

MATLAB 八讲



该教程转至东华大学-宋新山

小木希望学園

シャオムウ希望学園

Xiao Mu Hope School

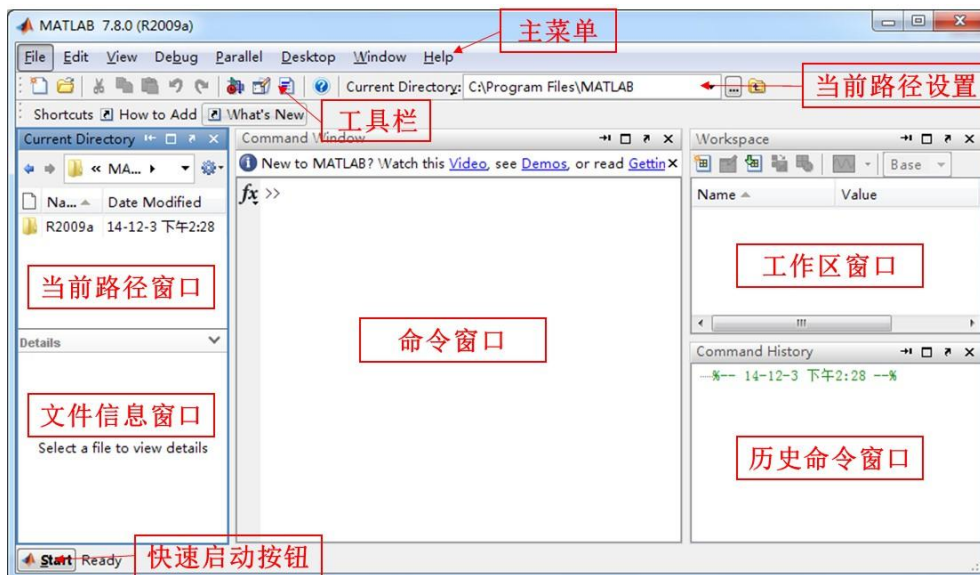
牟天蔚

2019.01

第一章 MATLAB 界面简介*

1.1 MATLAB 主界面

MATLAB 的默认窗口如下图所示，其中包括主菜单、工具栏、命令窗口、历史命令窗口、工作区浏览窗口和当前路径窗口。



1.2 命令窗口

命令窗口是 MATLAB 的重要组成部分，是用户和 MATLAB 交互的工具，是 MATLAB 执行函数命令的窗口。默认情况下，MATLAB 的命令窗口总是打开的（如图示）。

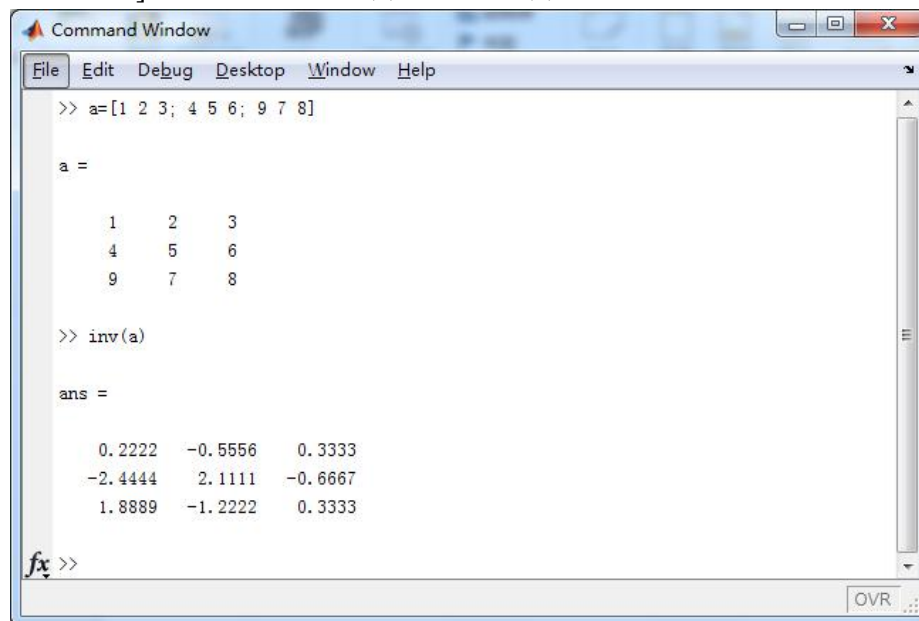


在命令窗口中>>后面可直接输入命令语句，并得到计算结果。就象在演算纸上直接进行运算一样。

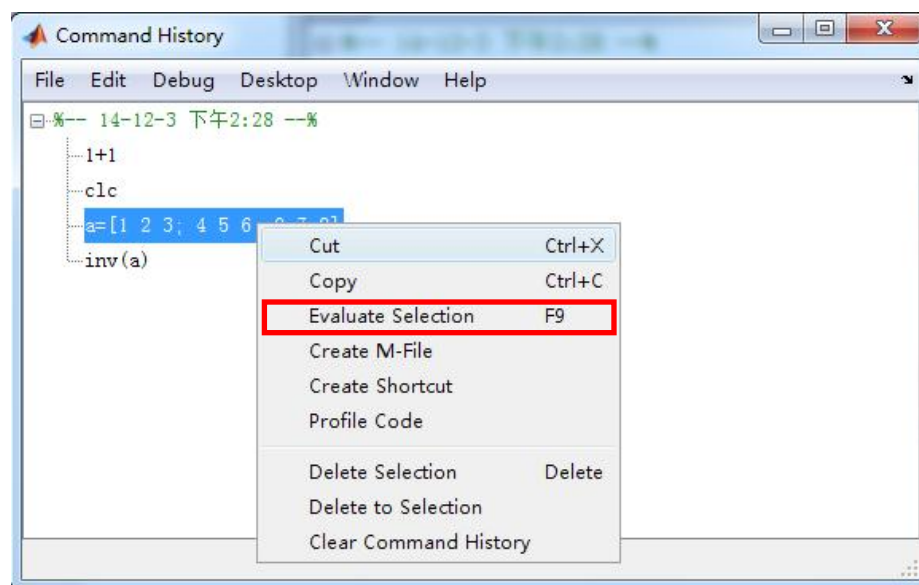
例如：

- (1) 输入 1+1，回车，得到：ans=2
- (2) 输入 a=[1 2 3; 4 5 6; 9 7 8]（表示 3×3 矩阵），并接着输入 inv(a)（对矩阵 a 求逆），立即得到 ans=[0.2222 -0.5556 0.3333; -2.4444 2.1111 -0.6667; 1.8889

