目录

第四章 循环结构

循环(loop)是一种 matlab 结构，它允许我们多次执行一系列的语句。循环结构有两种基本形式:while 循环和 for 循环。两者之间的最大不同在于代码的重复是如何控制的。在 while 循环中，代码的重复的次数是不能确定的，只要满足用户定义的条件，重复就进行下去。相对地，在 for 循环中，代码的重复次数是确定的，在循环开始之前，我们就知道代码重复的次数了。

4.1 while 循环

只要满足一定的条件，While 循环是一个重复次数不能确定的语句块。它的基本形如下

```
while expression
    ...
    ...   } code block
    ...
end
```

如果 expression 的值非零(true)，程序将执行代码块(code block)，然后返回到 while 语句执行。如果 expression 的值仍然非零，那么程序将会再次执行代码。直到 expression 的值变为 0，这个重复过程结束。当程序执行到 while 语句且 expression 的值为 0 之后，程序将会执行 end 后面的第一个语句。

while 循环的伪代码为

```
while expr
    ...
    ...
    ...
end
```

我们将用 whlie 循环编写一个统计分析的程序。

例 4.1

统计分析在科学工程计算中，跟大量的数据打交道是非常平常的事，这些数据中的每一个数据都是对我们关心的一些特殊值的度量。本课程的第一次测验的成绩就是一个简单的例子。每一个成绩都对某一个学生在本课程中学到多少东西的度量。

许多的时候，我们并不关心某一个单个数据。我们可以通过总结得到几个重要的数据，以此告诉我们数据的总体情况。例如，一组数据的平均数(数学期望)和标准差。平均数的定义如下:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad (4.1)$$

其中 x_i 代表 n 个样本中的第 i 个样本。如果所有的输入数据都可以在一个数组中得到，这些数据的平均数就可以通过公式(4.1)直接计算出来，或应用 matlab 的内建函数 mean。

标准差的定义如下:

$$s=\sqrt{\frac{N\sum_{i=1}^{N}x_i^2-(\sum_{i=1}^{N}x_i)^2}{N(N-1)}} \qquad (4.2)$$

标准差则体现随机变量取值与其期望值的偏差。标准差的值较大，则表明该随机变量的取值与其期望值的偏差较大，反之，则表明此偏差较小。如果所有的输入数据都可以在一个数组中得到，这些数据的平均数就可以通过公式(4.2)直接计算出来，或应用 matlab 的内建函数 std。本例的目的是要通过公式 4.1，4.2 计算平均数和标准差，向大家介绍 while 循环的应用。我们要执行的算法是读取一个组数据，计算它们的平均数和标准差，最后输出结果。

答案:

程序必须能读取一系列的测量值，并能够计算出这些测量值的数学期望和标准差。在进行计算之前，我们有 while 循环来读取这些测量值。

当所有的测量值输入完毕，我们必须通过一定的方法来告诉程序没有其它的数据输入了。在这里，我们假设所有测量值均为非负数，我们用一个负数来表示数据输入完毕。当一个负数输入时，程序将停止读取输入值，并开始计算这些数据的数学期望和方差。

1.陈述问题因为我们假设所有的输入数据为非负数，则合适地问题陈述为:计算一组测量数的平均数和方差，假设所有测量值为非负数;假设我们事先不知道有多少个测量数。一个负数的输入值将代表测量值输入的结束。

2.定义输入值和输出值这个程序的输入是未知数目的非负数。输出为这些非负数的平均数和标准差。顺便还要打印出输入数据的数据，以便于检测程序的正确性。

3.设计算法这个程序可分为以下三大步:

```
Accumulate the input data
Calculate the mean and standard deviation
Write out the mean, standard deviation, and number of points
```

每一大步的为读取输入值。为达此目的，我们必须提示用户输入合适的值。当数据输入完毕，我们将计算出数据的个数，它们的和与平方和。这些步骤的伪代码如下所示

```
Initialize n, sum_x, and sum_x2 to 0
Prompt user for first number
Read in first x
while x >=0
    n←n+1
    sum_x←sum_x + x
    sum_x2←sum_x2 + x^2
    Prompt user for next number
    Read in next x
end
```

注意我们必须在 while 循环开始之前，我们必须读取第一个值，这样在 while 循环第一次运行中才有了检测值。

下一步，我们要计算出数学期望和标准差。这个步骤的伪代码就是公式（4.1）和(4.2)的 matlab 版本。

```
x_bar ← sum_x / n
std_dev ← sqrt((n*num_x2 – sum_x^2) / (n*(n-1)))
```

最后我们写出输出结果:

```
Write out the mean value x_bar
Write out the standard deviation std_dev
Write out the number of input data points n
```

4.将伪代码转化为相应的 matlab 语句最终的 matlab 程序如下

```
% Script file: stats_1.m
%
% Purpose:
% To calculate mean and the standard deviation of
```

```
% an input data set containing an arbitrary number
% of input values.
%
% Record of revisions:
% Date Programmer Description of change
% ==== ========== =====================
% 12/05/97 S. J. Chapman Original code
%
% Define variables:
% n -- The number of input samples
% std_dev -- The standard deviation of the input samples
% sum_x -- The sum of the input values
% sum_x2 -- The sum of the squares of the input values
% x -- An input data value
% xbar -- The average of the input samples
% Initialize sums.
n = 0; sum_x = 0; sum_x2 = 0;
% Read in first value
x = input('Enter first value: ');
% While Loop to read input values.
while x >= 0
    % Accumulate sums.
    n = n + 1;
    sum_x = sum_x + x;
    sum_x2 = sum_x2 + x^2;
    % Read in next value
    x = input('Enter next value: ');
end
% Calculate the mean and standard deviation
x_bar = sum_x / n;
std_dev = sqrt( (n * sum_x2 - sum_x^2) / (n * (n-1)) );
% Tell user.
fprintf('The mean of this data set is: %f\n', x_bar);
fprintf('The standard deviation is: %f\n', std_dev);
fprintf('The number of data points is: %f\n', n);
```

5.检测程序为检测这个程序，我们将手工算出一组简单数据的平均数和标准差，然后与程序产生的结果进行比对。如果我们用三个输入值:3，4 和 5，那么它的平均数和标准差分别为

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i = \frac{1}{3}\times 12 = 4$$

$$s = \sqrt{\frac{N\sum_{i=1}^{N} x_i^2 - (\sum_{i=1}^{N} x_i)^2}{N(N-1)}} = 1$$

我们把这些值输入程序后，产生的结果为

```
>> stats_1
Enter first value: 3
Enter next value: 4
Enter next value: 5
Enter next value: -1
The mean of this data set is: 4.000000
The standard deviation is: 1.000000
The number of data points is: 3.000000
```

这个结果说明了程序的正确性。在这个例子中，我们并没有完全遵循设计过程。这个失误导致这个软件有一个致命的缺陷。你能指出来它吗?

我们的错误就在于我们没有检测程序所有可能的输入类型。请重看一遍程序。如果我们不输入数或者只输入一个数，那么上面的公式就会出现除以 0 的情况。这种除 0 错误将会在导致在命令窗口内出现 divide-by-zero 的警告，导致输出值为无穷大(NaN)。我们需要修改这个程序来解决这个问题，告诉用户这个问题是什么，并在适当的时侯停止。这个程序的修定版本为 stats_2。在运行运算之前，我们必须检查是否有足够多的输入值。如果没有，程序将打印出错误提示信息，并且跳出。你自己检测一下这个版本的程序。

```
% Script file: stats_2.m
%
% Purpose:
% To calculate mean and the standard deviation of
% an input data set containing an arbitrary number
% of input values.
%
% Record of revisions:
% Date Programmer Description of change
% ==== ========== ====================
% 12/05/97 S. J. Chapman Original code
% 1. 12/06/97 S. J. Chapman Correct divide-by-0 error if
% 0 or 1 input values given.
%
% Define variables:
% n -- The number of input samples
% std_dev -- The standard deviation of the input samples
% sum_x -- The sum of the input values
% sum_x2 -- The sum of the squares of the input values
% x -- An input data value
% xbar -- The average of the input samples
% Initialize sums.
n = 0; sum_x = 0; sum_x2 = 0;
% Read in first value
x = input('Enter first value: ');
% While Loop to read input values.
while x >= 0
    % Accumulate sums.
    n = n + 1;
    sum_x = sum_x + x;
    sum_x2 = sum_x2 + x^2;
    % Read in next value
    x = input('Enter next value: ');
end
% Check to see if we have enough input data.
if n < 2 % Insufficient information
    disp('At least 2 values must be entered!');
else % There is enough information, so
    % calculate the mean and standard deviation
    x_bar = sum_x / n;
    std_dev = sqrt( (n * sum_x2 - sum_x^2) / (n * (n-1)) );
    % Tell user.
    fprintf('The mean of this data set is: %f\n', x_bar);
    fprintf('The standard deviation is: %f\n', std_dev);
    fprintf('The number of data points is: %f\n', n);
end
```

注意平均数和标准差可以通过 MATLAB 的内建函数 mean 和 std 计算得到，输入数据存储在一个向量内，并把向量作为函数的参数。在本章章末的练习中，将会要求你用标准的 matlab 函数创建一个新的版本程序。

4.2 for 循环

for 循环结构是另一种循环结构，它以指定的数目重复地执行特定的语句块。For 循环的形式如下

```
for index = expr
     Statement 1
     ...          } Body
     Statement n
end
```

其中 index 是循环变量（就是我们所熟知的循环指数），exp 是循环控制表达式。变量 index 读取的是数组 expr 的行数，然后程序执行循环体（loopbody），所以 expr 有多少列，循环体就循环多少次。expr 经常用捷径表达式的]方式，即 first:incr:last。

在 for 和 end 之前的语句我们称之为循环体。在 for 循环运转的过程中，它将被重复的执行。For 循环结构函数如下：

1.在 for 循环开始之时，matlab 产生了控制表达式

2.第一次进入循环，程序把表达式的第一列赋值于循环变量 index，然后执行循环体内的语句。

3.在循环体的语句被执行后，程序把表达式的下一列赋值于循环变量 index，程序将再一次执行循环体语句。

4.只要在控制表达式中还有剩余的列，步骤 3 将会一遍一遍地重复执行。我们要举大量的例子来说明 for 循环的操作。

第一，考虑下面的例子

```
for ii = 1:10
    Statement 1
    ...
    Statement n
end
```

在这种情况下，控制表达式产生了一个 1 x 10 数组，所以语句 1 到 n 将会被重复执行 10 次。循环系数 ii 在第一次执行的时侯是 1，第二次执行的时侯为 2，依次类推，当最后一次执行时，循环指数为 10。在第十次执行循环体之后，再也没有新的列赋值给控制表达式，程序将会执行 end 语句后面的第一句。注意在循环体在最后一次执行后，循环系数将会一直为 10。

