



Matlab语言程序设计

第5讲



◆ 主要内容

- M文件概述
- 数据的交互输入输出
- 函数文件
- 程序调试
- 函数句柄和匿名函数



5.1.1 Matlab命令的执行方式

- 交互式命令执行方式（命令窗口）

逐条输入，逐条执行，操作简单、直观，但速度慢，执行过程不能保留。

- M文件的程序执行方式

将命令编成程序存储在一个文件中（M文件），依次运行文件中的命令，可以重复进行。

- Matlab程序设计有传统高级语言的特征，又有自己独特的特点，可以利用数据结构的优点，使程序结构简单，编程效率高。



5.1.2 M文件的分类

- 用Matlab语言编写的程序，称为M文件。
是由若干Matlab命令组合在一起构成的，它可以完成某些操作，也可以实现某种算法。
- M文件根据调用方式的不同分为两类：
 - 命令文件（Script File）
 - 函数文件（Function File）
- 它们的扩展名都是.m

5.1.3 命令文件和函数文件的区别

- 命令文件没有输入参数，也不返回输出参数；函数文件可以带输入参数，也可以返回输出参数。
- 命令文件对工作空间中的变量进行操作，文件中所有命令的执行结果也返回工作空间中；函数文件中定义的变量为局部变量，当函数文件执行完毕时，这些变量也被清除。
- 命令文件可以直接运行；函数文件不能直接运行，要以函数调用的方式来调用它。

例5.1 建立文件将变量a、b的值互换。

```
clear
```

```
clc
```

```
a = 1:10;
```

```
b = [11,12,13,14;15,16,17,18];
```

```
c = a; a = b; b = c;
```

```
a
```

```
b
```

将文件保存为exch，并在命令窗口执行。

执行结果：

```
a =
```

```
11 12 13 14
```

```
15 16 17 18
```

```
b =
```

```
1 2 3 4 5 6 7 8 9 10
```

函数文件

fexch.m

```
function [a,b] = fexch(a,b)
```

```
c = a; a = b; b = c;
```

然后在命令窗口调用该函数文件：

```
clear;
```

```
x = 1:10;
```

```
y = [11,12,13,14;15,16,17,18];
```

```
[x,y] = fexch(x,y)
```

输出结果为：

a =

11 12 13 14

15 16 17 18

b =

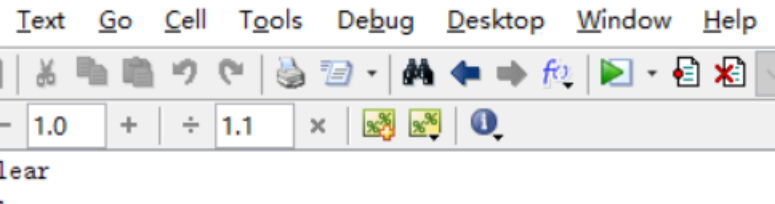
1 2 3 4 5 6 7 8 9 10

函数参数**a**，**b**，**c**未保留在工作空间中，**x**，**y**保留在工作空间中。

5.1.4 M文件的建立与打开

M文件是一个文本文件，可以用任何编辑程序来建立和编辑，一般使用Matlab提供的文本编辑器。

该编辑器是一个集编辑和调试于一体的工作环境。



The screenshot shows the MATLAB Editor window with the following content:

```
1 clear
2 clc
3 h0=0.1;
4 x1=0;
5
6 y1=hanshu(x1);
7 h=h0;
8 x2=x1+h;
9 y2=hanshu(x2);
10 if x2<=1
```

The function definition at the bottom is:

```
function y=hanshu(x)
y=x^2+1;
```

The status bar at the bottom indicates the current line is 8 and column is 9.

5.2 数据的交互输入

1、数据的交互输入

数据的交互输入，可以使用**input**函数。

调用格式为：

A = input(提示信息, 选项)

其中提示信息为一个字符串，用于提示用户输入数据。

例如：从键盘输入A矩阵，可以采用下面的命令来完成

A = input('输入A矩阵')

输入字符串，在input函数调用时采用's'选项。

例：**xm = input(' What''s your name?', 's')**

5.2 数据的交互输出

2、数据的输出

数据的输出，可以使用disp函数，调用格式为：

disp(输出项)

其中输出项既可以为字符串，也可以为矩阵。例如：

A = 'Hello, Tom';

disp(A)

输出为：Hello, Tom

又如：**A = [1,2,3;4,5,6;7,8,9];**

disp(A)