-1.2222 0.3333]。这就是原始矩阵 a 的逆矩阵（如图示）。



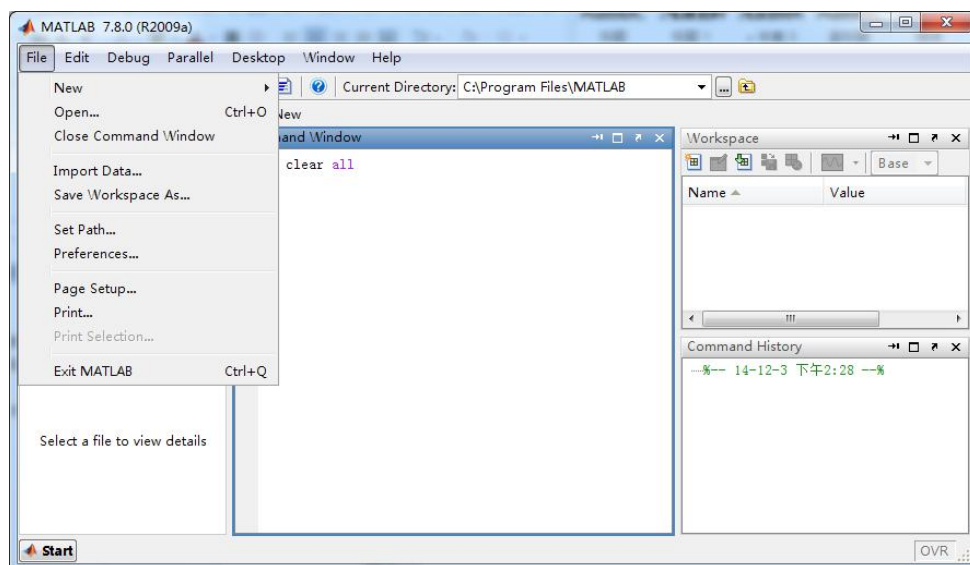
（3）输入 `clc`，Matlab 命令窗口里面的记录就会被擦除掉，但是历史命令窗口还保留原先输入的命令。双击历史命令窗口中的命令（或右键需要执行的命令选择 **Evaluate Selection**），MATLAB 能自动在命令窗口中输入先前输过的命令并执行。



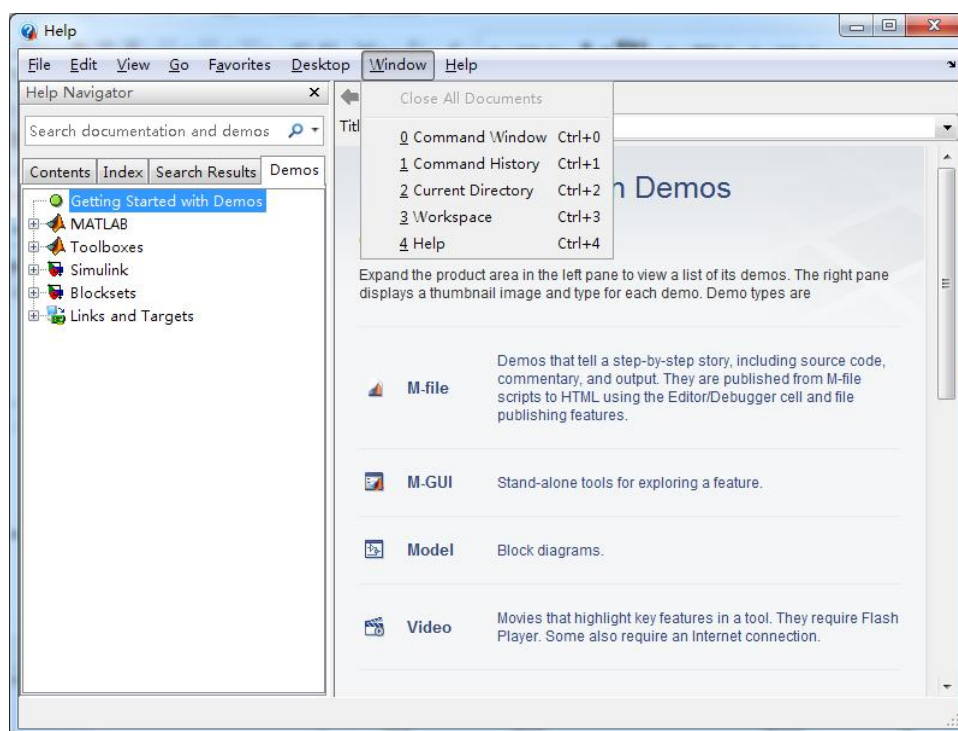
1.3 菜单和工具条

MATLAB 的菜单和其他一些应用程序相比，比较简单，但其功能十分强大。MATLAB 的菜单有 **F**ile、**E**dit、**W**indow 和 **H**elp。其中 **F**ile 和 **H**elp 菜单较常用。

File 主要用于对文件的操作（新建(new)、打开(open)、保存工作空间（Save Workspace）、显示工作空间（Show Workspace）、装入工作空间（Load Workspace）等（如图示）。**H**elp 菜单主要用于在编程或运算过程中提供帮助。要使用帮助，只要在命令提示符 `>>` 后输入 `help` 即可。当然也可以在帮助菜单中执行相应项目。



另外一个查看 MATLAB 功能的简单方法是在命令提示符后面输入 `demo` (如图示)。



工具条提供了常规的新建、打开、剪切、复制、粘贴、恢复、重做等功能，方便操作。执行这些功能时，不需要从菜单中选取，只要直接鼠标单击按钮即可。

1.4 工作空间浏览

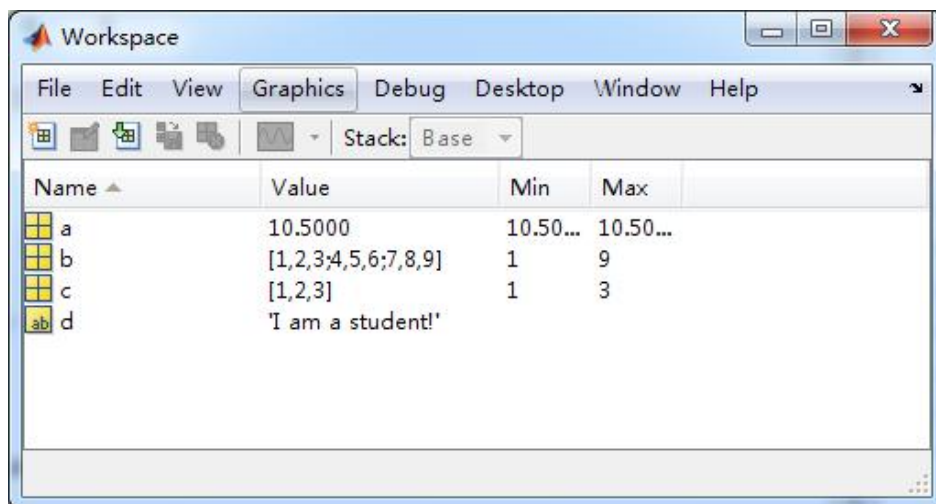
MATLAB 中，工作空间是一个重要概念。工作空间就是由运行 MATLAB 程序命令所生成的变量和 MATLAB 提供的常量所构成的空间。每打开一次 MATLAB，就会自动建立一个工作空间，工作空间在 MATLAB 运行期间一直存在，关闭 MATLAB 后，工作空间会自动消失。

例如，我们在 MATLAB 窗口内输入 `a=10.5`，`b=[1,2,3;4 5 6;9,8 7]`，`c=[1 2 3]`，

d='I am a student!'. 然后在命令窗口内输入 whos, 就会显示:

```
>> whos
```

Name	Size	Bytes	Class	Attributes
a	1x1	8	double	
b	3x3	72	double	
c	1x3	24	double	
d	1x15	30	char	