第二，考虑下面的例子。

```
for ii = 1:2:10
    Statement 1
    ...
    Statement n
end
```

在这种情况下，控制表达式产生了一个 1 x 5 数组，所以语句 1 到 n 将会执行 5 次。循环指数 ii 在第一次执行时为 1，第二次执行时为 3，依此类推，最后一次执行时为 9。在第五次执行循环体之后，再也没有新的列赋值给控制表达式，程序将会执行 end 语句后面的第一句。注意在循环体在最后一次执行后，循环系数将会一直为 9。

第三，考虑下面的例子。

```
for ii = [5 9 7]
    Statement 1
    ...
    Statementn
end
```

在这里，控制表达式是一个直接写出的 1 x 3 的数组，所以语句 1 到 n 将会执行 3 次，循环指数 ii 在第一次执行时为 1，第二次执行时为 3，第三次执行时为 7。循环指数在循环结束之后一直为 7。

最后，考虑下面的例子。

```
for ii = [1 2 3; 4 5 6]
```

```
    Statement 1
    ...
    Statement n
end
```

在这里，控制表达式是一个直接写出的 2 x 3 的数组，所以语句 1 到 n 将会执行 3 次，循环指数 ii 在第一次执行时为行向量$\begin{bmatrix}1\\4\end{bmatrix}$，第二次执行时为$\begin{bmatrix}4\\5\end{bmatrix}$，第三次执行时为$\begin{bmatrix}6\\7\end{bmatrix}$。这个例子说明循环指数可以为向量。

对应于 for 循环的伪代码为：

```
for index = expression
    Statement 1
    ...
    Statement n
end
```

例 4.2

阶乘（factorial）函数

为了说明 for 循环操作，我们将用 for 循环来计算阶乘函数。阶乘函数的定义如下：

```
N!=1                                    N=0
N!=N * (N-1) * (N-2) * ... * 3 * 2 * 1  N>0
```

计算 N 的阶乘的 matlab 代码为

```
n_factorial = 1
for ii = 1 : n
    n_factorial = n_factorial * ii;
end
```

假设我们要计算 5 的阶乘。如果 n 为 5，for 循环控制表达式将会产生行向量[1 2 3 4 5]。这种循环将会执行 5 次，ii 值按先后顺序依次为 1，2，3，4，5。n_factorial 最终的计算结果为 1 x 2 x 3 x 4 x 5=120。

例 4.3

计算 the day of year

the day of year 是指这一年已经逝去的天数（包括当天）。在平年中，它的取值范围为 1 到 365，在闰年中，它的取值范围 1 到 366。编写一个 matlab 程序，输入年，月，日，输入为对应的 the of year。

答案：

为了确定 the day of year，程序需要计算先前月份的天数之后，然后再计算当月已经过去了多少天，在求和的过程中将会用到 for 循环。因为每一个月的天数不尽相同，所以我们要确定每一个月的正确天数。我们用 switch 结构来确定它。

在闰年时，在二月后的某一天的 the day of year 将会比平年时大 1。因为在闰年的二月份多出一个 2 月 29 号。所以为了正确地计算出 the day of year，我们必须确定那一年是闰年。在公历中，闰年是这样规定的

1.能被 400 整除的年为闰年

2.能被 100 整除但不能被 400 整除的年不为闰年

3.能被 4 整除但不能被 100 整除年为闰年

4.其余的年份不为闰年

我们将用到 mod（求余）函数来确定一个数是否能被另一个数整除。如果函数的返回值为 0，则说一个数能被另一个数整除，否则，则不然。

下面是一个用于计算 the day of year 的程序。注意程序如何计算出前面月份总共的天数，如何应用 switch 结构确定每一月的天数。

```
% Script file: doy.m
```

```
%
% Purpose:
% This program calculates the day of year corresponding
% to a specified date. It illustrates the use switch
% and for constructs.
%
% Record of revisions:
% Date Programmer Description of change
% ==== ========== ====================
% 12/07/98 S. J. Chapman Original code
%
% Define variables:
% day                --Day (dd)
% day_of_year      --Day of year
% ii                 --Loop index
% leap_day          --Extra day for leap year
% month              --Month (mm)
% year               --Year(yyyy)
% Get day, month, and year to convert
disp('This program calculates the day of year given the ');
disp('current date.');
month = input('Enter current month (1-12):');
day = input('Enter current day(1-31):');
year = input('Enter current year(yyyy): ');
% Check for leap year, and add extra day if necessary
if mod(year,400) == 0
leap_day = 1; % Years divisible by 400 are leap years
elseif mod(year,100) == 0
leap_day = 0; % Other centuries are not leap years
elseif mod(year,4) == 0
leap_day = 1; % Otherwise every 4th year is a leap year
else
leap_day = 0; % Other years are not leap years
end
% Calculate day of year by adding current day to the
% days in previous months.
day_of_year = day;
for ii = 1:month - 1
% Add days in months from January to last month
switch (ii)
case {1,3,5,7,8,10,12},
day_of_year = day_of_year + 31;
case {4,6,9,11},
day_of_year = day_of_year + 30;
case 2,
day_of_year = day_of_year + 28 + leap_day;
end
end
% Tell user
fprintf('The date %2d/%2d/%4d is day of year %d.\n', ...
        month, day, year, day_of_year);
```

我们用下面已知的结果来检测这个程序。

1.1999 年不是闰年。它的 1 月 1 号对应的 day of year 是 1，12 月 31 号必定对应的是 365。

2.2000 年是一个闰年。它的 1 月 1 号对应的 day of year 是 1，12 月 31 号必定对应的是 366。

3.2001 年不是闰年。它的 1 月 1 号对应的 day of year 是 30。这个程序 5 次运行后的结果分别为

```
>> doy
This program calculates the day of year given the
current date.
Enter current month (1-12):1
Enter current day(1-31):1
Enter current year(yyyy): 1999

The date    1/ 1/1999 is day of year 1.

>> doy
This program calculates the day of year given the
current date.
Enter current month (1-12):12
Enter current day(1-31):31
Enter current year(yyyy): 1999
The date 12/31/1999 is day of year 365.

>> doy
This program calculates the day of year given the
current date.
Enter current month (1-12):1
Enter current day(1-31):1
Enter current year(yyyy): 2000

The date    1/ 1/2000 is day of year 1.

>> doy
This program calculates the day of year given the
current date.
Enter current month (1-12):12
Enter current day(1-31):31
Enter current year(yyyy): 2000

The date 12/31/2000 is day of year 366.

>> doy
This program calculates the day of year given the
current date.
Enter current month (1-12):3
Enter current day(1-31):1
Enter current year(yyyy): 2001

The date    3/ 1/2001 is day of year 60.
```

通过 5 次不同情况的检测，这个程序给出了正确的结果。

例 4.4

统计分析

执行如下算法：

输入一系列的测量数，计算它们的平均数和标准差。这些数可以是正数，负数或 0。

答案：

这个程序必须能够读取大量数据，并能够计算出这些测量值的平均数和标准差。这些测量值可以是正数，负数或 0。

因为我们再也不能用一个数来表示数据中止的标识了，我们要求用户给出输入值的个

数，然后用 for 循环读取所有数值。

下面的就是这个修定版本的程序。它允许各种输入值，请你自己验证下面 5 个输入值的平均数和标准差：3，-1，0，1，-2。

```
% Script file: stats_3.m
%
% Purpose:
% To calculate mean and the standard deviation of
% an input data set, where each input value can be
% positive, negative, or zero.
%
% Record of revisions:
% Date Programmer Description of change
% ==== ========== ====================
% 12/08/97 S. J. Chapman Original code
%
% Define variables:
% ii Loop index
% n The number of input samples
% std_dev The standard deviation of the input samples
% sum_x The sum of the input values
% sum_x2 The sum of the squares of the input values
% x An input data value
% xbar The average of the input samples
% Initialize sums.
sum_x = 0; sum_x2 = 0;
% Get the number of points to input.
n = input('Enter number of points: ');
% Check to see if we have enough input data.
if n < 2 % Insufficient data
    disp ('At least 2 values must be entered.');
else % we will have enough data, so let's get it.
    % Loop to read input values.
    for ii = 1:n
        % Read in next value
        x = input('Enter value: ');
        % Accumulate sums.
        sum_x = sum_x + x;
        sum_x2 = sum_x2 + x^2;
    end
    % Now calculate statistics.
    x_bar = sum_x / n;
    std_dev = sqrt((n * sum_x2 - sum_x^2) / (n * (n - 1)));
    % Tell user.
    fprintf('The mean of this data set is: %f\n', x_bar);
    fprintf('The standard deviation is: %f\n', std_dev);
    fprintf('The number of data points is: %f\n', n);
end
```

4.2.1 运算的细节

既然我们已经看了许多 for 循环的例子。在用 for 循环时，我们必须检查许多重要的细节。

1.没有必要缩进 for 循环的循环体。即使所有语句都左对齐，matlab 程序也会识别出这个循环。但缩进循环体能增强代码的可读性，所以建议大家缩进循环体。

好的编程习惯

对于 for 循环体总是要缩进两个或更多空格，以增强程序的可读性。

2.在 for 循环中，我们不能随意修改循环指数。循环指数常被用作计算器，
如果修改了它们将会导致一些奇怪而难以发现的错误。下面的一个例子将初始化一个函数的数组。但是语句“ii=5”的突然出现，导致只有 a(5)得到了初始化，它得到了本应赋给 a(1)，a(2)等等的值。

```
for ii = 1:10
    ...
    ii = 5 ; %Error!
    ...
    a(ii) = <calculation>
end
```