输出为：

1 2 3

4 5 6

7 8 9

%disp使输出格式更紧凑

例5.2 数据的交互输入输出

例5.2 求一元二次方程 $ax^2 + bx + c = 0$ 的根。

由于Matlab能进行复数运算，所以不需要判断方程的判别式，可直接根据求根公式求根。

```
a = input('a=?');  
b = input('b=?');  
c = input('c=?');  
d = b*b-4*a*c;  
x = [(-b+sqrt(d))/(2*a),(-b-sqrt(d))/(2*a)];  
disp(['x1=',num2str(x(1)),',x2=',num2str(x(2))]);
```

程序输出为：

```
a=?4  
b=?78  
c=?54  
x1=-0.7188,x2=-18.7812
```



5.3 程序控制结构

MATLAB有3种基本的程序结构：

- 顺序结构
- 选择结构
- 循环结构

任何复杂的程序都可以由这3种基本结构构成。



5.3.1 程序的暂停

程序的暂停

程序执行过程中暂停，可用`pause`函数，其调用格式为：

`pause`（延迟描述）

如果省略延迟时间，直接使用`pause`，则将暂停程序，直到用户按任一键后程序继续执行。

若要强化中止程序的运行可按**Ctrl+C**键。



5.3.2 选择结构

选择结构是根据给定的条件成立或不成立，分别执行不同的语句。

Matlab用于实现选择结构的语句有

- **if语句**
- **switch语句**
- **try语句**

5.3.2 选择结构

1. if语句

在Matlab中，if语句有3种格式。

(1)单分支if语句

语句格式：

if 条件

语句或语句组

end

 **关键字end不能少**

例如：当x大于等于70且小于80时，x的值放大1.2倍

```
if x>=70& x<80
```

```
    x=1.2*x;
```

```
end
```



5.3.2 选择结构

(2) 双分支if语句

语句格式:

if 条件

语句组 1

else

语句组 2

end

当条件成立时，执行语句组1，否则执行语句组2，然后再执行if语句的后续语句。

5.3.2 选择结构

例5.3 计算分段函数：

$$y = \begin{cases} \cos(x+1) + \sqrt{x^2+1}, & x \leq 10 \\ x\sqrt{x+\sqrt{x}}, & x > 10 \end{cases}$$

程序如下：

```
x = input('请输入x的值: ');  
if x <= 10  
    y = cos(x+1)+sqrt(x*x+1);  
else  
    y = x*sqrt(x+sqrt(x));  
end  
y
```

也可以用单分支if语句来实现：

```
x = input('请输入x的值: ');  
y = cos(x+1)+sqrt(x*x+1);  
if x>10  
    y = x*sqrt(x+sqrt(x));  
end  
y
```

5.3.2 选择结构

(3) 多分支if语句

语句格式:

if 条件1

语句组 1

elseif 条件2

语句组 2

...

elseif 条件m

语句组 m

else

语句组n

end

🔔 条件1、条件2...条件
m是并列关系

5.3.2 选择结构

if语句的嵌套

if 条件1

if 条件2

语句1

elseif 条件3

语句2

else

语句3

end

else

if 条件4

语句4

else

语句5

end

end

🔔 注意

条件1与条件2是递进关系

条件2与条件3是并列关系

5.3.2 选择结构

交互式输入三个边长。判定能否构成三角形以及所构成的三角形的类型，分别输出“非三角形”、“一般三角形”、“等腰三角形”或“等边三角形”。

```
a = input('请输入第一个边长: a=');
b = input('请输入第二个边长: b=');
c = input('请输入第三个边长: c=');
if ((a+b)>c&&(a+c)>b&&(b+c)>a)
    if a==b&&b==c
        disp('等边三角形')
    elseif a==b||b==c||a==c
        disp('等腰三角形')
    else
        disp('一般三角形')
    end
else
    disp('不能构成三角形')
end
```

```
a=input('请输入第一条边长: a=');
b=input('请输入第二条边长: b=');
c=input('请输入第三条边长: c=');
if ((a+b)>c&&(a+c)>b&&(b+c)>a)
    if a==b||b==c||a==c
        disp('等腰三角形')
        if a==b&&b==c
            disp('等边三角形')
        end
    else
        disp('一般三角形')
    end
else
    disp('不能构成三角形')
end
```

5.3.2 选择结构

2、switch语句

switch语句根据表达式的取值不同，分别执行不同的语句，其语句格式：

switch 表达式

case 表达式1

语句组1

case 表达式2

语句组2

...

