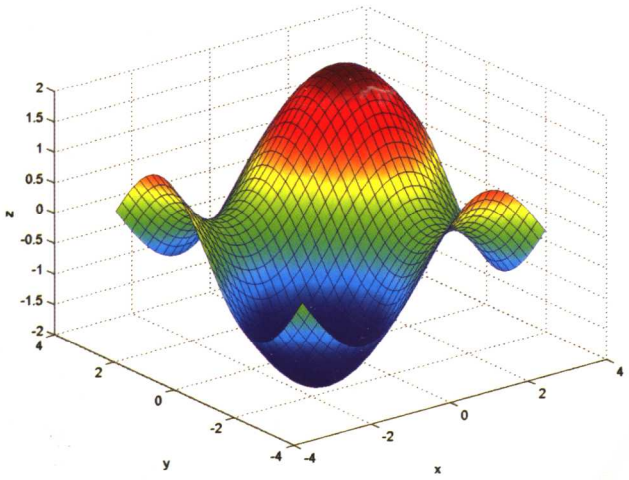
精通Matlab与C/C++混合程序设计

刘 维 编著



北京航空航天大学出版社

CD-ROM INCLUDED

精通 Matlab 与 C/C++混合程序设计

刘 维 编著

北京航空航天大学出版社

内 容 简 介

本书主要介绍如何运用 Matlab 与 C/C++进行混合程序设计。本书全面详细介绍了 Matlab C 数学库、Matlab C++数学库、Matcom、Matlab COM Builder、Matlab Engine 及编译 Matlab 独立可执行程序等 Matlab 混合程序设计的内容。

本书共分为 7 章，主要内容包括：Matlab 编程的基础知识、Matlab C 语言接口、如何生成可独立运行的 Matlab 程序、在 Visual C++中调用 Matlab 程序、Matcom、Matlab COM Builder 与 Visual C++混编程以及在 Visual C++中调用 Matlab C++数学库。本书各章都包含大量的实例程序，可供寻求将 Matlab 程序脱离 Matlab 环境的 Matlab 程序设计人员、寻求高效算法库的 C/C++开发人员学习和参考。

本书采用的开发和运行环境为：Visual C++ 6.0 与 Matlab 6.5。

图书在版编目(CIP)数据

精通 Matlab 与 C/C++混合程序设计/刘维编著. —北京：
北京航空航天大学出版社，2005.6
ISBN 7-81077-626-6

Ⅰ.精… Ⅱ.刘… Ⅲ.①算法语言—程序设计
②C 语言—程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2005)第 032772 号

精通 Matlab 与 C/C++混合程序设计

刘 维 编著

责任编辑 张冀青

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(100083) 发行部电话：(010)82317024 传真：(010)82328026

http://www.buaapress.com.cn E-mail:bhpress@263.net

涿州市新华印刷有限公司印装 各地书店经销

*

开本：787×1092 1/16 印张：19.75 字数：506 千字

2005 年 6 月第 1 版 2005 年 6 月第 1 次印刷 印数：5 000 册

ISBN 7-81077-626-6 定价：36.00 元

前　言

最早接触 Matlab 是在大学期间参加“大学生数学建模竞赛”的时候，那时候惟一的感觉就是“相见恨晚”。接着在读研究生做课题的时候，开始使用 Matcom 编译 Matlab 程序以期获得更快的处理速度，使用 Matcom 的 C++矩阵库以期在享受高效率矩阵运算库的同时，实现和 Visual C++ 6.0 开发环境的无缝连接。后来，MathWorks 公司将 Matcom 收购并将其功能整合到 Matlab 中。直到 Matlab 6.5 的推出，Matlab 与 C/C++混合编程增加了 Matlab C 语言接口、Matlab C++数学库、Matlab COM Builder 和 Matcom 等诸多内容，由此 Matlab 与 C/C++进行混合程序设计的方法也派生出诸多“门派”。

Matlab C 语言接口即 Matlab 提供的一组 C 语言 API 函数以供用户调用。这组 C 语言 API 函数是 Matlab 和用户 C 代码之间的桥梁。用户可以在 Matlab 的 MEX 文件中调用 C 语言 API 函数，也可以在纯 C/C++开发环境中调用 C 语言 API 函数。