好的编程习惯
在循环体中绝不修改循环指数的值。

3.我们在第二章已经学过，用赋值的方法可以扩展一个已知的数组。例如，语句

```
arr = 1:4;
```

定义了一个数组[1 2 3 4]。如果执行语句

```
arr(8) = 6;
```

将会产生一个八元素数组[1 2 3 4 0 0 0 6]。不幸的是，每一次扩展数组，都要经过以下步骤:第一步，创建一个新数组。第二步，把旧数组的元素复制到新数组当中。第三步，把扩展的元素写入新数组。第四步，删除旧数组。对于大数组来说这些步骤是相当耗时的。
当一个 for 循环中存储了一个预先未定义的数组，在第一次循环执行的时侯，循环结构迫使 matlab 重复执行以上步骤。从另一方面说，如果在循环开始之前数组预先分配了数组的大小，那么复制就不需要了，执行代码的速度也将加快。下面代码片段向大家展示了在循环开始之前如何预先分配数组。

好的编程习惯
在循环执行开始之前，总是要预先分配一个数组，这样能大大增加循环运行的速度。

4.用 for 循环和向量计算是非常常见的。例如，下面的代码片段用 for 循环计算 1 到 100 之间的所有整数的平方，平方根，立方根。

```
for ii = 1:100
    square(ii) = ii ^2;
    square_root(ii) = ii ^ (1/2);
    cube_root(ii) = ii ^ (1/3);
end
```

下面一个代码片段是用向量计算上面的问题。

```
    ii = 1:100;
    square = ii .^2;
    square_root = ii .^ (1/2);
    cube_root(ii) = ii .^ (1/3);
```

尽管两种算法得到了相同的结果，但两者并不同等价。因为 for 循环算法比向量算法慢 15 倍还多。这是由于 matlab 通过每一次循环时，每行都要翻译执行一次。也相当于 matlab 翻译执行了 300 行代码。相反，如果用向量算法，matlab 只需要翻译执行 4 行代码。所以用向量语句它的执行速度非常快。
向量算法的缺点是需要很大的内存，因为一些间接的数组需要创建。这经常是一小点损失，所以要比 for 循环算法好的多。
在 matlab 中，用向量算法代替循环的算法的过程称之为向量化(vectorization)。向量化能够改进许多的 matlab 程序。

好的编程习惯

那种既可以用向量可以解决的问题，也可以用循环解决的问题，最好用向量解决，这是因为向量执行的速度快。

例 4.5

比较向量算法和循环为了比较循环和向量算法执行若无事所用的时间，用两种方法编程并测试三个运算所花的时间。

1.用 for 循环计算 1 到 10000 的之间每一个整数的平方，而事先不初始化平方数组。

2.用 for 循环计算 1 到 10000 的之间每一个整数的平方，而事先初始化平方数组。

3.用向量算法计算计算 1 到 10000 的之间每一个整数的平方。

答案:

这个程序必须用上面提供的三种方示计算出 1 到 10000 之间的每一个整数的平方，并测试每一个种算法的时间。测试时间要用到 matlab 函数 tic 和 toc。tic 函数复位内建计时器，而 toc 函数则从最后一次调用 tic 以秒开始计时。

因为在许多的计算机中它的时间钟是相当粗略的，所以有必要多运行几次以获得相应的平均数。

下面就是用三种方法编出的 matlab 程序。

```
% Script file: timings.m
%
% Purpose:
% This program calculates the time required to
% calculate the squares of all integers from 1 to
% 10,000 in three different ways:
% 1. Using a for loop with an uninitialized output
% array.
% 2. Using a for loop with an preallocated output
% array.
% 3. Using vectors.
%
% Record of revisions:
% Date Programmer Description of change
% ==== ========== ====================
% 12/08/97 S. J. Chapman Original code
%
% Define variables:
% ii, jj Loop index
% average1 Average time for calculation 1
% average2 Average time for calculation 2
% average3 Average time for calculation 3
% maxcount Number of times to loop calculation
% square Array of squares
% leap_day Extra day for leap year
% month Month(mm)
% year Year(yyyy)
% Perform calculation with an uninitialized array
% "square". This calculation is done only once
% because it is so slow.
maxcount = 1; % One repetition
tic; % Start timer
for jj = 1:maxcount
    clear square % Clear output array
    for ii = 1:10000
        square(ii) = ii^2; % Calculate square
    end
```

```
end
average1 = (toc)/maxcount; % Calculate average time
% Perform calculation with a preallocated array
% "square". This calculation is averaged over 10
% loops.
maxcount = 10; % One repetition
tic; % Start timer
for jj = 1:maxcount
    clear square % Clear output array
    square = zeros(1,10000); % Preinitialize array
    for ii = 1:10000
        square(ii) = ii^2; % Calculate square
    end
end
average2 = (toc)/maxcount; % Calculate average time
% Perform calculation with vectors. This calculation
% averaged over 100 executions.
maxcount = 100; % One repetition
tic; % Start timer
for jj = 1:maxcount
    clear square % Clear output array
    ii = 1:10000; % Set up vector
    square = ii.^2; % Calculate square
end
average3 = (toc)/maxcount; % Calculate average time
% Display results
fprintf('Loop / uninitialized array = %8.4f\n', average1);
fprintf('Loop / initialized array = %8.4f\n', average2);
fprintf('Vectorized = %8.4f\n', average3);
```

4.2.2 break 和 continue 语句

有两个附加语句可以控制 while 和 for 循环:break 和 continue 语句。break 语句可以中止循环的执行和跳到 end 后面的第一句执行，而 continue 只中止本次循环，然后返回循环的顶部。如果 break 语句在循环体中执行，那么体的执行中止，然后执行循环后的第一个可执行性语句。用在 for 循环中的 break 语句的例子如下:

程序执行的结果为:

```
%test_break.m
for ii = 1:5;
    if ii == 3;
        break;
    end
    fprintf('ii = %d \n', ii);
end
disp('End of loop!');
>> test_break
ii = 1
ii = 2
End of loop!
```

注意 break 语句在 ii 为 3 时执行，然后执行 `disp('End of loop!');`语句而不执行 `fprintf('ii = %d \n', ii);`语句。

continue 语句只中止本次循环，然后返回循环的顶部。在 for 循环中的控制变量将会更

新到下一个值，循环将会继续进行。下面是一个在 for 循环中的 continue 的例子。

```
%test_continue.m
for ii = 1:5;
    if ii == 3;
        continue;
    end
    fprintf('ii = %d \n', ii);
end
disp('End of loop!');
```

程序运行的结果为;

```
>> test_continue
ii = 1
ii = 2
ii = 4
ii = 5
End of loop!
```

注意 continue 语句在 ii 为 3 时执行，然后程序返回循环的顶部而不执行 fprintf 语句。break 和 continue 语句可用在 while 循环和 for 循环中。

4.2.3 循环嵌套

一个循环完全出现在另一个循环当中，这种情况经常发生。如果一个循环完全出现在另一个循环当中，我们称这两个循环为**带嵌套的循环**。下面的例子用两重 for 循环嵌套来计算并写出结果。

```
for ii = 1:3
    for jj = 1:3
        product = ii * jj;
        fprintf('%d * %d = %d \n',ii,jj,product);
    end
end
```

在这个例子中，外部的 for 循环将把 1 赋值于循环指数 ii，然后执行内部 for 循环。内部循环的循环体将被执行 3 次，它的循环指数 ii 将会先后被赋值为 1，2，3。当完全执行完内部的循环后，外部的 for 循环将会把 2 赋值于循环指数 ii，然后内部的 for 循环将会再次执行。直到外部 for 循环执行 3 次，这个重复过程结束。产生的结果为

```
1 * 1 = 1
1 * 2 = 2
1 * 3 = 3
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
```

注意外部 for 循环指数变量增加之前，内部 for 循环要完全执行完。

当 matlab 遇到一个 end 语句，它将与最内部的开放结构联合。所以第一个 end 语句与语句“for jj = 1:3”，第二个 end 语句与语句“for ii = 1:3”联合。如果在循环嵌套中一个 end 语句突然被删除，将会产生许多难以发现的错误。

如果 for 循环是嵌套的，那么它们必须含有独立的循环变量。如果它们含有相同的循环变量，那么内部循环将改变外部循环指数的值。

如果 break 或 continue 语句出现在循环嵌套的内部，那么 break 语句将会在包含它的最内部的循环起作用。

```
for ii = 1:3
    for jj = 1:3
        if jj ==3;
```

```
                break;
            end
            product = ii * jj;
            fprintf('%d * %d = %d \n',ii,jj,product);
        end
        fprintf('End of inner loop\n');
    end
    fprintf('End of outer loop\n');
```

如果内部循环指数 jj 为 3，那么 break 语句开始执行，这将导致程序跳出内部循环。程序将会打印出"End of inner loop"，外部循环指数将会增加 1，内部循环的执行重新开始。产生的输出值为:

```
1 * 1 = 1
1 * 2 = 2
End of inner loop
2 * 1 = 2
2 * 2 = 4
End of inner loop
3 * 1 = 3
3 * 2 = 6
End of inner loop
End of outer loop
```

4.3 逻辑数组与向量化

在第二章中，我们提出 matlab 有两个基本类型的数据类型:数字型与字符型。数字型数据包括数字，字符型数据包含字符。除这两个数据类型之外，还有第三类数据类:逻辑型。

"逻辑"数据类型在 matlab 中并不真实存在。其实，它是带特定逻辑属性标准数字型数据类型。逻辑型数组通过所有的关系运算符和逻辑运算符创建。它们区别于数字型的是在调用 whos 命令时，(logical)会出现在类型的后面。

例如，考虑下面的语句

```
a = [1 2 3; 4 5 6; 7 8 9];
b = a > 5;
```

这些语句将会产生两个数组 a 和 b。a 将会产生一个数组 $\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}$，b 将会产生一个特殊的含有逻辑属性 $\begin{bmatrix}0&0&0\\0&0&1\\1&1&1\end{bmatrix}$。当调用 whos 命令时，结果如下。注意 b 后面的(logical)修饰符。

```
>> whos
  Name        Size                    Bytes  Class

  a           3x3                        72  double array
  b           3x3                         9  logical array

Grand total is 18 elements using 81 bytes
```

我们还可以用 logical 函数给一个数组加上一个逻辑属性。例如，语句 c=logical(a)，将会把 a 值赋于 c，从而使 c 带有一定的逻辑性:

一个数组的逻辑属性可以通任何的数学运算去除。例如，如果我们在 c 数组加 0，数组的值不会改变，而它的逻辑属性将会消失

```
>> c=b+0

c =

     0     0     0
     0     0     1
     1     1     1

>> whos
  Name         Size                     Bytes  Class

  a            3x3                         72  double array
  b            3x3                          9  logical array
  c            3x3                         72  double array

Grand total is 27 elements using 153 bytes
```

4.3.1 逻辑数组的重要性

逻辑数组有一个重要的属性——它在算术运算中能提供一个屏蔽(mask)。屏蔽(mask)是指一个数组，它从另一个数组选择所需的元素参与运算。指定的运算只在选择的元素上执行，而不执行原有的元素。

例如，假设数组 a 和 b 的定义如上节所示。那么语句 a(b)=sqrt(a(b))会计算 a 中相应的元素的平方根，相应的元素是指与 b 数组中的非零元素相对应的数组 a 中的元素。其他元素保持不变。

```
>> a(b)=sqrt(a(b))

a =

    1.0000    2.0000    3.0000
    4.0000    5.0000    2.4495
    2.6458    2.8284    3.0000
```

对于一个数组的子集快速而简单，而不用循环和选择结构。

下面的语句，是用循环结构和选择结构计算上述问题。

```
for ii = 1:size(a,1)
    for jj = 1:size(a,2)
        if a(ii,jj) > 5
            a(ii,jj)=sqrt(a(ii,jj));
        end
    end
end

b = a > 5;
a(b) = sqrt(a(b));
```

例 4.6

用逻辑数数组进行屏蔽运算为了比较循环结构，选择结构与应用逻辑数组运算的快慢，我们进行下面两个计算，并对它进行计时。

1.创建一个含 10000 个元素的数组，其值依次为 1 到 10000 之间的整数。用 for 循环和 if 结构计算大于 5000 的元素的平方根。

2.创建一个含 10000 个元素的数组，其值依次为 1 到 10000 之间的整数。用逻辑数组计算大于 5000 的元素的平方根。

答案:

这个程序必须创建一个含 10000 个元素的数组，其值依次为 1 到 10000 之间的整数。

用两种不同的方法计算出大于 5000 的元素的平方根。

比较两种方法运行速度的 matlab 程序如下所示:

```
% Script file:logical1.m
%
% Purpose:
%     This program calculates the time required to
%     calculate the square roots of all elements in
%     array a whose value exceeds 5000. This is done
%     in two differents ways:
%     1.Using a for loop and if construct.
%     2.Using a logical array.
%
% Record of revisions:
%     Date                Programmer              Description of change
%     ====              ================      ==============================
% 06/01/02          S.   J.   Chapman             Original code
%
% Define variables:
%     a                          --Array of input values
%     b                          --Logical array to serve as a mask
%     ii,jj                    --Loop index
%     average1                 --Average time for calculation 1
%     average2                 --Average time for calculation 2
%     maxcount                  --Number of times to loop calculation
%     month                     --Month (mm)
%     year                     --Year (yyyy)
%
% Perform calculation using loops and branches
maxcount = 1;                  % One repetition
tic;                           % Start timer
for jj = 1:maxcount
      a = 1:10000;             %Declare array a
      for ii = 1:10000
            if a(ii) > 5000
                  a(ii) = sqrt(a(ii));
            end
      end
end
average1 = (toc)/maxcount;%Calculate average time
%
% Perform calculation using logical arrays.
maxcount = 10;           %One repetition
tic;                     %Start timer
for jj = 1:maxcount
      a = 1:10000;       %Declare array a
      b = a > 5000;      %Create mask
      a(b) = sqrt(a(b));    %Take square root
end
average2 = (toc)/maxcount;    %Calculate average time
%
% Display result
fprintf('Loop /if approach = %8.4f\n',average1);
fprintf('Logical array approach = %8.4f\n',average2);
```

这个程序在 cpu 为奔腾 III(主频为 733mhz)的计算机运行得到结果如下

```
>> logical1
Loop /if approach =      0.1200
```

```
Logical array approach =      0.0060
```

正如我们看到的，用逻辑数组方法速度是另一种方法的 20 倍。

好的编程习惯

如果用可能的话，可用逻辑函数选择数组中的元素。如果逻辑数组进行运算，要比循环快得多。

4.3.2 用 if/else 结构和逻辑数组创建等式

逻辑数组经常被用来替代 for 循环中的 if/else 结构。正如我们在上节所看到的，把逻辑运算当作一个屏蔽来选择数组中的某些元素进行运算。如果你要利用那些没有被选择到的元素进行运算，只需要在逻辑屏蔽上加一个非运算符(-)。例如，假设我们要计算一个二维数组中所有的大于 5 的元素的平方根，然后其余的数的平方。利用循环和选择结构的代码如下:

```
for ii = 1:size(a,1)
      for jj = 1:size(a,2)
            if a(ii,jj) > 5
                  a(ii,jj) = sqrt(a(ii,jj));
            else
                  a(ii,jj) = a(ii,jj)^2;
            end
      end
end
```

用逻辑数组运算的代码如下:

```
b = a > 5
a(b) = sqrt(a(b));
a(~b) = a(~b) .^2;
```

显然用逻辑数组的方法运算速度要快得多。

测试 4.1

本测试提供了一个快速的检查方式，看你是否掌握了 4.1 到 4.3 的基本内容。如果你对本测试有疑问，你可以重读 4.1 到 4.3，问你的老师，或和同学们一起讨论。在附录 B 中可以找到本测试的答案。

检测下列 for 循环，说出它们的循环次数:

```
1. for index = 7:10
2. for jj = 7:-1:10
3. for index = 1:10:10
4. for ii = -10:3:-7
5. for kk = [0 5; 3 3]
```

检测下列循环，确定循环指数 ires 的最终值。

```
6.   ires = 0;
      for index = 1:10;
            ires = ires + 1;
      end

7.   ires = 0;
      for index = 1:10;
            ires = ires + index;
      end

8.   ires = 0;
      for index1 = 1:10;
            for index2 = index1:10
```

```
                if index2 == 6
                    break;
                end
                ires = ires + 1;
            end
        end

9.  ires = 0;
    for index1 = 1:10;
        for index2 = index1:10
            if index2 == 6
                continue;
            end
            ires = ires + 1;
        end
    end
```

10. 编写一个 matlab 语句，计算下列的函数值

定义域为-6π<t<6π，每隔 π 取一次值。用两种方法进行运算，一次用循环和选择语句，

$$f(t)=\begin{cases}\sin t & \text{for all t where sint} > 0\\ 0 & \text{elsewhrer}\end{cases}$$

另一次用逻辑数组。

4.4 附加例子

例 4.7

用最小二乘法画噪声数据的近似曲线

下落物体将会作匀加速度运动，它的速度符合下面的公式

$$v(t) = at + v_0 \qquad (4.3)$$

*v(t)*代表物体在 *t* 时刻的速度。加速度为 *g*，初速度 v_0 为 *0*。这个公式出现在基础物理学中，我们大家都非常的熟悉。如果我们要画出下落物体的速度时间图象，我们得到的(*v*，*t*)测量值应当在同一条直线上。但是，学习物理的同学都知道，在实验室得到的测量值不一定是直线。为什么会这样呢?因为所有的测量都有误差。在所有测量值中都有一定的噪声。

在工程和科研方面，有许多像这个例子一样带有噪声，而我们希望得到最符合的结果。这个问题叫做线性待定问题。给出一系列带噪声的测量值(*x*，*y*)，它遵循一条直线，如何确定“最符合”这条直线的解析式呢。

如果我们确定了待定系数 *m* 和 *b*，那么我们就确定了解析式 4.4。

$$y=mx+b \qquad (4.4)$$

确定待定系数 *m* 和 *b* 的标准方法为最小二乘法。之所以称为最小二乘法，是因为根据偏差的平方和为最小的条件来选择常数 *m* 和 *b* 的。公式如下:

$$m=\frac{(\sum xy)-(\sum x)\bar{y}}{(\sum x^2)-(\sum x)\bar{x}} \qquad (4.5)$$

$$b=\bar{y}-m\bar{x} \qquad (4.6)$$

其中，$\sum x$ 代表所有测量值 *x* 之和，$\sum y$ 代表所有测量值 *y* 之和，$\sum xy$ 代表所有对应的 *x* 与 *y* 的乘积之和，$\bar{x}$ 代表测值量 *x* 的数学期望。$\bar{y}$ 代表测值量 *y* 的数学期望。已知有一系列含有噪声的数据(*x*，*y*)，编写程序用最小二乘法计算出 *m* 和 *b*。数据要求从键盘输入，画出每一个数据点还有画出最适合的直线。

答案:

1. 陈述问题

已知有一系列含有噪声的数据(x，y)用最小二乘法计算 m 和 b。数据要求从键盘输入，画出每一个数据点还有画出最适合的直线。

2. 定义输入输出值

这个程序所需的输入值为点的个数，以及点的坐标。输出是用最小二乘法得到的斜率以及 y 上的截距。

3. 设计算法

这个问题被分解为 6 个大步骤:

```
Get the number of input data points
Read the input data values
Calculate the required statistics
Calculate the slop and intercept
Write out the slop and intercept
Plot the input points and the fitted line
```

第一大步是读取输入量的个数，所以我们要用到 input 函数。下一步我们要在 for 循环中使用 input 函数读取输入量(x，y)。每一个输入值将会产生一个数组([x，y])，然后这个函数将会返回这个数组到调用程序。注意在这里应用 for 循环是合适的，因为我们事先知道循环要执行多少次。

上述步骤的伪代码如下:

```
Print message describing purpose of the program
n_points ← input('Enter number of [x y] pairs:');
for ii = 1:n_points
    temp ← ('Enter [x y] pairs:');
    x(ii) ← temp(1);
    y(ii) ← temp(2);
end
```

下一步，我们必须计算出相关的统计量。这些统计量是$\sum x$，$\sum y$，$\sum xy$，$\sum x^2$。伪代码如下

```
Clear the vairables sum_x, sum_y, sum_x2, and sum_xy
for ii = 1:n_points
    sum_x ← sum_x + x(ii)
    sum_y ← sum_y + y(ii)
    sum_x2 ← sum_x2 + x(ii)^2
    sum_xy ← sum_xy + x(ii) * y(ii)
end
```