case 表达式m

语句组m

otherwise

语句组 n

end

switch子句后面的表达式的结果是一个数或一个字符串；case子句后面的内容可以为一个数或一个字符串，或者是由多个数或多个字符串构成的元胞。



5.3.2 选择结构

某商场对顾客所购买的商品实行打折销售，标准如下：

$\text{price} < 200$ 没有折扣

$200 \leq \text{price} < 500$ 3%折扣

$500 \leq \text{price} < 1000$ 5%折扣

$1000 \leq \text{price} < 2500$ 8%折扣

$2500 \leq \text{price} < 5000$ 10%折扣

$5000 \leq \text{price}$ 14%折扣

输入所售商品的价格，求其实际销售价格。

5.3.2 选择结构

```
price = input('请输入商品价格' ) ;
```

```
switch fix(price/100)
```

```
    case{0,1}                %价格小于200
```

```
        rate = 0;
```

```
    case{2,3,4}
```

```
        rate = 3/100;        %价格大于等于200但小于500
```

```
    case num2cell(5:9)
```

```
        rate = 5/100;        %价格大于等于500但小于1000
```

```
    case num2cell(10:24)
```

```
        rate = 8/100;        %价格大于等于1000但小于2500
```

```
    case num2cell(25:49)
```

```
        rate = 10/100;       %价格大于等于2500但小于5000
```

```
    otherwise
```

```
        rate = 14/100;       %价格大于等于5000
```

num2cell函数将数值矩阵转化为元胞数组。

```
end
```

```
price = price*(1-rate)      %输出商品实际销售价格
```



5.3.2 选择结构

3. try语句

try语句是一种试探性执行语句，其语句格式为：

try

语句组1

catch

语句组2

end

try语句先试探性执行语句组1，如果在执行过程中出现错误，则将错误信息赋给保留的lasterr变量，并转去执行语句组2。

5.3.2 选择结构

矩阵乘法运算要求两矩阵的维数相容，否则会出错。先求两矩阵的乘积，若出错则自动转去求两矩阵的点乘。

```
A = [1,2,3;4,5,6];
```

```
B = [7,8,9;10,11,12];
```

```
try
```

```
    C = A*B;
```

```
catch
```

```
    C = A.*B;
```

```
end
```

```
C
```

```
lasterr      %显示出错原因
```

```
C =
```

```
    7    16    27
```

```
   40    55    72
```

```
ans =
```

```
Error using ==> mtimes
```

```
Inner matrix dimensions  
must agree.
```



5.3.3 循环结构

循环是指按照给定的条件，重复执行指定的语句。Matlab 提供了两种实现循环结构的语句：**for**语句和**while**语句。

1、for语句

for语句的格式为：

for 循环变量 = 表达式1：表达式2：表达式3

 循环体语句

end

其中表达式1的值为循环变量的初值，表达式2的值为步长，表达式3的值为循环变量的终值。步长为1时，表达式2可以省略。

5.3.3 循环结构

已知 $y = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \cdots + \frac{1}{n^2}$, 当n=100时, 求y的值。

```
y = 0;n = 100;
```

```
for i=1:n
```

```
    y = y+1/i^2;
```

```
end
```

```
y
```

输出结果为:

```
y =
```

```
    1.6350
```

利用Matlab的特点, 常用向量运算来代替循环操作, 程序可以如下:

```
n = 100;
```

```
i = 1:n;
```

```
f = 1./i.^2;
```

```
y = sum(f)
```

5.3.3 循环结构

for循环是根据数组的列数决定其循环执行的次数。每执行一次，取数组的一列作为变量的值。执行结束后，就取下一列作为变量的值，直到数组的最后一列。

```
vx=[7 3 10 5;1 2 3 4];
```

```
k=0;
```

```
for x=vx
```

```
    k=k+1;
```

```
    vy(:,k)=x.^2;
```

```
end
```

```
k
```

```
disp([vx;vy])
```

只有当为多列数组，**for**循环才反复执行多次！

```
i=0;
```

```
for n=(1:10)'
```

```
    i=i+1;
```

```
    x(n)=sin(n*pi/10);
```

```
end
```