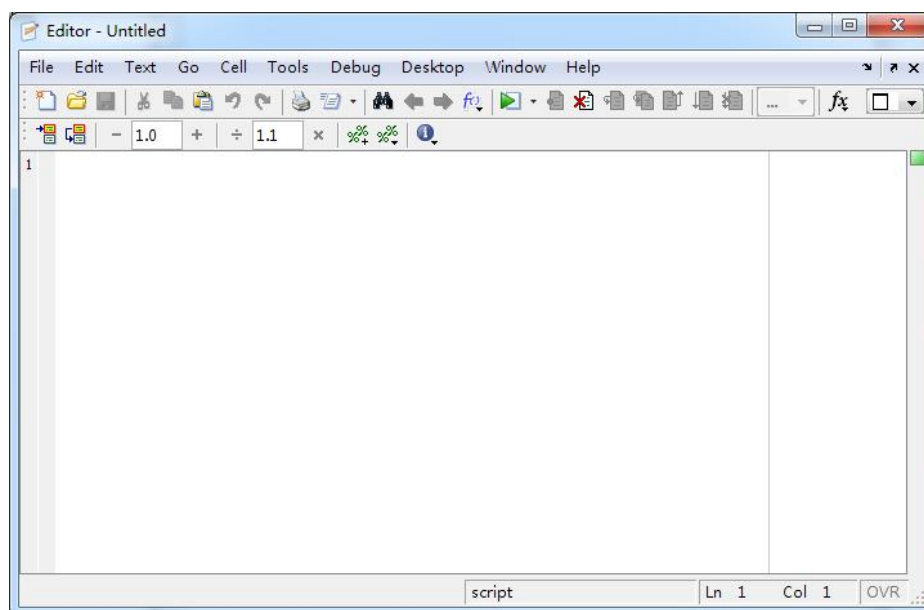


双击工作空间的一个变量, 将打开一个数组编辑器, 可以对该变量的数值类型 (Long、Short 等)、具体数值以及数组的行列等进行修改 (可自己练习)。

在命令窗口输入 clear all 命令, workspace 中的所有变量将被清除!

1.5 程序编辑器

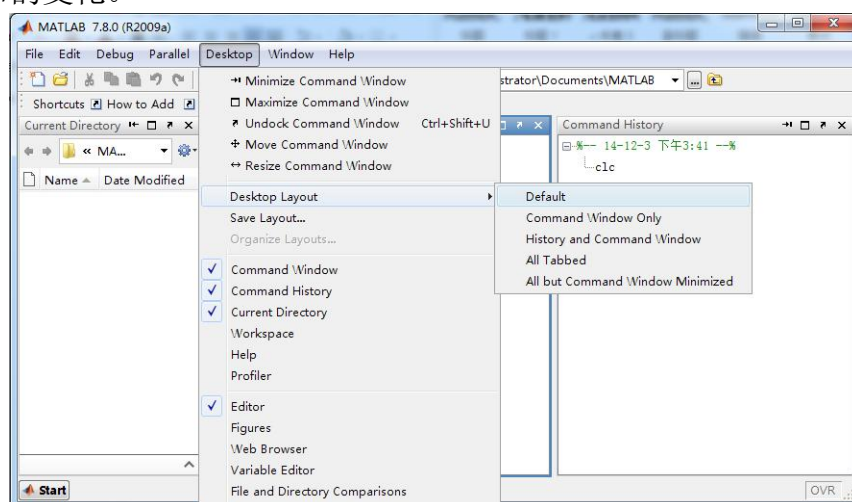
MATLAB 提供了一个内置的具有编辑和调试功能的程序编辑器, 使用该程序编辑器能够调试和编辑 M 文件, 所谓 M 文件, 实际就是 MATLAB 的程序文件, 该文件可以将一系列的 MATLAB 命令组织在一起执行, 这对解决一些复杂问题有重要意义。要打开程序编辑器, 选取 File→New→M-File 即可得到该编辑器界面 (如图示)。



需要注意：MATLAB 中语言编辑所用的字符均为英文字符，中文字符包括标点符号在内都视为错误而不能被正确执行。Matlab 对英文大小写敏感例如：A=10 与 a=10；是分别将 10 赋值给变量 A 和 a。

1.6 本章练习

- (1) 认识 MATLAB 的各种操作界面。
- (2) 在 MATLAB 命令窗口中输入 `path`、`demo`、`help`、`clc` 命令，观察输出结果。
- (3) 首先在 MATLAB 命令窗口中输入一些变量，然后利用命令 `who`、`whos` 查看这些变量，或者用菜单中的 `Workspace` 选项查看这些变量，并试改变这些变量的数值类型、变量数值、数组或矩阵的大小等。用 `load` 命令调用内置变量 `west0479`，并用 `disp` 命令查看。
- (4) 查看 MATLAB 的当前路径，将其设置为 MATLAB 根目录。用你名字的拼音字母链接学号（例如：`liming12345678`）在 D 盘下新建一个文件夹，并将 MATLAB 路径设置到这个文件夹下。
- (5) 使用 `Desktop` 菜单下 `Desktop Layout` 子菜单下的 `Default` 命令，查看 MATLAB 工作窗口的变化。



第二章 MATLAB 基本使用方法

2.1 简单计算

(1) 直接输入法

在命令窗口中直接输入数学表达式，按 Enter 即可得到运算结果：

例 2-1：圆柱体的底面半径为 5，高为 10，计算圆柱体的体积。

```
>> pi*5^2*10
```

```
ans =
```

```
785.3982
```

当没有将结果赋予一个变量时，MATLAB 自动将结果赋予一个零时变量 ans

(2) 存储变量法

利用存储变量法再次求例 2-1。

先计算圆柱体的底面积 s，再利用底面积乘以高计算体积 v。

```
>> s=pi*5^2
```

```
s =
```

```
78.5398
```

```
>> v=s*10
```

```
v =
```

```
785.3982
```

注意：Matlab 里面的数学运算法则是：运算从左到右进行，先括号，再指数，再乘除，后加减。这里指的括号是指小括号（）。中括号[]和花括号{}有别的用途。

例 2-2：判断式子 $5+5*5^{(5+5)}-5$ 的运算顺序

2.2 常用数学函数

例 2-3：已知三角形的三条边长度分别是 1,2, $\sqrt{3}$ ，求长度为 1 和 2 的两条边的夹角大小。

利用余弦定理进行求解。在命令窗口输入：

```
>>a=1;b=2;c=sqrt(3);
```

```
>>cos_alpha=(a^2+b^2-c^2)/(2*a*b)
```

```
cos_alpha=
```

```
0.5000
```

```
>>alpha=acos(cos_alpha)
```

```
alpha=1.0472
```

```
>>alpha=alpha*180/pi
```

```
alpha=60.0000
```