Matlab C++数学库是 Matlab 提供的一组封装好的矩阵运算数学库，其使用方法和 Matlab 环境中的编写方法非常相似，如果用户用 VC++（为了书写方便，书中出现的 VC++是 Visual C++的简写）实现用户界面，而又希望寻找一组高效的矩阵运算数学库的话，Matlab C++数学库是一个不错的选择。

Matlab COM Builder 可以将 Matlab 的用 *.m 文件表达的函数编译为 COM 组件。这也是 MathWorks 公司推荐的一种进行 Matlab 混合编程的方法。很多用 Matlab 编译器编译有错误的文件用 Matlab COM Builder 却能很好地解决。只是具体操作起来，在 VC++中调用 COM 组件比调用 C++数学库和 C 语言 API 函数略微复杂。

Matcom 是第一个可以将 Matlab 的 *.m 文件编译为 C/C++代码的工具。现在，MathWorks 公司已经将其集成到 Matlab 中，没有必要再使用 Matcom 来编译 *.m 文件了。但是 Matcom 的 C++矩阵库仍然有使用的价值，相对于 Matlab C++数学库来说，其使用起来更为简单和方便。

可以看出，上述各种 Matlab 与 C/C++混合程序设计的方法各有千秋，具体使用时还要结合开发者的具体情况。但无论使用哪种方法，Matlab 的数据结构与 C/C++的数据结构之间的相互访问和转换都是关键，这也是本书的重点所在，希望读者在读本书的过程中注意。

本书的所有源代码都可以在附带的光盘中找到。

另外，为了与书中程序对应及保证全文体例上的统一，本书中的符号全部采用正体书写。

由于作者的水平有限，如果读者对本书的内容有疑问或者发现书中有错误的地方，请发送邮件到 matlab_vc_program@yahoo.com.cn 与作者讨论或批评指正，谢谢！

在本书的编写过程中得到了很多同志的支持与帮助。特别感谢李璐、李群、路瑞强、伍炜、周志勇、王国房六位同志，由于本书涉及 Visual C++ 6.0 与 Matlab 程序设计的诸多方面，很多关键问题都是在与六位同志的讨论中解决的。不仅如此，他们还为本书提供了很多的宝贵资料。感谢齐春溪女士不辞辛劳地完成了本书所有章节的初步排版工作，并且找出了本书初稿中的诸多错误。

最后感谢所有对本书的完成提供过帮助的人们，没有他们的帮助和付出，本书也不可能完成。

作　者

2004 年 12 月

目　　录

第 1 章　Matlab 程序设计初步

第 2 章　Matlab 与 C 语言的接口

第 6 章　Matlab COM Builder 与 VC++

第 7 章　VC++调用 Matlab C++数学库

参考文献

第 1 章　Matlab 程序设计初步

1.1　Matlab 程序设计特点

Matlab 语言是一种解释型的高级语言，它包含自己的数据结构、程序流控制及文件输入输出等功能。Matlab 语句可以在 Matlab 控制窗口中直接执行，也可以采用脚本(script) *.m 文件和函数(function) *.m 文件的形式来实现。

1.1.1　Matlab Script 文件

采用 Matlab Script 文件，可以一次执行多条 Matlab 语句，这是一种比较简单的实现方式。由于 Matlab 的变量不需要定义，所以，可以很方便地按照程序的计算流程编写 Matlab Script 文件。如下面的 testscript.m 文件就是一个采用 Script 一次执行多条 Matlab 语句的实例。

```
x=-pi:0.01:pi;                          %pi 在 Matlab 中是常量，代表圆周率
y=sin(x);                               %生成绘制数据
ynoise=y+(rand(1,length(y))-0.5)*0.2;   %加均匀噪声
plot(x,y,'r.');
hold on;                                %重复绘制不擦除背景
plot(x,ynoise);
hold off;
```