5.3.3 循环结构

2、while语句

while语句的一般格式为：

while 条件

 循环体语句

end

其执行过程为：

若条件成立，则执行循环体语句，执行后再判断条件是否成立，如果不成立则跳出循环。

5.3.3 循环结构

从键盘输入若干个数，当输入0时结束输入，求这些数的平均值和它们的和。

```
sum = 0;
n = 0;
x = input('Enter a number(end in 0):');
while(x~=0)
    sum = sum+x;
    n = n+1;
    x = input('Enter a number(end in 0):');
end
if(n>0)
    sum
    mean = sum/n
end
```

输出结果为：

```
Enter a number(end in 0):67
Enter a number(end in 0):89
Enter a number(end in 0):93
Enter a number(end in 0):70
Enter a number(end in 0):0
sum =
    319
mean =
    79.7500
```



Matlab语言程序设计

第6讲



5.3.3 循环结构

3、break语句和continur语句

一般与if语句配合使用。

break语句用于终止循环的执行。

当在循环体内执行到该语句时，程序将跳出循环，继续执行循环语句的下一语句。

continue语句控制跳过循环体中的某些语句。

当在循环体内执行到该语句时，程序将跳过循环体中所有剩下的语句，继续下一次循环。



5.3.3 循环结构

例：求[100,200]之间第一个能被21整除的整数。

```
for n = 100:200  
    if rem(n,21)~=0;  
        continue  
    end  
    break  
end  
n
```

程序输出结果为：

```
n =  
  
    105
```



5.3.3 循环结构

break 示例1

```
var=[1 3 5 -1 0 7 8 0];
```

```
a=[];
```

```
k=1;
```

```
while var(k)
```

```
    if var(k)==5
```

```
        break
```

```
    end
```

```
    a= var(k).^2
```

```
    k=k+1;
```

```
end
```

5.3.3 循环结构

break 示例2

```
i=0;j=0;k=0;
```

```
for i=1:2
```

```
    for j=1:2
```

```
        for k=1:2
```

```
            if (k==2)
```

```
                disp('退出循环');
```

```
                break;
```

```
            end
```

```
            str=sprintf('I=%d, J=%d, K=%d',i,j,k);
```

```
            disp(str);
```

```
        end
```

```
    end
```

```
end
```

```
disp('程序运行结束');
```



5.3.3 循环结构

continue 示例

```
i=0;
```

```
for i=1:6
```

```
    if (i>3)
```

```
        continue
```

```
    else
```

```
        str=sprintf('I=%d',i);
```

```
        disp(str);
```

```
    end
```

```
end
```

```
str=sprintf('循环结束I=%d',i);
```

```
disp(str);
```



5.4 函数文件

函数文件也是一种M文件，函数文件定义函数。
Matlab提供的标准函数大部分是由函数文件定义的。

5.4.1 函数文件的基本结构

函数文件由function语句引导，其基本结构为：

function 输出形参表 = 函数名 (输入形参表)

注释说明部分

函数体语句

其中，以function关键字开头的一行为引导行，表示该M文件是一个函数文件。

当输出形参多于一个时，应该用方括号括起来。

5.4.1 说明

1. 关于函数文件名

函数的M文件名称必须和函数名称保持一致。

2. 关于注释说明部分

注释说明包括3部分：

① 紧随引导行之后以%开头的第一注释行。

函数功能简要描述，作为lookfor 和help查询的内容。

② 第一注释行及之后连续的注释行。

函数输入/输出参数的含义及调用格式说明。

③ 与在线帮助文本相隔一空行的注释行。

函数文件编写和修改的信息，如作者和版本等。

3. 关于return语句

如果在函数文件中插入了return语句，则执行到该语句就结束函数的执行，流程转至调用该函数的位置。



5.4.1 说明

例5.10 编写函数文件，求半径为 r 的圆的面积和周长。

函数文件如下：

```
function [s,p] = fcircle(r)
```

```
% FCIRCLE calculate the area and perimeter of a circle  
of radii r
```

```
% r    圆半径
```

```
% s    圆面积
```

```
% p    圆周长
```

```
%2006年2月30日编
```

```
s = pi*r*r;
```

```
p = 2*pi*r;
```



5.4.1 说明

将以上函数文件以文件名fcircle.m保存，然后在命令窗口调用。

```
[s,p] = fcircle(10)
```

输出结果是：

s =

314.1593

p =

62.8319

采用help命令或lookfor命令可以显示出注释说明部分的内容。

```
help fcircle
```

屏幕显示

FCIRCLE calculate the area and perimeter of a circle of radii r

r 圆半径

s 圆面积

p 圆周长

5.4.2 函数调用

函数调用的一般格式是：

[输出实参表] = 函数名(输入实参表)