Matlab 中提供了一些基本函数表：三角函数表、指数函数表和对数函数表、取整和求余函数表如下所示：

Matlab 三角函数表

序号	函数名	描述	序号	函数名	描述
1	acos/acosd(x)	反余弦函数	9	sec/secd(x)	正割函数
2	acot/acotd(x)	反余切函数	10	sin/sind(x)	正弦函数
3	acsc/acscd(x)	反余割函数	11	tan/tand(x)	正切函数
4	asin/asind(x)	反正弦函数	12	acosh/cosh(x)	(反) 双曲余弦函数
5	atan/atand(x)	反正切函数	13	acoth/coth(x)	(反) 双曲余切函数
6	cos/cosd(x)	余弦函数	14	acsch/csch(x)	(反) 双曲余割函数
7	cot/cotd(x)	余切函数	15	asech/sech(x)	(反) 双曲正割函数
8	csc/cscd(x)	余割函数	16	asinh/sinh(x)	(反) 正弦函数
			17	atanh/tanh(x)	(反) 双曲正切函数

$$\text{双曲正弦: } \sinh x = \frac{e^x - e^{-x}}{2}$$

$$\text{双曲余弦: } \cosh x = \frac{e^x + e^{-x}}{2}$$

$$\text{双曲正切: } \tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$\text{双曲余切: } \coth x = \frac{1}{\tanh x} = \frac{e^x + e^{-x}}{e^x - e^{-x}}$$

$$\text{双曲正割: } \operatorname{sech} x = \frac{1}{\cosh x} = \frac{2}{e^x + e^{-x}}$$

$$\text{双曲余割: } \operatorname{csch} x = \frac{1}{\sinh x} = \frac{2}{e^x - e^{-x}}$$

Matlab 指数函数和对数函数表

序号	函数名	描述	序号	函数名	描述
1	x^	乘方运算符	8	nthroot(x)	返回实数的 n 次根
2	exp(x)	求幂 (以 e 为底)	9	pow2(x)	求以 2 为底的幂
3	expm1(x)	指数减 1 (exp(x)-1)	10	reallog(x)	求非负实数的自然对数
4	log(x)	求自然对数 (e 为底) ln	11	realpow(x)	求非负实数的乘方
5	log10(x)	求以 10 为底的对数	12	realsqrt(x)	求非负实数的平方根
6	log1p(x)	求 x+1 的自然对数 ln(x+1)	13	sqrt(x)	求平方根
7	log2(x)	求以 2 为底的对数			

Matlab 取整和求余函数

序号	函数名	描述	序号	函数名	描述
1	fix(x)	取整	5	mod(x)	求模或者有符号取余
2	floor(x)	取不大于 x 的最大整数	6	rem(x)	求除法的余数

3	ceil(x)	取不小于 x 的最小整数	7	sign(x)	求符号函数
4	round(x)	四舍五入			

$$\text{sgn}(x) = \begin{cases} 1, x > 0 \\ 0, x = 0 \\ -1, x < 0 \end{cases}$$

2.3 数学运算符

数学表达式的各种符号在 Matlab R2010a 中的对应符号如下表：

数学运算符及其功能

符号	功能	实例	符号	功能	实例
+	加法	3+5=8	^	乘方	3^5=243
-	减法	3-5=-2	.^	矩阵乘方	
*	乘法，矩阵乘法	3*5=15	'	矩阵共轭转置	
.*	点乘，矩阵对应元素相乘		.'	矩阵转置	
/	除法，矩阵除法	3/5=0.6	=	赋值运算符	A=5，把 5 赋值给 A
./	点除，矩阵对应元素相除				

(1) 矩阵乘法和点乘

```
>> A=magic(3)
```

```
A =
```

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

```
>> B=ones(3)*10
```

```
B =
```

```
10    10    10
10    10    10
10    10    10
```

```
>> C1=A.*B
```

```
C1 =
```

```
80    10    60
30    50    70
40    90    20
```

```
>> C2=A*B
```

```
C2 =
```

```

150    150    150
150    150    150
150    150    150

```

可以看出，C1 为矩阵 A 和 B 的乘积，C2 的每个元素是 A 和 B 对应元素的乘积。

(2) 矩阵的乘方和数组的乘方

继续 (1) 中的运算

```
>> C3=A^2
```

```
C3 =
```

```

    91    67    67
    67    91    67
    67    67    91

```

```
>> C4=A.^2
```

```
C4 =
```

```

    64     1    36
     9    25    49
    16    81     4

```

C3 为矩阵 A 的平方，C4 为矩阵对应元素的平方。

2.4 关系运算符

MATLAB 中主要有 6 种关系运算符。

关系运算符

符号	描述	符号	描述
<	小于	>=	大于等于
<=	小于等于	==	等于
>	大于	~=	不等于

需要注意的是：MATLAB 中当比较的是两个同样大小的矩阵时，则分别比较其中的对应元素，并返回一个同样大小的逻辑矩阵（结果真时为 1 (true)，结果假时 0 (false)），当对标量和数组进行比较时，则将该标量和数组中的每一个元素进行比较。

例如：>>a=[1 2;3 4];

```
>>b=magic(2)
```

```
b =
```

```

     1     3
     4     2

```

```
>>c=a>b %判断 a 是否大于 b
```

```
c =
```

```

     0     0
     0     1

```

```
>>d=a~=b %判断 a 是否不等于 b
d =
     0     1
     1     1
```

2.5 逻辑运算符与逻辑运算函数

Matlab 中常用的逻辑运算符如下表:

Matlab 中常用的逻辑运算符

符号	描述	符号	描述
&	逻辑与, a&b, a、b 均为非 0 时返回 1, 否则返回 0		逻辑或, a b, a、b 均为 0 时返回 0, 否则返回 1
&&	逻辑与, 只适用于标量, 不能判断矩阵。a&&b, 当 a 的值为假时, 则忽略 b 的值		逻辑或, 只适用于标量。a b, 当 a 的值为真时, 则忽略 b 的值。
~	逻辑非, ~a, a 为非 0 时返回 0, a 为 0 时返回 1	xor	逻辑异或, xor(a,b), a、b 均为非 0 或均为 0 时返回 0, 否则返回 1

需要指出的是: 上述逻辑运算符对 a,b 为矩阵或数值均可进行。若 a、b 为两个同样大小的矩阵, 则返回一个相同维数的逻辑矩阵, 逻辑矩阵的元素值(0 或 1)为两个矩阵在相同位置处的对应元素的逻辑运算结果。

例如: >>a=[1 2;3 4];

```
>>b=eye(2);
```

```
>>c=a&b %逻辑与运算
```

```
c =
```

```
     1     0
     0     1
```

```
>>d=xor(a,b) %逻辑异或运算
```

```
d =
```

```
     0     1
     1     0
```

例如:

```
>>a=5,b=9
```

```
a=
```

```
5
```

```
b=
```

```
9
```

```
>>c1=(a<b)&&(b/a==fix(b/a))
```

```
c1=
```

```
0
```

```
>>c2=(a<b)|| (b/a==fix(b/a))
```

```
c2=
```

```
1
```