上述 Script 文件的执行结果如图 1-1 所示。

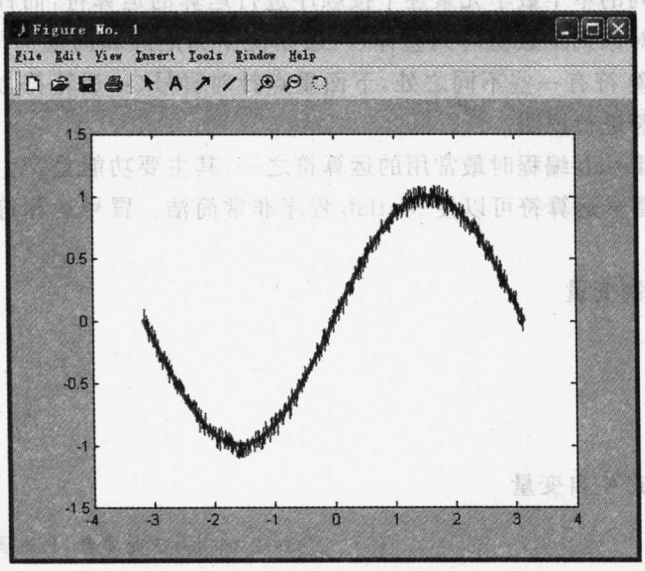


图 1-1　testscript.m 文件执行结果

1.1.2 Matlab 表达式

Matlab 的主要运算符如表 1-1 所列。

表 1-1 Matlab 的主要运算符

算术运算符		关系运算符		逻辑运算符	
+	数值阵列元素加	<	小于	&	按位与
-	数值阵列元素减	<=	小于等于	|	按位或
.*	数值阵列元素乘	>	大于	~	按位非
./	数值阵列元素右除	>=	大于等于	xor	异或
.\	数值阵列元素左除	==	相等	&&	逻辑与
:	冒号运算符	~=	不相等	||	逻辑或
.^	数值阵列元素幂				
.'	数值阵列元素转置				
'	数值阵列转置(对于实数与.'相同;对于复数,转置的同时对复数求共轭)				
*	数值阵列乘				
/	数值阵列右除				
\	数值阵列左除				
^	数值阵列幂				

Matlab 的算术运算符对于矩阵运算是非常方便的,可以大致将其分为针对数值阵列元素和针对数值阵列整体的两类数学运算符。其中针对数值阵列元素的数学运算符的运算方式可以理解为是数值阵列的单个数学元素逐个按顺序进行运算的运算符,而针对数值阵列整体的数学运算符的运算对象则是数值阵列整体(一般情况下为矩阵和向量)。Matlab 的运算符与普通高级语言的运算符有一些不同之处,下面重点针对冒号(:)运算符、点(.)运算符及左除(\)和右除(/)运算符进行说明。

":"运算符是 Matlab 编程时最常用的运算符之一,其主要功能是产生一个等间距的数据向量。恰当地使用冒号运算符可以使 Matlab 程序非常简洁。冒号运算符最常见的两个用途如下。

① 产生循环控制变量

```
for i=1:100
    %循环程序
end
```

② 生成数据时产生自变量

```
x=-pi:0.01:pi;                    %pi 在 Matlab 中是常量,代表圆周率
y=sin(x);
```

“.”运算符和其他运算符联合使用，则表示对阵列的元素进行操作。如 A. * B 表示 A 和 B 的相同位置元素进行相乘，A * B 则表示阵列 A 与 B 相乘(矩阵相乘或者向量相乘)。

“/”和“\”分别表示 Matlab 数值阵列右除和左除，其中 A/B 表示线性方程 AX＝B 的解，A\B 表示线性方程 XA＝B 的解；如果“/”或“\”与“.”配合使用，则表示数值阵列相同位置的元素分别进行右除或左除，其中 A.\B 表示 B(i)/A(i)，A./B 表示 A(i)/B(i)。

下面的一组 Matlab 语句则说明了 Matlab 算术运算符的使用方法。

```
%保存为 testoperat.m 文件
A=[1 2 3;4 5 6;7 8 9;];
B=[1 1 1;1 1 1;1 1 1;];

%矩阵元素相加、相减、相乘
C=A+B;
D=A-B;
E=A. * B;

%矩阵元素左除和右除
%其中 F 和 G 的结果相同
%均为：
%    1.0000    0.5000    0.3333
%    0.2500    0.2000    0.1667
%    0.1429    0.1250    0.1111
F=A.\B;
G=B./A;

%冒号运算符
N=1:100;

%幂运算符
A1=A.^2;

%转置
%Z1 为：
%       1.0000+1.0000i   4.0000+1.0000i   7.0000+1.0000i
%       2.0000+1.0000i   5.0000+1.0000i   8.0000+1.0000i
%       3.0000+1.0000i   6.0000+1.0000i   9.0000+1.0000i
%Z2 为：
%       1.0000-1.0000i   4.0000-1.0000i   7.0000-1.0000i
%       2.0000-1.0000i   5.0000-1.0000i   8.0000-1.0000i
%       3.0000-1.0000i   6.0000-1.0000i   9.0000-1.0000i
Z=A+B*i;                %构造复数
Z1=Z.';
Z2=Z';
```

```
A=[1 1 1;2 1 2;1 2 3;];
B=[1 1 1]';

%计算方程 AX=B 的解
%x+y+z=1
%2x+y+2z=1
%x+2y+3z=1
X=A\B;                    %X=[0.5000 1.0000 -0.5000]'

%计算方程 XA=B' 的解
%x+2y+z=1
%x+y+2z=1
%x+2y+3z=1
X=B'/A;                   %X=[1 0 0]
```