注意：函数调用时，各实参出现的顺序、个数，应与函数定义时相同。

例5.11 利用函数文件，实现直角坐标(x,y)与极坐标(ρ, θ)之间的转换。

函数文件：tran.m:

```
function [rho,theta] = tran(x,y)
rho = sqrt(x*x+y*y);
theta = atan(y/x);
```

调用tran.m的命令文件main1.m:

```
x = input('please input x=:');
y = input('please input y=:');
[rho,the] = tran(x,y);
rho
the
```

5.4.3 函数的嵌套调用

在Matlab中，函数可以嵌套调用，即一个函数可以调用别的函数。

一个函数调用自身称为函数的递归调用。

例5.12 利用函数的递归调用，求n!。

n! 本身就是以递归的形式定义的：

$$n! = \begin{cases} 1, & n \leq 1 \\ n(n-1)!, & n > 1 \end{cases}$$

显然，求n! 需要求(n-1)!, 这时可采用递归调用。

函数如下：

```
function f = factor(n)
```

```
if n<=1
```

```
    f = 1;
```

```
else
```

```
    f = factor(n-1)*n; %递归调用求(n-1)!
```

```
end
```



5.4.2 函数调用

在命令文件中调用该函数文件，求 $s = 1! + 2! + 3! + 4! + 5!$ 。

```
s = 0;
```

```
for i = 1:5
```

```
    s = s + factor(i);
```

```
end
```

```
s
```

在命令窗口运行命令文件，结果如下：

```
s =
```

```
153
```

5.4.3 函数参数的可调性

与其他高级语言不同，Matlab中函数所传递参数数量可以改变。Matlab用两个预定义变量nargin和nargout分别记录调用该函数时的输入实参和输出实参的个数。

例5.13 nargin用法示例

函数文件examp.m:

```
function fout = examp(a,b,c)
if nargin == 1
    fout = a;end
if nargin == 2
    fout = a+b;end
if nargin == 3
    fout = (a*b*c)/2;
end
```

命令文件:

```
x = [1:3];
y = [1;2;3];
examp(x)
examp(x,y')
examp(x,y,3)
```

5.4.4 局部变量和全局变量

Matlab中，函数文件中的变量是**局部变量**。

如果不同函数共用同一变量，可以把这个变量定义为全局变量。

所有函数都可以对它进行存取和修改。

全局变量用global命令定义，格式为：

global 变量名

例5.13 全局变量应用示例。

先建立函数文件score.m，该函数将输入的参数加权相加：

function f = score(x,y)

global APHA BETA

f = APHA*x + BETA*y;

在命令窗口中输入：

global APHA BETA

APHA = 1;

BETA = 2;

s = score(1,2)

输出为：

s =

5

5.5 程序调试

Matlab提供的程序调试功能包括：通过文本编辑器，以及在命令窗口结合具体的命令对程序进行调试。

5.5.1 程序调试概述

应用程序的错误有两类：**语法错误**和**运行时的错误**。

语法错误，例如：

```
A = 87;
```

```
B = 9.3;
```

```
C = A+*B;
```

系统给出相应的错误信息，并标出错误在程序中的行号。

```
??? Error: File: Untitled1.m Line: 3 Column: 7
```

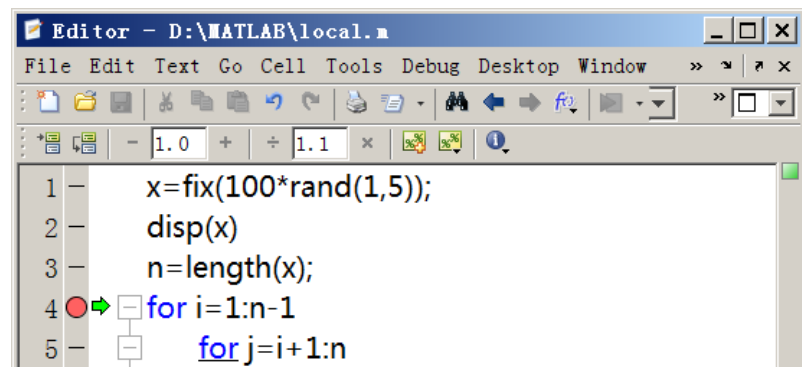
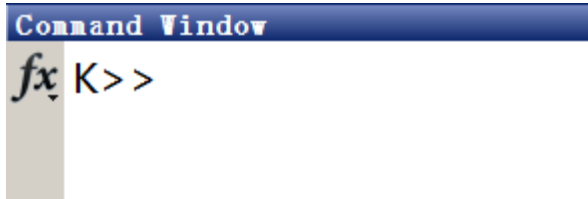
```
Unexpected MATLAB operator.
```