注意：Matlab 中运算符的优先级如下表：

优先级	运算符
1	圆括号()
2	转置(.')、共轭转置(')、乘方(^)、矩阵乘方(^)
3	标量(+)、减法(-)、取反(~)
4	乘法(.*)、矩阵乘法(*)、除法(/)、矩阵除法(/)
5	加法(+)、减法(-)、逻辑非(~)
6	冒号运算符(:)
7	小于(<)、小于等于(<=)、大于(>)、大于等于(>=)、等于(==)、不等于(~=)
8	数组逻辑与(&)
9	数组逻辑或()
10	逻辑与(&&)
11	逻辑或()

$-(1+4)^2/5*2*3+32>100\&1$

$1+4=5$

$5^2=25$

-25

$-25/5=-5$

$-5*2=-10$

$-10*3=-30$

$-30+32=2$

$2>100=0$

$0\&1=0$

除了逻辑运算符以外，Matlab 还提供了大量逻辑运算函数，可以满足程序中的更多要求。Matlab 中逻辑运算函数如下表：

Matlab 中逻辑运算函数

函数	功能	调用格式举例
all	判断数组元素是否全部为非零	$b=all(a)$, $b=all(a,dim)$
any	判断数组是否存在非零元素	$b=any(a)$, $b=any(a,dim)$
false	逻辑 0 (假)	false, false(n) 等
true	逻辑 1 (真)	true, true(n) 等
find	查找非零元素的下标及其值	$ind=find(x)$, $ind=find(x,k)$
is*	查看元素状态	代表一类函数，如 iscell 等
iskeyword	判断字符串是否为 matlab 的关键字	$tf=iskeyword('str')$, iskeyword str
isvarname	判断字符串是否为有效变量名	$tf=isvarname('str')$, isvarname str
logical	将数值变量转化为逻辑变量	$k=logical(a)$
xor	逻辑“异或”	$c=xor(a, b)$

(1) 函数 `all(a)` 返回与矩阵 `a` 相同列数的行向量, 测定矩阵 `a` 中每一列的元素是否都非零, 若某列中所有元素全部非零, 则在该列位置处返回 1, 否则返回 0;

(2) 函数 `any(a)` 返回与矩阵 `a` 相同列数的行向量, 测定矩阵 `a` 中每一列是否有非零元素, 若某列中有非零元素, 则在该列位置处返回 1, 否则返回 0。

例如 `>> a=[1 2 0;3 0 0; 2 1 0]`

`a =`

```

    1     2     0
    3     0     0
    2     1     0

```

`>>all(a)`

`ans = 1 0 0`

`>>any(a)`

`ans = 1 1 0`

(3) 函数 `exist('a')` 用于测试变量 `a` 是否存在, 若存在则返回 1, 否则返回 0。

(4) `k=find(x)` 返回数组 `x` 的非零元素的位置, 若无非零元素, 则返回空矩阵。

`[i,j]=find(x)` 返回矩阵 `x` 中非零元素的行(存贮在 `i` 向量)和列(存贮在 `j` 向量)位置。

`[i,j,k]=find(x)` 返回矩阵 `x` 中非零元素的行和列位置, 并返回包含所有非零元素的向量。例如:

`>>x=[1 0 0 3 4];`

`>>y=eye(3);`

`>>a=find(x)`

`a = 1 4 5`

`>>[i,j]=find(y)`

`i =`

```

    1
    2
    3

```

`j =`

```

    1
    2
    3

```

(5) “`is*`”函数用于检测矩阵所处的状态。`isempty(a)` 函数用于检查 `a` 是否空矩阵; `isletter(s)` 函数用于检查字符串 `s` 中的元素是否为字母; `isglobal(a)` 函数用于检测变量 `a` 是否全局变量。

2.6 Matlab 中的标点符号

在 Matlab 中, 标点符号有着充分的意义, 可以用标点符号进行运算, 或者用标点符号代表特定的意义。Matlab 中的标点符号如下表:

Matlab 中的标点符号

标点符号	定义	标点符号	定义
分号 (;)	数组行分隔符; 取消	点 (.)	小数点; 结构体成员

	运行显示		访问号
逗号 (,)	数组列分隔符; 函数参数分隔符	省略号 (...)	续行符
圆括号 (())	指定运算优先级; 函数参数调用; 数组索引	引号 (")	定义字符串
方括号 ([])	定义矩阵	等号 (=)	赋值语句
花括号 ({})	定义单元数组	感叹号 (!)	调用操作系统运算
冒号 (:)	在数组中应用较多, 如生成等差数列	百分号 (%)	注释语句的标识

下面对常用符号进行介绍:

(1) 分号 (;)

分号用于区分数组的行, 或者用于一个语句的结尾处, 取消运行的显示。

例如:

A=ones(3); 看上去什么也没有发生, 但是在 workspace 里面已经存在变量 A。

>> B=ones(3) 这里给出了 B=ones (3) 的运行结果, workspace 里面也有 B。

B =

```

1     1     1
1     1     1
1     1     1

```

直接输入 A 也可以用来查看结果, 例如:

>>A

A =

```

1     1     1
1     1     1
1     1     1

```

(2) 百分号 (%)

百分号用于注释, 提高程序的可读性。百分号后面出现的文本可以是中文也可以是英文。

例如:

>>A=ones(3) % create a 3*3 magic matrix

注释语句在 matlab 中间是用绿色表示的。注释语句不会被执行!

(3) 省略号 (...)

当语句太长, 可以用省略号续行。例如:

A=8+7+6+6-97665+...

5-9+48*7

2.7 Matlab 中常用的操作命令

命令	功能	命令	功能
clc	清除命令窗口	hold	图形保持命令
clear	清除工作区中的变量	quit	退出 matlab

clf	清除图形窗口	save	保存内存变量
disp	显示变量或文字内容	path	显示搜索目录

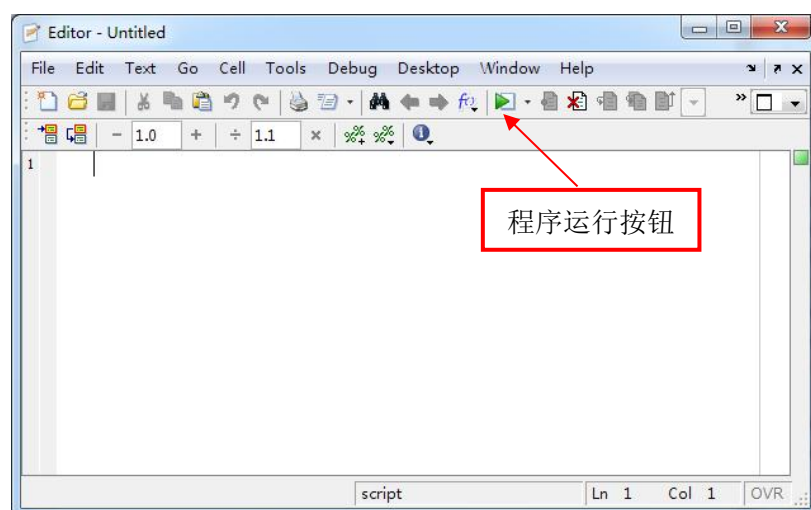
2.8 Matlab 脚本文件

对于一些简单的问题，当需要的命令数很少时，用户可以直接在 Matlab 的命令窗口输入命令。但是，对于多数问题，所需的命令很多，或者需要逻辑计算，进行流程控制。此时，采用直接输入命令的方法会很不方便。针对这个问题，一个合理的方法就是使用脚本文件（**M-文件**）。脚本文件是代码的集合，即允许用户将一系列 Matlab 命令输入到一个简单的脚本文件中，然后在 Matlab 中执行该文件，则会依次执行该文件中编辑的各道命令。

