average(x)
% AVERAGE Mean of vector elements.
% AVERAGE(X),where X is a vector, is the mean of vector elements.
% Non - vector input results in an error.
```

```
[m,n] = size(x);
if (~((m == 1) | (n == 1)) | (m == 1 & n == 1))
    error('Input must be a vector')
end
y = sum(x)/length(x);          % Actual computation
```

将上面的程序保存在名为 average.m 的文件中。可在命令窗口编制生成矢量的语句，并调用 average 函数：

```
>> z = 1:99;            % 生成 100 个数
>> average(z)
ans =
    50
```

从形式上看：函数文件内定义的变量 x 与 y 只在函数文件内部起作用。而命令文件中的变量在文件执行完成后仍然保留在工作空间中。

本例中，注意观察下面两种在命令窗口中调用函数文件的输入格式，如果不对的话，会出现如下错误显示。

```
>> k = [1 2;5 6];
>> average(k)
Error using average (line 4)
input must be a vector

>> k = [9];
>> average(k)
Error using average (line 4)
input must be a vector
```

可见，都显示错误提示，这就是本例中下面语句所起的作用。

```
[m,n] = size(x);
if (~((m == 1) | (n == 1)) | (m == 1 & n == 1))
    error('Input must be a vector')
```

【例 5-3】 编写函数文件，求半径为 r 的圆面积和周长。

解：首先新建编辑器文件，注意函数文件一定要在编辑器里编写，函数文件内容如下：

```
% 输入变量 r 为圆半径、输出变量 s 和 p 分别代表圆面积和圆周长
function [s,p] = fc(r)      % fc 为函数文件名
s = pi * r * r;
p = 2 * pi * r;             % 此语句应有分号，否则在命令窗口调用该函数时会出现多个 p 值
```

将以上函数文件保存，以文件名 fc.m 存入 MATLAB 默认路径下，然后在命令窗口中调用此函数：

```
>> [mianji,zhouchang] = fc(6)          % 调用函数文件 fc
```

运行结果如下：

```
mianji =
  113.0973
zhouchang =
   37.6991
```

注意：[mianji,zhouchang]中两个输出的变量名字是随意设置的，不一定设置为 mianji 和 zhouchang。fc(6)中的 6 是随意输入的半径值。

5.2 程序流程控制

在 MATLAB 中，除了按正常顺序执行程序中的命令和函数以外，还提供了 8 种控制程序流程的语句，即 for、while、if、switch、try、continue、break、return。本节中只介绍 for、while 和 if 的用法。

5.2.1 循环控制语句

在实际工程中会遇到许多有规律的重复运算，因此在程序设计中需要将某些语句重复执行。被重复执行的语句称为循环体，每循环一次，都必须做出是否继续重复的决定，这个决定所依据的条件称为循环的终止条件。MATLAB 提供了两种循环方式：for-end 循环和 while-end 循环。for 循环和 while 循环的区别在于，for 循环结构中循环体的执行次数是确定的，而 while 循环结构中循环体执行的次数是不确定的。

1. for 循环

for 语句为计数循环语句，for 循环允许一组命令以固定的和预定的次数重复，也就是说，已知循环次数情况下，采用 for 循环。for 循环的一般形式为

```
for v = 表达式
语句体
end
```

MATLAB 的 for 循环与其他计算机语言一样，for 和 end 必须配对使用。

【例 5-4】 极坐标绘制花瓣图可用 for 循环语句完成（如图 5-3 所示）。

解：程序代码如下：

```
theta = -pi:0.01:pi;
rho(1,:) = 2 * sin(5 * theta).^2;
rho(2,:) = cos(10 * theta).^3;
rho(3,:) = sin(theta).^2;
rho(4,:) = 5 * cos(3.5 * theta).^3;
for k = 1:4                          %循环 4 次
    subplot(2,2,k)
    polar(theta,rho(k,:))            %绘制极坐标图
end
```

若不用循环语句，则反复用 subplot 函数也可完成，其程序如下：