5.5.2 程序调试过程

程序调试一般是针对程序的逻辑错误

程序调试的一般过程：

- ① 设置断点，让程序在断点处暂停运行。
- ② 进入调试状态控制程序运行，检查中间结果。
- ③ 根据结果分析逻辑错误产生的原因。
- ④ 根据分析的结果，修改程序。



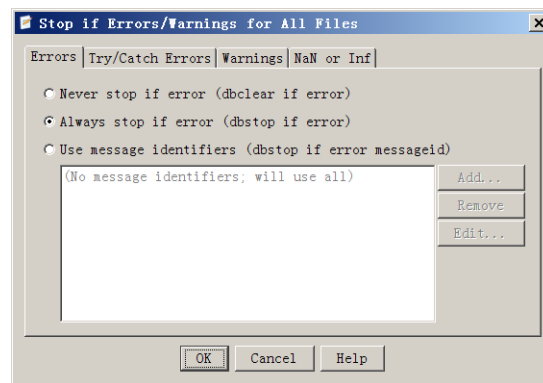
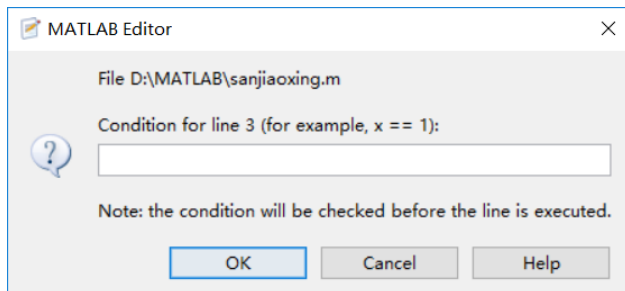
5.5.3 程序断点

程序断点主要有三类：

标准断点：当程序执行到断点位置对应的代码时进入到调试模式。

条件断点：当程序执行到某个条件满足时进入到调试模式。

错误断点：当程序执行时，如果出现了错误或者警告信息进入到调试模式。比如计算结果是NaN、Inf等情况。可以和try配合使用。



5.5.4 程序调试的运行控制

在调试状态控制程序的运行

step: 单步运行，不进入函数；



step in: 单步运行，进入函数；



step out: 如果已经进入函数，则运行完剩余部分代码，退出函数；否则直接运行到下一个断点。



```
Editor - D:\MATLAB\local.m
File Edit Text Go Cell Tools Debug Desktop Window Help
[Icons] Stack: Base fx
1 - x=fix(100*rand(1,5));
2 - disp(x)
3 - n=length(x);
4 - for i=1:n-1
5 -     for j=i+1:n
6 -         if x(i)<x(j)
7 -             y=x(i);x(i)=x(j);x(j)=y;
8 -         end
```

5.5.1 程序调试快捷键

快捷键

F12: 设置/取消断点  `for i=1:n-1` 

F10: 单步运行, 不进入函数

F11: 单步运行, 进入函数

Shift + F11: 进入function之后, 通过该指令退出function

F5: 直接跳到下一断点, 如果断点在for循环中, 则F5一次, 循环执行一次

Shift + F5: 退出断点调试

5.6 函数句柄和匿名函数

函数句柄(function handle)是MATLAB中的一类特殊的数据结构，将一个函数封装成一个变量，使其能够像其它变量一样在程序的不同部分传递。

利用函数句柄创建匿名函数，基本语法格式：

```
hmps=@(x) 1./(sin(x)+cos(x))
```

```
hmps=@(x,y) 1./(sin(x)+cos(y))
```

@符号表示等号左边是一个函数句柄。@后面的(参数列表)定义了函数的输入参数，最后一部分是函数表达式。

5.6 函数句柄和匿名函数

函数句柄的好处

- ①提高函数运行速度。
- ②可以与变量一样方便地使用。

```
>>fhandle=@sin
```

```
>>fhandle(pi)
```

等同于sin(pi)

```
>>fhandle=@cos
```

```
>>fhandle(pi)
```

等同于cos(pi)

```
fun = @(x) (x(1) - 0.2)^2 + (x(2) - 1.7)^2 + (x(3) - 5.1)^2;
```

```
x = ga(fun,3,[],[],[],[],[],[],[],[2 3])
```