（1）脚本文件的用法

新建脚本文件有两种方法，单击工具栏中的“新建”图标，或者选择 File|New|M-file。新建后系统会打开文件编辑窗口，在窗口中输入文件内容即可。

注意：编辑好的 M 文件必须保存后才能点击运行。



例：编写求圆柱体体积的脚本文件。

新建一个 M 文件，在编辑窗口中输入如下命令：

```
% script m-file example: calculate the volume and surface area of a colume
r=1; %the radius of the colume
h=1; % the height of the colume
s=2*r*pi*h+2*pi*r^2; % calculate the surface area
v=pi*r^2*h; % calucate the volume
disp('the surface area of the colume is :'), disp(s)
disp('the volume of the colume is:'), disp(v)
```

编辑完成后，可以选择 Debug| save and run 命令保存并立即执行该脚本。或采用快捷键 F5，这里选择保存，文件名为 colume，并执行该脚本的代码。结果如下：

```
>> colume
the surface area of the colume is :
```


12.5664

the volume of the colume is:

3.1416

在使用脚本的时候需要注意：在当前工作区中存在与脚本 m 文件同名的变量时，若输入该文件名，则系统将其作为变量执行。比如：

>>colume=30 (系统中工作区 workspace 中生成了 colume 这个变量并赋值 30)

>>colume

colume=

30

可见在工作区中定义了 colume 变量后，在命令窗口中输入 colume，显示的是变量 colume 的值，而没有执行 m 文件 colume。此时可以用 clear 命令清除 colume 这个变量，

>>clear('colume')

再执行 colume 脚本

>> colume

the surface area of the colume is :

12.5664

the volume of the colume is:

3.1416

2.9 本章练习

1. 计算下式

(1) $\sin(60^\circ)$; (2) e^3 ; $\cos(\frac{3}{4}\pi)$

2. 设 $u=2, v=3$, 计算:

(1) $4\frac{uv}{\log v}$; (2) $\frac{(e^u + v)^2}{v^2 - u}$; (3) $\frac{\sqrt{u-3v}}{uv}$

3. 判断下面语句的运算结果

(1) $4 < 20$; (2) $4 \leq 20$; (3) $4 == 20$; (4) $4 \sim 20$; (5) $'b' < 'B'$

4. 设 $a=39, b=58, c=3, d=7$, 判断下面表达式的值

(1) $a > b$; (2) $a < c$; (3) $a > b \&\& b > c$; (4) $a == d$; (5) $a | b > c$; (6) $\sim d$

5. 编写脚本，计算第 2 题中的表达式

6. 三角形的三条边长度分别为 $a=5\text{cm}, b=6\text{cm}, c=7\text{cm}$, 编写脚本，求三角形的最小角的角度值。

7. 练习以下内容并给出解释：(1) $x=5, z=x>[1\ 2\ 3;4\ 5\ 6;7\ 8\ 9]$; (2) $x=5*\text{ones}(3), z=x>[1\ 2\ 3;4\ 5\ 6;7\ 8\ 9]$; (3) $z=\text{strcmp}('yes','no'), z=\text{strcmp}('yes','yes')$ 。

8. 设有如下矩阵 $a=[1\ 2\ 3\ 0;3\ 2\ 0\ 0;5\ 0\ 1\ 0;6\ 1\ 2\ 0]$, $b=\text{eye}(4)$, 求 $a\&b$, $a|b$, $\sim a$, $\text{xor}(a,b)$ 。

第三章 数组

标量运算是数学运算的基础,但对多个数重复进行计算时,仍采用标量的形式则会浪费系统时间,为此 MATLAB 引入了数组运算的形式。

3.1 数组的创建

(1) 直接构造数组

输入数组中的每一个元素可直接构造出数组,需要以“[”开始,以逗号或空格为间隔输入元素值,然后以“]”结束即可。例如:

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

```
>>x=10*sin(a)+10*cos(a)
```

```
x = 13.8177 4.9315 -8.4887 -14.1045 -6.7526 6.8075 14.1089 8.4386 -4.9901
```

(2) 冒号法

基本格式为 **X=初始值:增量:终了值**。就返回以初始值开始,以增量为步长,直到不超过终了值的所有元素构成的数组向量,默认步长为 1.0。

例如: `>>b=1:1.5:10`

```
b = 1.0000 2.5000 4.0000 5.5000 7.0000 8.5000 10.0000
```

```
>>size(b)
```

```
ans = 1 7(表示数组 b 为 1 行 7 列)
```

(3) 调用函数 Linspace 或 Logspace

函数 Linspace (a,b,n) 返回以 a 为起点, b 为终点的等间距共含有 n 个元素的数组。函数 Logspace (a,b,n) 与函数 Linspace (a,b,n) 的功能相似, 只是在函数 Linspace (a,b,n) 的基础上再对其中的每个元素进行 10 的指数幂运算。

例如: >>linspace(0,1,5)

```
ans = 0      0.2500    0.5000    0.7500    1.0000
```

>>logspace(0,1,5)

```
ans = 1.0000    1.7783    3.1623    5.6234   10.0000
```

(4) 组合方法

利用已有的数组可以生成新的数组。既可以利用原有数组的全部元素, 也可以提取其中的部分元素组合为新的数组。

例如: >>x=1:2:6 %生成数组 x

```
x = 1      3      5
```

>>y=6:2:10 %生成数组 y

```
y = 6      8     10
```

>>z=[x,y] %组合为新数组 z, 数组的连接

```
z = 1      3      5      6      8     10
```

>>w=[x(1:2),y(2:3),3.14] %组合为新数组 w

```
w = 1.0000    3.0000    8.0000   10.0000    3.1400
```

3.2 数组中元素的引用

(1) 利用下标直接对数组中元素进行引用

例如>>x=1:2:10 %定义数组 x

```
x = 1      3      5      7      9
```

>>y=x(2)+0.1 %引用数组 x 中的第二个元素

```
y = 3.1000
```

>>z=x(1:3) %引用数组 x 中的第一、二、三个元素

```
z = 1      3      5
```

>>w=x([1,2,5]) %引用数组 x 中的第一、二、五个元素

```
w = 1      3      9
```

如果要引用整行或者整列元素, 可以用冒号 (:) 实现, 例如:

>>A=[2 5 7 10; 1 3 0 12];

>>B=A(1,:)

```
B =
```

```
2      5      7     10
```

B 取 A 数组的第一行, 所有列的数。

>>C=A(2, 2:end)

```
C =
```

```
3      0     12
```

C 取 A 数组的第二行, 第 2 列至最后一列的数。

(2) 数组的扩充

如果在超出数组范围的位置写入时, Matlab 将自动对数组进行扩充。如果数

组当前元素位置和将要写入新数值的位置之间缺失数据，缺失数据填充为 0。例如：

```
>>A=[2 5 7; 1 3 0]
>>A(4,1)=3
A=
2 5 7
1 3 0
0 0 0
3 0 0
```