```
theta = -pi:0.01:pi;
rho(1,:) = 2 * sin(5 * theta).^2;
rho(2,:) = cos(10 * theta).^3;
rho(3,:) = sin(theta).^2;
rho(4,:) = 5 * cos(3.5 * theta).^3;
subplot(2,2,1)    polar(theta,rho(1,:)) %绘制极坐标图
subplot(2,2,2)    polar(theta,rho(2,:))
subplot(2,2,3)    polar(theta,rho(3,:))
subplot(2,2,4)    polar(theta,rho(4,:))
```

同样可以绘制出花瓣图形，如图 5-3 所示。

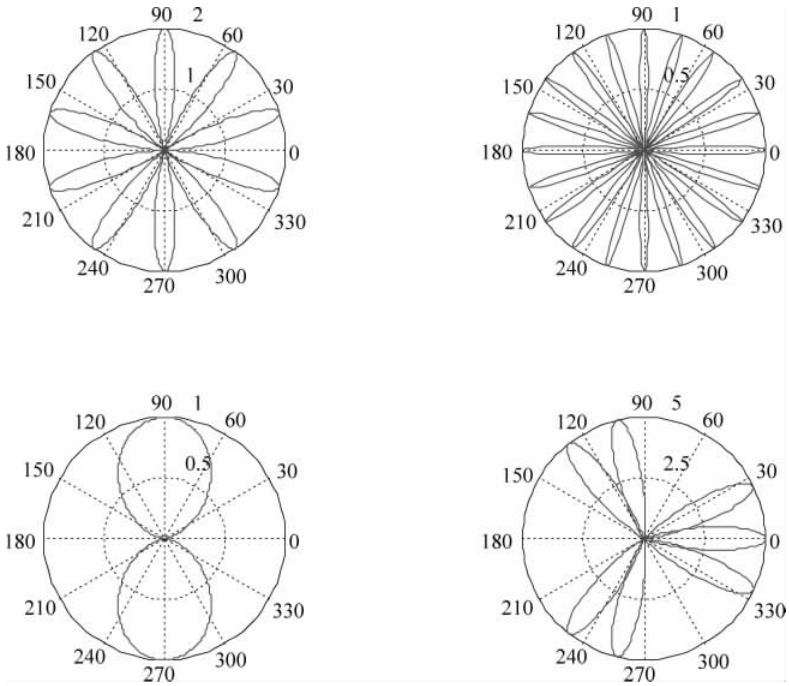

图 5-3　花瓣图

【例 5-5】 简单的 for 循环示例。在新建编辑器中输入下列命令，运行 for 循环。

解：程序代码如下：

```
>> for n = 1:8
        x(n) = sin(n * pi/10);
    end
```

在命令窗口输入 x，即

```
>> x   % 输入 x,然后按 Enter 键
x =
    0.3090    0.5878    0.8090    0.9511    1.0000    0.9511    0.8090    0.5878
```

说明：设置 n，n 从 1 到 8，求所有语句的值，直至下一个 end 语句。第 1 次通过 for 循环n＝1；第 2 次，n＝2，如此继续，直至 n＝8。在 n＝8 以后，for 循环结束，然后运行 end 语句后面的命令。此例结果显示所计算的 x 元素值。

【例 5-6】 for 循环可以嵌套，在嵌套中每一个 for 都必须与 end 相匹配，否则循环将出错。

解：程序代码如下：

```
for n = 3:6
    for m = 7: - 1:4
    x(n,m) = n^2 + m^2;              % x(n,m)表示 n 行 m 列矩阵
    end
  disp(n)                            % 显示 n 值
end
```

按 Enter 键，则出现

```
     3
     4
     5
     6

>> x                                  % 输入 x，按 Enter 键则出现

x =
     0     0    10    17    26    37    50
     0     0    13    20    29    40    53
     0     0    18    25    34    45    58
     0     0    25    32    41    52    65
     0     0    34    41    50    61    74
     0     0     0    52    61    72    85
```

例 5-6 中，n 最大为 6，m 最大为 7，所以产生的矩阵为 6 行 7 列，空项自动补 0。

2. while 循环

while 语句是条件循环语句，while 循环以不定的次数求一组语句的值，直到循环条件不成立为止，即只要在表达式里的所有元素为真，就执行 while 和 end 语句之间的语句。while 循环的一般形式是：