下一步我们必须计算出斜率 m 和 y 轴上的载距 b。这一步的伪代码就是公式(4.4)和(4.5)。

```
x_bar ← sum_x / n_points
y_bar ← sum_y / n_points
slope ← (sum_xy-sum_x * y_bar)/( sum_x2 –sum_x * x_bar)
y_int ← y_bar – slope * x_bar
```

最后，我们必须写出和画出相应的结果。输入的坐标点要用圆点画出，不用连接线而用最小二乘法得到解析式对应的直线用 2pixel 的实直线画出。在此之前我们要用到 hold on 命令。画完直线之后，调用 hold off 命令。我们在图象中将会添加相应的标题，以及相应的图例。

4. 转化为 matlab 语句

```
% Script file: lsqfit.m
% Purpose:
% To perform a leastsquares fit of an input data set
% to a straight line, and print out the resulting slope
% and intercept values. The input data for this fit
% comes from a userspecified input data file.
%
```

```
% Record of revisions:
% Date Programmer Description of change
% ==== ========== ====================
% 01/03/99 S. J. Chapman Original code
%
% Define variables:
% ii Loop index
% n_points Number in input [x y] points
% slope Slope of the line
% sum_x Sum of all input x values
% sum_x2 Sum of all input x values squared
% sum_xy Sum of all input x*y yalues
% sum_y Sum of all input y values
% temp Variable to read user input
% x Array of x values
% x_bar Average x value
% y Array of y values
% y_bar Average y value
% y_int yaxis intercept of the line
disp('This program performs a leastsquares fit of an ');
disp('input data set to a straight line.');
n_points = input('Enter the number of input [x y] points: ');
% Read the input data
for ii = 1:n_points
temp = input('Enter [x y] pair: ');
x(ii) = temp(1);
y(ii) = temp(2);
end
% Accumulate statistics
sum_x = 0;
sum_y = 0;
sum_x2 = 0;
sum_xy = 0;
for ii = 1:n_points
    sum_x = sum_x + x(ii);
    sum_y = sum_y + y(ii);
    sum_x2 = sum_x2 + x(ii)^2;
    sum_xy = sum_xy + x(ii) * y(ii);
end
% Now calculate the slope and intercept.
x_bar = sum_x / n_points;
y_bar = sum_y / n_points;
slope = (sum_xy - sum_x * y_bar) / ( sum_x2 - sum_x * x_bar);
y_int = y_bar - slope * x_bar;
% Tell user.
disp('Regression coefficients for the leastsquares line:');
fprintf(' Slope (m) = %8.3f\n', slope);
fprintf(' Intercept (b) = %8.3f\n', y_int);
fprintf(' No of points = %8d\n', n_points);
% Plot the data points as blue circles with no
% connecting lines.
plot(x,y,'bo');
hold on;
% Create the fitted line
xmin = min(x);
xmax = max(x);
ymin = slope * xmin + y_int;
```

```
ymax = slope * xmax + y_int;
% Plot a solid red line with no markers
plot([xmin xmax],[ymin ymax],'r','LineWidth',2);
hold off;
% Add a title and legend
title ('\bfLeastSquaresFit');
xlabel('\bf\itx');
ylabel('\bf\ity');
legend('Input data','Fitted line');
grid on
```

5. 检测程序为了检测这个程序，我们将采用一些简单的数据，如果输入数据所对应的点都在同一条

直线，那么产生的斜率和截距必定是那条直线的斜率和截距。这组数据为

```
 [1.1 1.1]
 [2.2 2.2]
 [3.3 3.3]
 [4.4 4.4]
 [5.5 5.5]
 [6.6 6.6]
 [7.7 7.7]
```

它的斜率和截距分别为 1.0 和 0.0。我们将用这些值来运行这个程序，结果如下:

```
>> lsqfit
This program performs a leastsquares fit of an
input data set to a straight line.
Enter the number of input [x y] points: 7
Enter [x y] pair: [1.1 1.1]
Enter [x y] pair: [2.2 2.2]
Enter [x y] pair: [3.3 3.3]
Enter [x y] pair: [4.4 4.4]
Enter [x y] pair: [5.5 5.5]
Enter [x y] pair: [6.6 6.6]
Enter [x y] pair: [7.7 7.7]
Regression coefficients for the leastsquares line:
  Slope (m) =       1.000
  Intercept (b) =       0.000
  No of points =              7
```

图象如图 4.1(a)

我们将在这些值上加入一些噪声，如下所示

```
 [1.1 1.01]
 [2.2 2.30]
 [3.3 3.05]
 [4.4 4.28]
 [5.5 5.75]
 [6.6 6.48]
 [7.7 7.84]
```

再次运行程序，所得的结果如下:

```
>> lsqfit
This program performs a leastsquares fit of an
input data set to a straight line.
Enter the number of input [x y] points: 7
Enter [x y] pair: [1.1 1.01]
Enter [x y] pair: [2.2 2.30]
Enter [x y] pair: [3.3 3.05]
Enter [x y] pair: [4.4 4.28]
Enter [x y] pair: [5.5 5.75]
```

```
Enter [x y] pair: [6.6 6.48]
Enter [x y] pair: [7.7 7.84]
Regression coefficients for the leastsquares line:
  Slope (m) =        1.024
  Intercept (b) =     -0.120
  No of points =              7
```

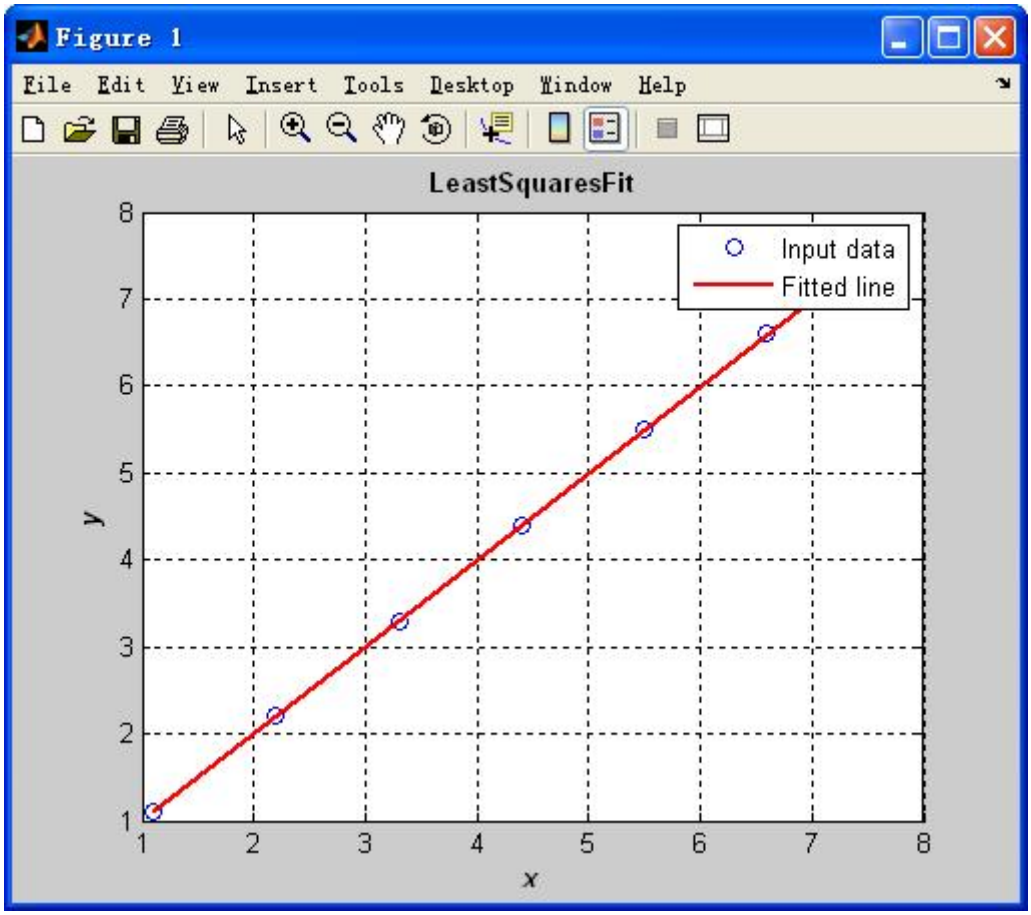

图 4.1(a)

我们用手动计算很容易就能得到上面两个程序的正确结果。第二个程序图象如图 4.2(b)所示。

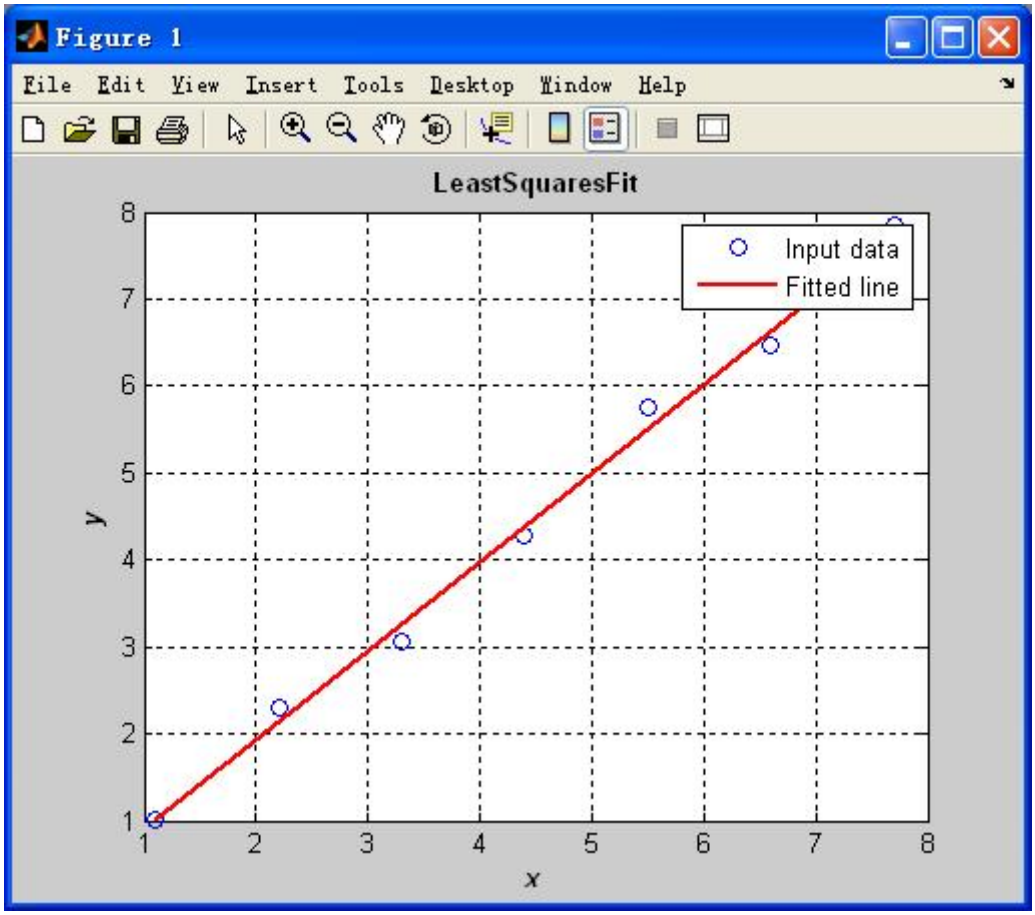

图 4.2(b)