(3) 数组中移除元素

可以使用空数组[]的一种特殊用法来完成。空数组不包含任何元素。但是由于数组必须保持矩形，因此元素必须整行或整列移除。例如：

```
>>A=[2 5 7; 1 3 0; 2 4 6]

2 5 7
1 3 0
2 4 6
>>A[3,:]=[] %移除第三行中所有的元素。
```

3.3 数组运算

MATLAB 数组运算包括标量与数组之间的运算和数组与数组之间的运算，与矩阵运算不同，这两种运算都是在元素之间进行的。

(1) 标量与数组的运算

标量与数组进行的四则运算就是该标量与数组中每个元素之间进行的四则运算。

```
例如：>>x=1:2:10 %定义数组 x
x =1      3      5      7      9
>>y=2*x+1 %标量和数组 x 之间的四则运算
y =3      7     11     15     19
```

(2) 数组与数组之间的运算

当两个数组具有相同的维数时，它们之间可以进行四则运算，其实这种运算是对应元素之间的四则运算。需要注意的是数组之间的乘法和除法运算必须在运算符号前加“.”，否则将会被示为矩阵运算。

```
例如 x=1:2:10 %定义数组 x
x =1      3      5      7      9
y=11:2:20 %定义数组 y
y =11     13     15     17     19
>>z1=x+y
z1 =12     16     20     24     28
>>z2=x-y
z2 =-10    -10    -10    -10    -10
>>z3=x.*y % 数组与数组的相乘是对应元素的相乘，因此必须用“点”乘。
z3 =11     39     75    119    171
```

```
>>z4=x./y
z4 =0.0909    0.2308    0.3333    0.4118    0.4737
```

注意：由于数组和数组之间的四则运算都是对应元素之间的运算，因此数组的维数一定相同，即两个数组间必须要有相同的行和列，数组与单个数的运算则可以不用点乘，数组中的每个元素都与这个数进行运算。

(3) 数组的逻辑运算

如果两个数组的维数相同或者其中一个标量，对于两个数组中单个元素的逻辑操作就可以共同执行。结果将是原数组具有相同大小，去布尔值（0 或 1）的数组。通过下例了解逻辑数组的操作方式。

例如，在命令窗口中输入如下命令：

```
>> A=[2 5;1 3]
A =
     2     5
     1     3
>> B=[0 6;3 2]
B =
     0     6
     3     2
>> A>=4
ans =
     0     1
     0     0
>> A>=B
ans =
     1     0
     0     1

>> C=[1 2 3 4];
>> A>C
??? Error using ==> gt
Matrix dimensions must agree.
```

3.4 数组操作

(1) 数组的行列转换

数组的行列转换是通过运算符“'”进行的。

例如：>>x=1:2:5

```
x =1     3     5
>>y=x' %将行向量转换为列向量
y =1
    3
    5
```

(2) 数组的下标

通过对数组下标的引用可抽取数组内的元素或对数组进行扩展。

例如: >>u=[1:3;4:6;7:9] %定义一个 3 行 3 列的数组 u

```
u = 1     2     3
     4     5     6
     7     8     9
```

>>v1=u(2:3,1:2) %从 u 中抽取第二、三行, 一、二列的元素组成新数组 v1

```
v1 = 4     5
     7     8
```

>>v2=u(:,[1,3]) %从 u 中抽取第所有行, 一、三列的元素组成新数组 v2

```
v2 = 1     3
     4     6
     7     9
```

通过引用下标, 可以直接输入元素来扩展数组。

例如: 对上述数组 u

>>u(4,:)= [11 12 13] %对数组 u 的行进行扩展

```
u = 1     2     3
     4     5     6
     7     8     9
    11    12    13
```

>>u(:,4)= [1; 1; 1; 1] %对数组 u 的列进行扩展

```
u = 1     2     3     1
     4     5     6     1
     7     8     9     1
    11    12    13     1
```

(3) 数组操作的几个函数

rand(m,n)—生成 m 行 n 列的数组 (矩阵), 其中元素为 0-1 之间的随机数;

ones(m,n)—生成 m 行 n 列的数组 (矩阵), 其中的元素全部为 1;

size(u)—返回数组 u 的行数和列数;

length(u)—返回数组 u 的行数和列数中最大的一个;

3.5 字符串数组

虽然 MATLAB 的功能主要是数值处理能力, 但是用户在计算中可能要经常使用字符串进行提示, 为此 MATLAB 提供了对文本的支持, 并将文本作为字符串数组进行处理。

(1) 字符串的定义和引用

字符串的定义需要用单引号“'”将文本括起来, 其中的每一个字符占用一个元素的空间, 即占用 8 个字节。

例如: >>s='How do you do!'

```
s =How do you do!
```

```
>>size(s)
```

```
ans =1     14
```

对字符串内元素的引用可以和普通数组一样进行。

例如: >>who=s(8:10)

```
who = you
```

而且可以利用已有字符串组合为新的字符串。

```
例如: >>s1='How are you!';
      >>s2='Fine,thank you!';
      >>s=[s,s2]
      s =How do you do!Fine,thank you!
```

(2) 利用字符串提示进行键盘输入

该功能是通过函数 input 来完成的。

```
例如>>x=input('Please input your ID:')
      Please input your ID:100
      x =100
```

(3) 字符串处理的常用函数

表 1 字符串处理的常用函数

函数名	描述
abs	abs(x)返回 x 的绝对值, 若 x 为字符串数组, 则返回其对应的 ascii
char	char(ascii)将 ascii 码转换为对应的字符, s=char(t1,t2...)将字符串 t1,t2...按行组成字符串矩阵 s
int2str	s=int2str(n)把整数 n 转换为字符串 s
lower	lower(s)将字符串 s 转换为小写
mat2str	将矩阵中的元素转换为字符
num2str	把数字转换为字符 num2str(10) => '10'
setstr	将 ascii 值码转换为对应的字符
str2mat	把字符串转换为一个文本矩阵
str2num	把数字字符串转换为数字
upper	uper(s)将字符串 s 转换为大写
blanks	blanks(n)返回 n 个空格的字符串
deblank	deblank(s)从字符串 s 中清除后续的空格'hello world'
ischar	ischar(s)若 s 为字符串则返回 1, 否则返回 0
isspace	isspace(s)字符串 s 中空格处返回 1,否则返回 0,形成由 0 和 1 组成的数组
strcmp	strcmp(s1,s2)若 s1,s2 相同则返回 1, 否则返回 0, 区分大小写
findstr	k=findstr(s1,s2)返回字符串 s2 在字符串 s1 中的开始位置 'world'
isletter	isletter(s)在字符串 s 中为字母处返回 1, 否则返回 0, 形成 0、1 数组
strrep	strrep(s1,s2,s3)在字符串 s1 中, 用 s3 代替 s2 'space'
strncmp	strncmp(s1,s2,m)如果 s1 和 s2 的前 m 个字符相同,则返回 1,否则返回 0
strcat	strcat(s1,s2) 在横向连接 s1、s2, 成一行向量 [s1 s2]
strvcat	strvcat(s1,s2)在竖向连接 s1、s2, 成一系列向量 [s1;s2]