Matlab 的关系运算符和逻辑运算符与 C 语言的使用方法一样，只是 C 语言中表示非的“!”运算符在 Matlab 中用“～”来代替。另外，Matlab 的逻辑运算符可以直接对数值阵列进行操作，例如：

```
A=[1 1 0 0 1];
B=[0 1 0 0 1];
C=A&B;                    %C=[0 1 0 0 1]
D=A|B;                    %D=[1 1 0 0 1]
E=～A;                    %E=[0 0 1 1 0]
F=xor(A,B);               %F=[1 0 0 0 0]
```

1.1.3 Matlab 函数

除了采用 Script 文件编写 Matlab 程序以外，Matlab 语言也可以通过函数的形式编写 Matlab 程序。通过 Matlab 函数编写的 Matlab 程序便于维护，而且对于使用者而言，只需要了解函数的输入和输出即可，因此，要比 Script 文件容易使用得多。

Matlab 函数的定义方法如图 1-2 所示：

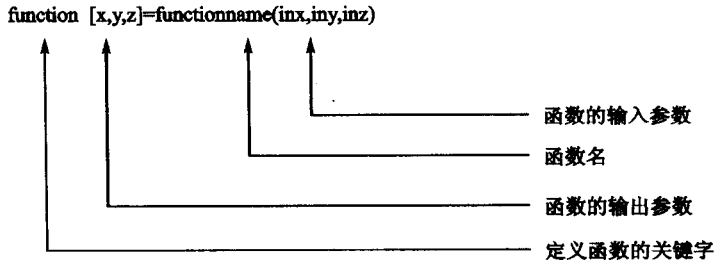


图 1-2 Matlab 函数的结构

函数体的写法与 Script 文件的写法类似，使用函数的时候，需要注意的是：第一，保存 Matlab 函数程序的时候必须采用函数名作为文件名，其扩展名为. m；第二，Matlab 函数输入的外部变量如果在函数体内发生变化的话，函数结束的时候，外部变量的值不会发生变化，类似于 C 语言输入参数的值传递。

如编写下面的函数，试图调换 x 和 y 的值，实际上这样做是不能实现的。

```
%保存为 change.m 文件
function [] = change(x,y)
t = x;
x = y;
y = t;
```

测试程序如下：

```
x = 10;
y = 1;
change(x,y);              %此时 x = 10,y = 1
```

另外，Matlab 有两个函数 nargin 和 nargout，分别返回 Matlab 函数的输入和输出参数的个数。如果采用元组变量 varargin 和 varargout 来替代常规的输入和输出参数，则函数就可以接受数目可变的输入和输出参数。下面的例子就是用来说明这两个函数的用法。

```
%保存为 testnarin_out.m 文件
function [varargout] = testnarin_out(varargin)
% function [x,y,z] = testnarin_out(m,n,k)
disp('输入参数的个数');
disp(nargin);
disp('输出参数的个数');
disp(nargout);
if(nargout>nargin)
    error('输出参数的个数不能大于输入参数的个数!\n');
end
varargout{1:nargout} = varargin{1:nargout};
```

测试程序如下：

```
m = rand(1,10);n = randn(5);z = magic(4);
testnarin_out(m,n);
x = testnarin_out(m,n);
[x,y,z] = testnarin_out(m,n);
```

则输出结果如下所示：

```
输入参数的个数
2
输出参数的个数
0
```

```
输入参数的个数
2
输出参数的个数
1
输入参数的个数
2
输出参数的个数
3
??? Error using==> testnarin_out
输出参数的个数不能大于输入参数的个数!\n
```