这个例子用到了第三章中许多画图的例子。用 hold 命令在同一坐标下，画出了多个图象。用 LineWidth 属性来改变直线的宽度。用转义序列来设标题字体。

例 4.8

小球的轨迹如果我们假设处于真空中，且忽略地球的曲率。在地球任意一点向空中抛出一小球将会产生类似于图 4.2（a）所示的曲线。球在时刻 t 的高度将会遵守公式(4.7)。

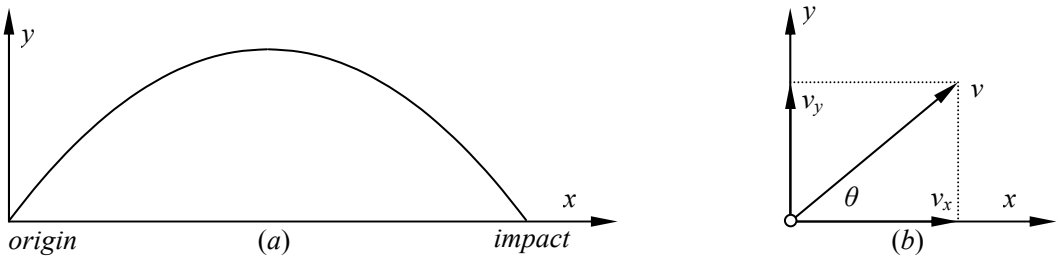

图 4.2

(*a*)如果物体上抛，将会产生一个抛特线轨迹。(*b*)速度可以分解为水平速度和竖直速度，水平速度和合速度之间的夹角为 θ。

$$y(t) = y_0 + v_{y0}t + \frac{1}{2}gt^2 \qquad \textbf{(4.7)}$$

其中 y_0 是初始高度，v_{y0} 代表初速度，g 代表重力加速度。水平位移的计算公式为

$$x(t)=x_0+v_{x0}t \qquad (4.8)$$

x_0 代表初始位移，v_{x0} 代表这个物体的水平初速度水平初速度，竖直初速度与合初速度之间的关系为

$$v_{x0}=v_0\cos\theta \qquad (4.9)$$

$$v_{y0}=v_0\sin\theta \qquad (4.10)$$

假设一个小球的初始位置为(x_0，y_0)为(0，0)，初速度为 20m/s，水平速度和合速度之间的夹角为 θ 度，编写一个程序，画出这个小球的轨迹，并计算小球再次落地与初始位置之间的距离。这程序应当能画出多个抛物线，θ 的取值从 5 到 85 度，每隔 10 度取一次，计算水平位移，θ 的取值从 0 到 90 度，每隔 1 度取一次。最终应当确定那一个 θ 值使得水平位移最大。还有打印不同的抛物线时要用不同的颜色。

答案:

为了解决问题，我们必须知道这个小球落地的时间。然后我们 4.7 到 4.10 计算出小球的位置。如果多次取不同的值，我们将画出小球的轨迹。

小球落地的时间 T，可以通过公式 4.7 得到。满足 $y(t)$=0。

$$y(t) = y_0 + v_{yo}t + \frac{1}{2}gt^2 \qquad \textbf{(4.7)}$$

$$0 = 0 + v_{yo}t + \frac{1}{2}gt^2$$

$$0 = (v_{yo} + \frac{1}{2}gt)t$$

t=0 舍去，故得到下面的值。

$$t_2 = -\frac{2v_{y0}}{g} \qquad (4.11)$$

从问题中我们可知，小球的初始位置为(x_0，y_0)为(0，0)，初速度为 20m/s，计算水平位移，θ 的取值从 0 到 90 度，每隔 1 度取一次。最后我们从基础物理学的课本可知，地球的重力加速度为 9.81m/s^2。

1. 陈述问题

当一个小球以初始角度 θ，初速度 v_0 抛出计算这个小球的落地位移。θ 的取值从 0 到 90 度，每隔 1 度取一次。确定那一个角度的水平位移最大。用出一系列的抛物线，这时 θ 的取值从 5 到 85 度，每隔 10 度取一次。用不同的颜色且粗一点的线画出落地位移最大的抛物线。

2. 定义输入量和输出量根据问题的定义，我们知道不需要输入量。输出为水平位移为最大时的 θ 角和指定抛物线的图象。

3. 设计算法问题分为 5 大步

```
Calculate the range of the ball for θ between 0 and 90°
Write a table of ranges
Determine the maximum range and write it out
Plot the trajectories for θ between 5 and 85°
Plot the maximum-range trajectory
```

因为我们精确地知道循环重复的次数，所以在这里用 for 循环是合适的。我们现在开始定义每一个大步骤的伪代码。为了计算每一个角度小球的落地位移，我们首先应当通过公式(4.9)和(4.10)计算水平初速度和竖直初速度。然后通过 4.11 计算出小球落地的时间。最后通过 4.7 计算出落地位移。具体的伪代码如下所示。注意在用 trigonometric 函数之前，我们必须把角度转化为弧度。

```
Create and initialize an array to hold ranges
for ii = 1:91
    theta ← ii – 1;
    vxo ← vo * cos(theta * conv);
    vyo ← vo * sin(theta * conv);
    max_time ← -2 * vyo / g;
    range(ii) ← vxo * max_time;
end
```

下一步，写出落地的表。伪代码如下

```
Write heading
```

```
for ii = 1:91
    theta ← ii – 1;
    print theta and range;
end
```

我们可以用 max 函数计算最大落地位移。调用这个函数返回最大值和它的位置。伪代码如下

```
[maxrange index] ← max(range);
Print out maximum range and angle (=index -1);
```

我们将用 for 循环嵌套来计算和画出抛物线。。为把所有抛物线都显示在屏幕上，在第一个抛物线的语句后，加上 hold on 命令。每画一个抛物线，就要用到一个 hold on 命令。在画最后一个抛物线时要用到 hold off 命令。我们将在抛物线上取 21 个重要的点，并找了这些的位置。我们将画出这些点。伪代码如下:

```
for ii = 5:10:85
% Get velocities and max time for this angle
theta ← ii – 1;
vxo ← vo * cos(theta * conv);
vyo ← vo * sin(theta * conv);
max_time ← -2 * vyo / g;
    Initialize x and y arrays
    for jj = 1:21
        time ←(jj – 1) * max_time / 20;
        x(time) ← vxo * time;
        y(time) ← vyo * time + 0.5 * g * time^2;
    end
    plot(x,y) with thin green lines
    Set "hold on" after first plot
end
Add titles and axis labels
```

最后，用不同的颜色且粗一点的线画出落地位移最大的抛物线。

```
vxo ← vo * cos(max_angle * conv);
vyo ← vo * sin(max_angle * conv);
max_time ← -2 * vyo / g;
Initialize x and y arrays
for jj = 1:21
    time ← (jj-1) * max_time / 20;
    x(jj) ← vxo * time;
    y(jj) ← vyo * time + 0.5 * g * time^2;
end
plot(x,y) with a thick red line
hold off
```

4. 转化 matlab 语句

```
% Script file: ball.m
%
% Purpose:
% This program calculates the distance traveled by a ball
% thrown at a specified angle "theta" and a specified
% velocity "vo" from a point on the surface of the Earth,
% ignoring air friction and the Earth's curvature. It
% calculates the angle yielding maximum range, and also
% plots selected trajectories.
%
% Record of revisions:
% Date Programmer Description of change
```

```
% ==== ========== ====================
% 12/10/97 S. J. Chapman Original code
%
% Define variables:
% conv Degrees to radians conv factor
% gravity Accel. due to gravity (m/s^2)
% ii, jj Loop index
% index Location of maximum range in array
% maxangle Angle that gives maximum range (deg)
% maxrange Maximum range (m)
% range Range for a particular angle (m)
% time Time(s)
% theta Initial angle (deg)
% traj_time Total trajectory time (s)
% vo Initial velocity (m/s)
% vxo Xcomponent of initial velocity (m/s)
% vyo Ycomponent of initial velocity (m/s)
% x Xposition of ball (m)
% y Yposition of ball (m)
% Constants
conv = pi / 180;        % Degreestoradians conversion factor
g = -9.81;                % Accel. due to gravity
vo = 20;                  % Initial velocity
%Create an array to hold ranges
range = zeros(1,91); % Calculate maximum ranges
for ii = 1:91
    theta = ii - 1;
    vxo = vo * cos(theta*conv);
    vyo = vo * sin(theta*conv);
    traj_time = -2 * vyo / g;
    range(ii) = vxo * traj_time;
end
% Write out table of ranges
fprintf ('Range versus angle theta:\n');
for ii = 1:91
    theta = ii - 1;
    fprintf(' %2d %8.4f\n',theta, range(ii));
end
% Calculate the maximum range and angle
[maxrange index] = max(range);
maxangle = index - 1;
fprintf ('\nMax range is %8.4f at %2d degrees.\n',maxrange, maxangle);
% Now plot the trajectories
for ii = 5:10:85
    % Get velocities and max time for this angle
    theta = ii;
    vxo = vo * cos(theta*conv);
    vyo = vo * sin(theta*conv);
    traj_time = -2 * vyo / g;
    % Calculate the (x,y) positions
    x = zeros(1,21);
    y = zeros(1,21);
    for jj = 1:21
        time = (jj - 1) * traj_time/20;
        x(jj) = vxo * time;
        y(jj) = vyo * time + 0.5 * g * time^2;
    end
```

```
        plot(x,y,'b');
        if ii == 5
        hold on;
    end
    end
    % Add titles and axis lables
    title ('\bfTrajectory of Ball vs Initial Angle \theta');
    xlabel ('\bf\itx \rm\bf(meters)');
    ylabel ('\bf\ity \rm\bf(meters)');
    axis ([0 45 0 25]);
    grid on;
    % Now plot the max range trajectory
    vxo = vo * cos(maxangle*conv);
    vyo = vo * sin(maxangle*conv);
    traj_time = -2 * vyo / g;
    % Calculate the (x,y) positions
    x = zeros(1,21);
    y = zeros(1,21);
    for jj = 1:21
        time = (jj - 1) * traj_time/20;
        x(jj) = vxo * time;
        y(jj) = vyo * time + 0.5 * g * time^2;
    end
    plot(x,y,'r','LineWidth',3.0);
    hold off
```