```
while 表达式
    语句体
    end
```

while 和 end 必须配对使用。

【例 5-7】 利用 while 循环，求解使 $n!$ 达到 100 位数的第一个 n 是多少。

解：在新建编辑器中输入下列程序代码，求解 n：

```
n = 1;
while prod(1:n)< 1e100       % 阶乘小于 100 位的第一个数
n = n + 1;
end
```

保存程序，单击 run 运行，然后在命令窗口输入 n，即

```
>> n
n =
    70
```

【例 5-8】 计算从 1 开始，多少个自然数的和大于 200。

解：在新建编辑器中输入下列程序代码：

```
s = 0;
m = 0;
while s< = 200
m = m + 1;
s = s + m;
end
```

保存程序，单击“运行”按钮，然后在命令窗口输入 s，m，即

```
>> s,m
s =
   210
m =
    20
```

5.2.2　条件控制语句

在复杂的运算中常常需要判断是否满足某些条件，以选择下一步的方法和策略。一般使用条件语句完成这类判断和选择。

1．if-end 语句

if-end 语句是最简单的条件语句，其一般形式为

```
if 表达式
    语句体
end
```

关键字 if 后的表达式确定了判断条件。如果表达式满足条件，则执行 if 和 end 之间的所有语句，否则跳出 if 结构，执行 end 后面的语句。

2．if-else-end 语句

if-else-end 语句在 if 和 end 之间增加一个 else 选择，语句的一般形式为

```
if 表达式
    语句体 1;
else
    语句体 2;
end
```

当表达式的值为真时，执行语句体 1，否则执行语句体 2。

3．if-elseif-end 语句

在 else 子句中也可嵌套 if 语句，构成 elseif 结构，elseif 结构实际上实现了多重条件选择，其一般形式为

```
if 表达式 1
语句体 1;
elseif 表达式 2
语句体 2;
else
语句体 3;
end
```

在这种结构中，首先计算表达式 1，如果条件满足执行语句体 1，然后跳出 if 结构，如果不满足表达式 l 的条件，再计算表达式 2，如果表达式 2 的条件满足，则执行语句体 2，然后跳出 if 结构，如果前面的表达式都不满足，就执行语句体 3。

根据程序设计的需要可以使用多个 elseif 语句，也可以省略 else 语句。

【例 5-9】 利用色彩及标记区分数据点的范围(如图 5-4 所示)。

解：在编辑器中编制下列程序,并绘图。

```
n = 100;
x = 1:n;
y = randn(1,n);                    % 创建 100 个数据的随机行矢量
hold on
for i = 1:n
if y(i)< - 1
    plot(x(i),y(i),' * g')         % 小于 - 1 的点用绿色的 * 标出
elseif y(i)> = - 1 & y(i)< = 1
    plot(x(i),y(i),'ob')           % - 1～1 的点用蓝色的 o 标出
else y(i)> 1
    plot(x(i),y(i),'xr')           % 大于 1 的点用红色的 x 标出
    end
end
hold off
```

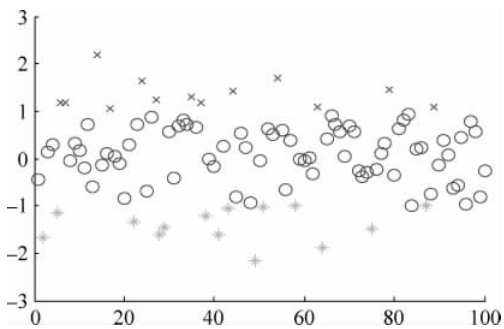

图 5-4 用色彩及标记区分数据点的范围

习题 5

5-1 命令文件与函数文件的主要区别是什么?

5-2 编制函数功能文件 $y=x^2+\cos(x)$；在命令窗口随意设置自变量,调用该函数文件,求运行结果。

5-3 编制函数文件实现直角坐标(x,y)与极坐标(ρ,θ)之间的转换。已知极坐标的矢径 $\rho=\sqrt{x^2+y^2}$,极坐标的极角 $\theta=\arctan\left(\frac{y}{x}\right)$。