1.1.4 Matlab 的向量运算

Matlab 程序设计语言是解释型语言，其循环语句的效率非常低。但是，由于在算法上作了特殊的优化，Matlab 程序设计语言对于向量和矩阵运算的效率却很高。因而在进行 Matlab 程序设计的时候，一个很重要的原则就是尽量少地使用循环语句。可以做一个测试，同样是乘法，对于采用 for 循环和阵列元素相乘运算符“. *”，两者执行时间可以相差数十倍。

```
%保存为 testfor.m 文件
%清空 workspace 的变量
clear all;
clc;
%Matlab 循环语句与向量运算的测试语句
a=1:1000000;
b=1000000:-1:1;
sum_axb=0 ;
tic;                                %计时开始
for i=1:1000000
    sum_axb=sum_axb+a(i)*b(i);
end;
time1=toc;                          %计时结束并输出采用循环语句进行运算的时间

sum_axb=0;
tic;                                %计时开始
sum_axb=sum(a.*b);
time2=toc;                          %计时结束并输出采用向量运算消耗的时间

result1=strcat('采用循环语句运算消耗的时间为:',num2str(time1),'秒');
result2=strcat('采用向量运算消耗的时间为:    ',num2str(time2),'秒');

disp(result1);
disp(result2);
```

测试结果为：

```
采用循环语句运算消耗的时间为:0.718 秒
采用向量运算消耗的时间为:    0.031 秒
```

将 Matlab 的循环语句运算转换为向量运算需要在 Matlab 程序编写过程中加入一些技巧，其中最常见的形式如下。

① 用向量化运算代替 for 和 while 循环运算。如：

```
for i=-pi:0.1:pi
    y(i)=sin(i);
end
```

可以用下面的向量运算来代替。

```
x=-pi:0.1:pi;
y=sin(x);
```

另外，如果上例中还有两个向量之间的运算，可以用类似“.*”的数值阵列元素的运算函数来代替。即

```
for i=1:1000000
    sum_axb=sum_axb+a(i)*b(i);
end;
```

可以被

```
sum_axb=sum(a.*b);
```

来代替。

② 采用一些经过优化的 Matlab 向量运算函数。

经常用于向量运算的 Matlab 函数如表 1-2 所列。

表 1-2　Matlab 常用向量函数

函数名	函数功能
all	判断数值阵列是否全部非零
any	判断数值阵列是否有非零元素
reshape	变换数值阵列的各维元素数目
find	返回非零元素在阵列中的位置及其数值
sort	将数组按升序排序
sum	求数组的和
repmat	扩展阵列

除了 repmat 函数，表 1-2 中其他的函数都比较容易理解。下面就以 repmat 函数为例，说明如何采用 Matlab 的向量函数替换循环语句的技巧。repmat 函数用于扩展 Matlab 阵列，对于矩阵

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

它的扩展矩阵 B 为

$$B = \text{repmat}(A, 5, 3) = \begin{bmatrix} 1 & 2 & 1 & 2 & 1 & 2 \\ 3 & 4 & 3 & 4 & 3 & 4 \\ 1 & 2 & 1 & 2 & 1 & 2 \\ 3 & 4 & 3 & 4 & 3 & 4 \\ 1 & 2 & 1 & 2 & 1 & 2 \\ 3 & 4 & 3 & 4 & 3 & 4 \\ 1 & 2 & 1 & 2 & 1 & 2 \\ 3 & 4 & 3 & 4 & 3 & 4 \\ 1 & 2 & 1 & 2 & 1 & 2 \\ 3 & 4 & 3 & 4 & 3 & 4 \end{bmatrix}$$