5. 检测程序

为了检测这个程序，我们计算手动计算了一些值，用来程序的输出结果作比较。

θ	$v_{xo}=v_o\cos\theta$	$v_{yo}=v_o\sin\theta$	$t_2=-2v_{yo}/g$	$x=v_{xo}t_2$
0^o	20m/s	0	0	0
5^o	19.92m/s	1.74m/s	0.355s	7.08m
40^o	15.32m/s	12.86m/s	2.621s	40.15m
45^o	14.14m/s	14.14m/s	2.883s	40.77m

当 ball 程序运行时，将 91 行含有角度和位移的结果。为了节省空间我们只打印其中的一部分。

```
    >> ball
    Range versus angle theta:
      0    0.0000
      1    1.4230
      2    2.8443
      3    4.2621
      4    5.6747
      5    7.0805
    ...
     40   40.1553
     41   40.3779
     42   40.5514
     43   40.6754
     44   40.7499
     45   40.7747
     46   40.7499
     47   40.6754
     48   40.5514
     49   40.3779
     50   40.1553
    ...
```

```
    85    7.0805
    86    5.6747
    87    4.2621
    88    2.8443
    89    1.4230
    90    0.0000

Max range is    40.7747 at 45 degrees.
```

画图的结果为图 4.3。程序运算结果与手动运算结相符。当角度为 45 度时，位移最大。

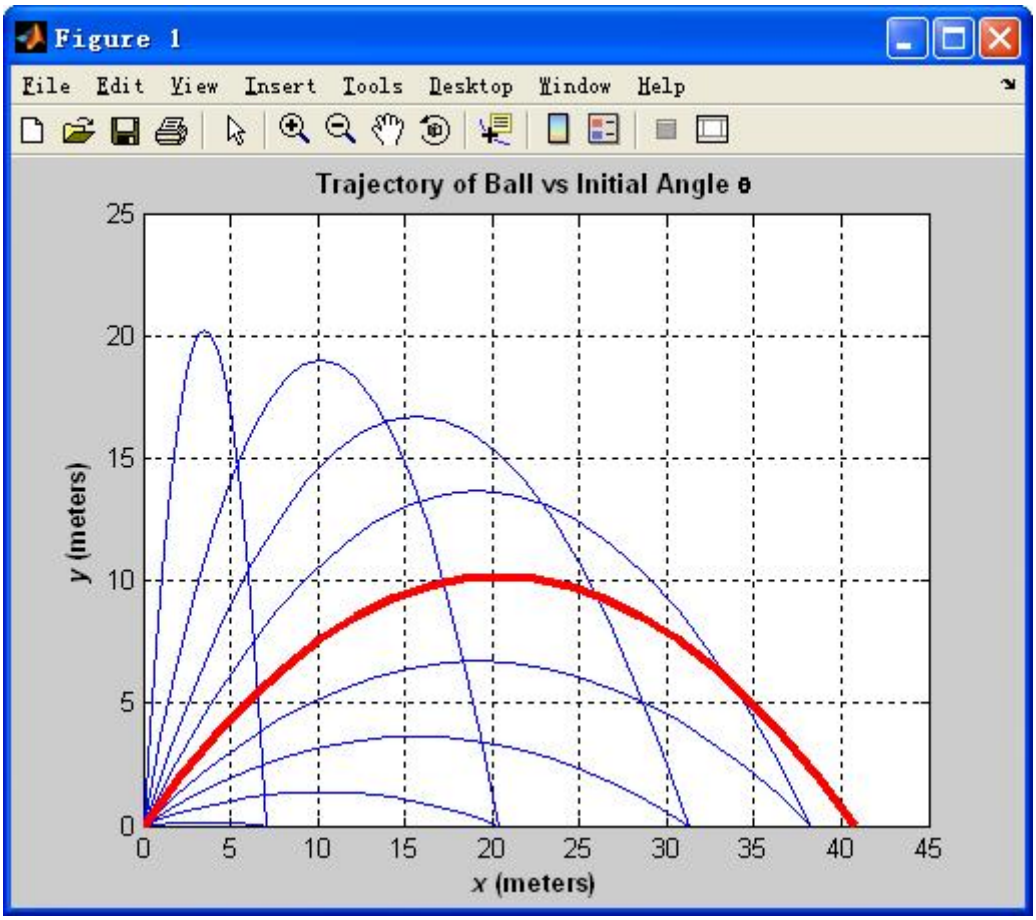

图 4. 3

This example uses several of the plotting capabilities that we introduced in Chapter 3. It uses the axis command to set the range of data to display, the hold command to allow multiple plots to be placed on the same axes, the LineWidth property to set the width of the line corresponding to the maximum-range trajectory, and escape sequences to create the desired title and x- and y-axis labels.

这个例子用到在第三章中介绍的许多画图功能。我们用 axis 命令来显示水平位移，用 hold 命令让多幅图象在同一坐标系出现。用 linewidth 属性调整曲线的宽度。用转义序列创建所需的标题以及 x，y 坐标轴的标签。

但是这个程序不是最高效的，因为许多的循环可以用向量算法代替。练习题 4.11 将要求你重写并改进 ball.m。

4.5 总结

在 matlab 中有两种基本的循环形式，while 循环和 for 循环。while 循环中，代码的重复的次数是不能确定的，只要满足用户定义的条件，重复就进行下去。相对地，在 for 循环中，代码的重复次数是确定的，在循环开始之前，我们就知道代码重复的次数

了。在两种循环中均可使用 break 语句以跳出循环。

4.5.1 好的编程习惯总结

在有选择结构和循环结构的编程中，要遵循以下的编程指导思想.如果你长期坚持这些原则，你的代码将会有很少的错误，有了错误也易于修改，而且在以后修改程序时，也使别人易于理解.

1.对于 for 循环体总是要缩进两个或更多空格，以增强程序的可读性。

2.在循环体中绝不修改循环指数的值。

3.在循环执行开始之前，总是要预先分配一个数组。这样能大大增加循环运行的速度

4.如果用可能的话，可用逻辑函数选择数组中的元素。如果逻辑数组进行运算，要比循环快得多。

5.如果用可能的话，可用逻辑函数选择数组中的元素。如果逻辑数组进行运算，要比循环快得多。

4.5.2 matlab 总结

下面的总结列举了本章出现的所有特殊符号，命令和函数，后面跟的是简短的描述。

break	break 语句可以中止循环的执行和跳到 end 后面的第一句执行
continue	continue 语句只中止本次循环，然后返回循环的顶部。
for 循环	在 for 循环中，代码的重复次数是确定的
tic 函数	复位内建计时器
toc 函数	从最后一次调用 tic 以秒开始计时
while 循环	while 循环中，代码的重复的次数是不能确定的，只要满足用户定义的条件，重复就进行下去

4.6 练习

4.1

编写 matlab 语句计算 y(t)的值

$$y(t)=\begin{cases}-3t^2+5 & t\geq 0\\ 3t^2+5 & t<0\end{cases}$$

已知 t 从-9 到 9 每隔 0.5 取一次值。运用循环和选择语句进行计算。

4.2

用向量算法解决练习 4.1。

4.3

编写 matlab 语句计算并打印出 1 到 50 之间所有整数的平方。创建一个包含有每一个整数和他相应平方的和，注意在每一列加上合适的标签。

4.4

在 M 文件中编写程序，$y(x)=x^2-3x+2$，x 从 0.1 到 3 每隔 0.1 取一次值。用两种算法，一种是应用 for 循环，另一种是用数组。用 3point 粗的红虚线画出产生的函数。

4.5

在 M 文件中编写程序，计算阶乘 N!。注意 0!，如果 N<0 则报告出错。

4.6

检测下面的 for 语句，确定循环运行的次数。

a. for ii = -32768:32767
b. for ii = 32768:32767
c. for kk = 2:4:3
d. for jj = ones(5,5)

4.7

检测下面的 for 循环，确定每一次 for 循环结束的时侯 ires 的值。和每个 for 循环的次数。

```
a. ires = 0;
   for index = -10:10
       ires = ires + 1;
   end

b. ires = 0;
   for index = 10:-2:4
      if index == 0
         continue
      end
      ires = ires + index;
   end

c. ires = 0;
   for index = 10:-2:4
      if index == 0
         break;
      end
      ires = ires + index;
   end

d. ires = 0;
   for index1 = 10:-2:4
      for index2 = 2:2:index1
         if index2 ==6
            break;
         end
         ires = ires + index2;
      end
   end
```

4.8

检测下面的 while 循环，确定每一次 while 循环结束的时侯 ires 的值。和每个 while 循环的次数。

```
a. ires = i;
   while mod(ires,10) ~= 0
      ires = ires + 1;
   end

b. ires = 2;
   while ires <= 200
       ires = ires^2
   end

c. ires = 2;
   while ires > 200
```

```
        ires = ires^2;
    end
```

4.9

当下面的语句执行后，数组 arr1 的结果是多少。

```
a. arr1 = [1 2 3 4; 5 6 7 8; 9 10 11 12];
   mask = mod(arr1,2) == 0;
   arr1(mask) = -arr1(mask);

b. arr1 = [1 2 3 4; 5 6 7 8; 9 10 11 12];
   arr2 = arr1 <= 5;
   arr1(arr2) = 0;
   arr1(~arr2) = arr1(~arr2) .^2;
```

4.10

如何将一个普通的数字数组带有逻辑性?如何去除数字数组的逻辑性?

4.11

修改例 4.8 中的 ball 程序，有向量算法代替内部 for 循环。

4.12

修改例 4.8 中的 ball 程序，在适当的位置读取重力加速度。并计算出最大的水平位移。修改成功后，分别用加速度-9.8m/s^2，-9.7m/s^2，-9.6m/s^2 进行运算。重力加速度的减小将会对小球的落地位移产生什么样的影响。重力加速度的减小会不会对掷球的最佳角度产生影响。

4.13

修改例 4.8 中的 ball 程序，读取小球的初速度。修改成功后，分别用加 10m/s，20m/s，30m/s 进行运算。改变初速度将会对小球的落地位移产生什么样的影响，对最佳抛射角度有没有影响?

4.14

例 4.7 中的程序 lsqfit 在输入数据之前，先要读取要输入数据的个数。我们要修改这个程序使之能够用 while 循环读取任意数量的数据，当用户敲击回车键时结束读取。用例 4。7 中的数据检测你的程序。(提示当用户按下回车键而不输入其他的任何值，input 函数将会返回一个空数组。你能用函数判断一个数组是不是空数组，当它为 true 时停止读取数据。)

4.15

修改 4.7 中的程序 lsqfit，使它能够从 input1.dat 文件中读取它的输入值。文件中的数据是以行组织的，每一行都有一对(x，y)，如下所示:

```
1.1   2.2
2.2   3.3
...
```

用例 4.7 中的数据检测你的程序。(提示:我们用 load 命令从 input1 数组读数据。然后把 input1 的第一列赋值于数组 x，把 input1 的第二列赋值于数组 y)。

4.16

MATLAB Least-Squares Fit Function. MATLAB includes a standard function that performs a least-squares fit to a polynormal. Function polyfit calculates the least-squares fit of a data set to a polynormal of order N

MATLAB 最小二乘匹配函数。Matlab 包括一个标准函数，对多项式进行最小二乘匹配运算。函数 polyfit 用一系列数据对 N 阶多项式进行最小二乘匹配运算。

$$p(x)=a_nx^n + a_{n-1}x^{n-1} + ... + a_1x + a_0 \qquad (4.12)$$

其中 N 是任意大于等于 1。当 N=1 时，得到的是直线方程，带有斜率 a_1 和 y 轴截距 a_0，这个的形式如下：

p = polyfit(x,y,n)

x，y 代表向量 x，y，n 代表阶数。

应用函数 polyfit 修改例 4.7 中的 lsqfit 程序

4.17

在例 4.3 中，已知年月日，计算相应的 thedayofyear。在这个程序中，并没有检测是否输入了正确的年月日，它能接收无效的月和日，并产生无意义的结果。修改你的程序使之只能输入有效的年月日。如果输入的值无效，则提示用户出错，并且跳出执行。我们要求年应当大于 0，月只能是 1 到 12 之间的整数。日只能 1 到那一月的最大数之间的整数。用 switch 结构检查日是否正确。

4.18

编写 matlab 程序计算下列函数的值

$$y(x) = \ln\frac{1}{1-x}$$

x 为任意有意义的值，ln 为自然对数。用 while 循环编写程序，可以重复地输入合法的 x 的值并计算相应的函数值。当一个非法的 x 输入，则中止程序。（所有 x≥1 的数均为非法值）

4.19

斐波那契数列。含有 n 个数的斐波那契数列的定义如下：

f(1) = 1

f(2) = 2

f(n) = *f(n-1)* + *f(n-2)*

所以 *f(3)=f(2)+f(1)=2+1=3*，还有更多的数。在 M 文件中编写一程序，计算并写斐波那契数列中第 n（n>2）个数的值，n 由用户输入。用 for 循环进行计算。

4.20

通过二极管的电流

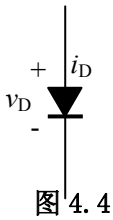

图 4.4

如图 4.4 所示的半导体二极管，通过它的电流可由下面的公式得到

$$i_D = I_o(e^{\frac{qv_D}{kT}} - 1) \qquad (4.13)$$

v_D 代表管压降，单位为伏（V）

i_D 代表通过二极管的电流，单位为安（A）

i_o 代表反向饱和电流

q 代表电子的电量 1.602×10^{-19} 库仑（C）

k 代表玻尔兹曼常数 1.38×10^{-23}J/K

T 代表开尔文温度

4.21

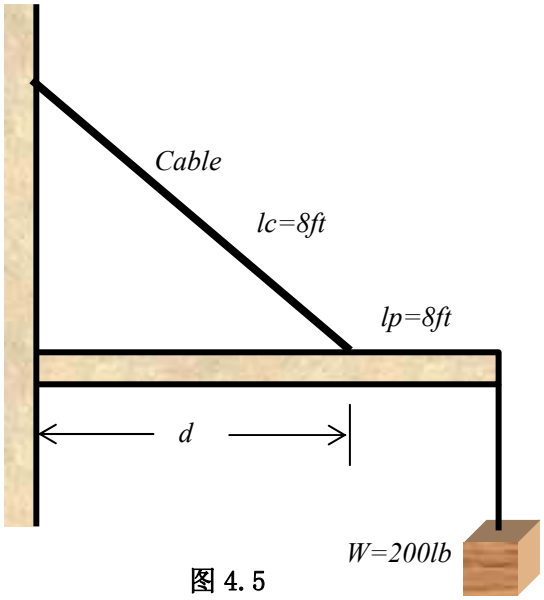

图 4.5

轻绳上的拉力。一重 200 英磅的物体被固定在一水平杆的末端。如图 4.5 所示这一水平杆由一轻绳固定。绳子上的拉力为

$$T = \frac{W \cdot lc \cdot lp}{d\sqrt{lp^2 - d^2}} \qquad (4.14)$$

T 代表绳子的拉力，W 代表物体的重量，lp 代表杆的长度，lc 为绳长，d 代表绳与杆的结点到墙面的距离。编写一个程序，以确定 d 为多大时，绳的拉力最小。为达此目的，d 应从 1 英尺到 7 英尺，每隔 1 英尺取一次值，并找出使拉力最小的 d。

4.22

病毒的繁殖。假设某生物学家做实验，测试一种特殊病毒在不同的培养基下的繁殖速率。实验表明 A 培养基中的病毒每 60 分钟复制一次，A 培养基中的病毒每 90 分钟复制一次。假设在两个培养基中开始的时候各有一个病毒。编写一个程序，计算 24 小时之内每隔三小时病毒在各自培养基中的个数，并画出图象。画两个图，一个用线性坐标，一个用线性对数坐标。24 小时之后两培养基中的病毒的数目分别是多少？

4.23

分贝

工程师们经常用分贝或 dB 来描述两功率之比。1dB 的定义如下，

$$dB = 10\log_{10}\frac{P_2}{P_1} \qquad (2.13)$$

P_2 是已测量的功率，P_1 代表参考功率。假设参考功率 P_1 是 1 瓦。P_2 从 1 到 20 瓦每隔 0.5 瓦取一次值，编写程序，计算相应的 dB 值，并画出 dB-P_2

4.24

图象

几何平均数(geometric mean)。数列中的元素从 x_1 到 x_n，它们的几何平均数的定义如下

$$\text{geometric mean}=\sqrt[n]{x_1x_2x_3...x_n} \qquad (4.16)$$

编写一个程序，它能接受任意个数的正输入值，并计算它们的算术平均数和几何平均

数。用 while 循环读取输入值，当输入一个负数中止输入数据。计算数列 10，5，2，5 的算术平均数和几何平均数，用以检测程序。

4.25

均方根平均数（rmsaverage）。均方根平均数是另一种计算数据平均数的方法。它的定义如下

$$\text{rms average} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2} \qquad (4.17)$$

编写一个程序，它能接受任意个数的正输入值，并计算它们的算太平均数和几何平均数。用 while 循环读取输入值，当输入一个负数中止输入数据。计算数列 10，5，2，5 的均方根平均数，用以检测程序。

4.26

调和均数(harmonic mean)。调和均数也是另一种计算数据平均数的方法。它的定义如下

$$\text{harmonic mean} = \frac{N}{\frac{1}{x_1}+\frac{1}{x_2}+\ldots+\frac{1}{x_N}} \qquad (4.18)$$

编写一个程序，它能接受任意个数的正输入值，并计算它们的算太平均数和几何平均数。用 while 循环读取输入值，当输入一个负数中止输入数据。计算数列 10，5，2，5 的调和均数，用以检测程序。

4.27

编写一个程序，能够计算一系列正数的算术平均数，几何平均数，均方根平均数，调和均数。可使用任意算法读取输入值。用下列数测试你的程序。

a. 4, 4, 4, 4, 4, 4, 4
b. 4, 3, 4, 5, 4, 3, 5
c. 4, 1, 4, 7, 4, 1, 7
d. 1, 2, 3, 4, 5, 6, 7

4.28

平均无故障时间（Mean Time Between Failure）。电器设备的可靠性是用平均无故障时间来度量的，它是指设备正常工作的平均时间。对包含许多电器设备的大系统来说，它的平均无障碍时间取决于每一个部分平均无障碍时间，我们可以通过第一个部分的平均无障碍时间计算整体的平均无障碍时间。如果系统的内部结构如图 4.6 所示，那么全局的 MTBF 为

$$\text{MTBF}_{sys} = \frac{1}{\frac{1}{\text{MTBF}_1}+\frac{1}{\text{MTBF}_2}+\ldots+\frac{1}{\text{MTBF}_n}} \qquad (4.19)$$

编写一个程序，读取每一个部分的平均无障碍时间，然后计算出全局的 MTBF。用下面的数据测试你的程序，subsystem1，2，3，4 的平均无障碍时间分别为 2000 小时，800 小时，3000 小时，5000 小时。

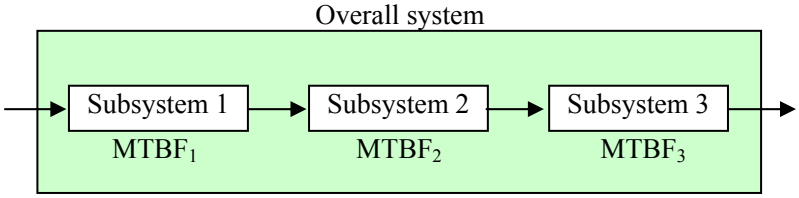

图 4. 6 MTBF