下段程序运用了 repmat 函数绘制如图 1-3 所示的三维曲面图。

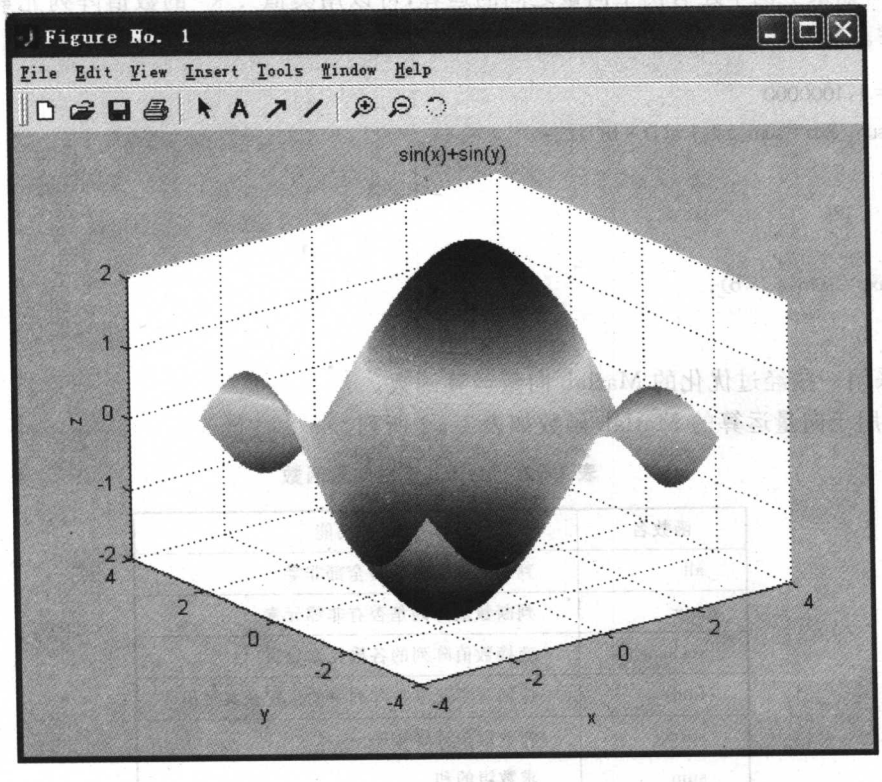


图 1-3 sin(x)+sin(y)的三维曲面图

```
%保存为 testrepmat.m 文件
%运用 repmat 函数
%绘制 sin(x)+sin(y)的三维曲面图
%首先清空 workspace 的变量
```

```
clear all;
clc;
x = -pi:0.01:pi;
y = -pi:0.01:pi;
y = y';
x1 = repmat(x,length(x),1);
y1 = repmat(y,1,length(y));
z = sin(x1) + sin(y1);
mesh(x1,y1,z);
xlabel('x');
ylabel('y');
zlabel('z');
title('sin(x) + sin(y)');
```

其实，采用向量运算代替循环语句的本质是用牺牲内存空间的做法来换取计算效率的提高。这是因为所有 Matlab 的内置函数都针对向量和矩阵运算进行了优化，因而对于 Matlab 的内置函数来说，只有当输入的数据是向量或者矩阵的形式时，其计算效率才是最高的。为了提高计算效率，就必须满足 Matlab 内置函数的这种要求，即在调用 Matlab 函数以前通过适当技巧构造好其所需要的向量或者矩阵，这便是 Matlab 程序设计向量化的本质所在。

1.1.5　Matlab 的程序控制

Matlab 用于程序控制的关键字如表 1 - 3 所列。

表 1 - 3　Matlab 常用的程序控制关键字

关键字	关键字的功能描述
if	与 else 和 elseif 关键字联合使用，根据设定的逻辑条件执行不同的代码
switch	与 else 和 otherwise 关键字联合使用，根据设定的变量值的不同执行不同的代码
while	根据设定的逻辑条件执行循环次数不定的循环模块
for	执行循环次数一定的循环模块
continue	适用于 for 或者 while 循环中，中止当前循环体语句的执行，直接进行下一次循环
break	退出当前层的循环体
return	直接返回到当前函数的调用函数中

下面分别举例说明 Matlab 程序控制关键字的使用方式。

1. if 和 switch 语句

if 和 switch 语句是 Matlab 程序设计中选择程序结构的两个关键字。其中 if 语句根据设定的逻辑条件执行不同的代码，而 switch 语句则根据设定的变量值的不同执行不同的代码。

if 可以单独使用，也可以与 else 和 elseif 联合使用，其使用方式有下面 3 种。

(1) 单独使用

if 逻辑表达式

```
        语句
    end
```

例如：

```
if A>B
   disp('A 大于 B');
end
```

(2) 和 else 联合使用

```
if 逻辑表达式
     语句 1
else
     语句 2
end
```

例如：

```
if A>B
    disp('A 大于 B');
else
    disp('B 大于等于 A');
end
```

(3) 和 elseif 联合使用

```
if 逻辑表达式
     语句 1
elseif
     语句 2
else
     语句 3
end
```

例如：

```
if A>B
    disp('A 大于 B');
else if A==B
    disp('A 等于 B');
else
    disp('B 大于 A');
end
```

而 switch 语句的使用方式如下。

```
switch 表达式
     case 数值 1
```