利用字符串提示和字符串处理函数,可以完成字符串和数值之间的混合处理和显示。

```
例如: >>radius=input('please give the radius of a circle:') %提示输入半径
      please give the radius of a circle:10
      radius =10
      >>area=input('then the area is pi*radius^2:') %提示输入面积计算公式
      then the area is pi*radius^2:pi*radius^2
      area =314.1593
      >>s=['so the radius of circle is' num2str(radius) ',and its area is' num2str(area)];
```



```

>>disp(s) %将字符串和计算结果混合显示在屏幕上
so the radius of circle is10,and its area is314.1593
可以利用字符串处理函数，对不同的字符串进行组合、查找、比较等。例如：
>>s1='I am an undergraduate'; %定义字符串 s1
>>s2='I come from Donghua University'; %定义字符串 s2
>>a=findstr(s2,'Donghua') %从字符串 s2 中查找"Donghua"
a=13
>>b=strcat(s1,s2) %将字符串 s1 和 s2 横向连接
b=I am a undergraduateI come from Donghua University
>>c=strvcat(s1,s2) %将字符串 s1 和 s2 竖向连接
c=I am an undergraduate
  I come from Donghua University

```

3.6 结构数组

在数据处理中，有时需要将不同数据类型的数组组合在一起，便于引用，为此 MATLAB 和其它一些语言一样提供了结构数组，它类似与数据库中的一条记录，该记录中可能包含不同的数据类型，如字符型、数值型，日期型等。

(1) 结构数组的定义

可以直接用赋值语句，也可以用函数 `struct` 定义。

例如：>>student.name='Tom'; %直接用赋值语句定义

```

>>student.age=20;
>>student.sex='male';
>>student.speciality='Enviromental Science';
>>student
student =name: 'Tom'
        age: 20
        sex: 'male'
        speciality: 'Enviromental Science'

```

上述定义中的内容默认为结构数组的第一个元素。然后可以追加元素。例如：

```

>>student(2).name='Jane';
>>student.age=19;
>>student(2).age=19;
>>student(2).sex='female';
>>student(2).speciality='Enviromental Science';
student(2)
        name: 'Jane'
        age: 19
        sex: 'female'
        speciality: 'Enviromental Science'

```

`struct` 形式为：**`s=struct('field1',value1, 'field2',value2,.....)`**。例如：

```

>>student=struct('name','Tom','age',20,'sex','male','speciality','Environmental
                Science','grade',80)
student =name: 'Tom'

```

```

age: 20
sex: 'male'
speciality: 'Environmental Science'
grade: 80

```

(2) 结构数组的基本操作

可以对结构数组进行直接调用、重新赋值或增加新属性。

例如: >> student(1).speciality %直接调用

```
ans = Environmental Science
```

>> student(1).speciality='Computer Science' %重新赋值

```
student = name: 'Tom'
```

```
age: 20
```

```
sex: 'male'
```

```
speciality: 'Computer Science'
```

```
grade: 80
```

除了可以在结构数组中追加数组元素（记录）外，也可追加属性。如可在上述“student”数组中追加“student.grade”项。

>> student(1).grade=80; %追加新属性

```
>> student(1)
```

```
ans = name: 'Tom'
```

```
age: 20
```

```
sex: 'male'
```

```
speciality: 'Environmental Science'
```

```
grade: 80
```

3.7 本章练习

1. 用不同方法创建数组 $x=[0 \ 2 \ 4 \ 6 \ 8 \ 10]$, $y=[0 \ \pi/4 \ 2\pi/4 \ 3\pi/4 \ \pi \ 2\pi]$ 以及 $z=[10^{0.5} \ 10^{0.6} \ 10^{0.7} \ 10^{0.8} \ 10^{0.9}]$ 。
2. 用 $P=\text{pascal}(5)$ 命令生成一个矩阵，并（1）提取 P 的第一列元素，存入变量 X ；（2）提取 P 的第一、第三列元素，存入变量 Y ；（3）扩展 P 为 5 行 6 列矩阵，使其第 6 列全部为 1。
3. 在大气环境质量评价中，常用平均指数法，即 $I=C/C_s$ ，其中 C_s 为某种污染物的大气环境质量标准， C 为该污染物的实测浓度， I 为评价指数，评价指数越大，则表明污染越严重。由于 I 是无量纲数值，因此可以将不同评价因子（一般为 SO_2 、 NO_2 、 TSP ）的评价指数进行累加，以得出区域总的大气环境质量状况，即 $I_{\text{总}}=I_{\text{so2}}+I_{\text{no2}}+I_{\text{TSP}}$ 。今已知有三个评价区域 area1、area2、area3。其监测结果为：
 $\text{area}(1).\text{so2}=0.028$; $\text{area}(1).\text{no2}=0.094$; $\text{area}(1).\text{tsp}=0.589$
 $\text{area}(2).\text{so2}=0.034$; $\text{area}(2).\text{no2}=0.102$; $\text{area}(2).\text{tsp}=0.423$
 $\text{area}(3).\text{so2}=0.024$; $\text{area}(3).\text{no2}=0.098$; $\text{area}(3).\text{tsp}=0.570$
 相应的评价标准分别为 $\text{SO}_2=0.15$, $\text{NO}_2=0.12$, $\text{TSP}=0.30$ （上述数值单位均为 mg/m^3 ），（1）试构造上述结构数组；（2）利用结构数组进行评价计算。
4. 对表 1 中的函数进行练习。

第四章 矩阵运算

矩阵运算是 MATLAB 非常强有力的功能之一。在 MATLAB 中有几十个矩阵运算函数，使其对矩阵的处理非常方便。

4.1 矩阵的定义

矩阵定义非常简单，矩阵用方括号括起来，同一行元素之间用空格或逗号分开，不同行之间用分号隔开。

例如：>> a=[1 2 3;4 5 6;9 8 5] %定义 3 行 3 列矩阵 a

```
a =  
    1     2     3  
    4     5     6  
    9     8     5
```

4.2 矩阵的基本运算

(1) 矩阵的加减

和数组的加减一样，表示矩阵中对应元素的加减，维数不同的矩阵不能相加减。矩阵和标量的加减为标量和矩阵中元素分别相加减。

A+B A-B

(2) 矩阵的乘法：

矩阵 A*B 能够相乘必须满足 A 的列数等于 B 的行数条件。标量与矩阵相乘表示标量和矩阵中元素分别相乘。

A.*B

A*B

A./B

A/B=A*inv(B)

(3) 矩阵的除法

对矩阵 A 和 B，若用运算符 A./B 则表示矩阵 A、B 中对应元素分别相除，这不是数学意义上的矩阵相除。若在“/”前不加“.”则表示数学意义上的矩阵相除，例如 A/B，就表示矩阵 A 乘以矩阵 B 的逆矩阵。对矩阵 B 求逆的函数是 inv(B)，所以 A/B 实际就是 A*inv(B)。如对上述 a 矩阵求逆：

```
>>inv(a)  
ans =  
   -3.8333    2.3333   -0.5000  
    5.6667   -3.6667    1.0000
```

-2.1667 1.6667 -0.5000

4.3 矩阵的转置

在 MATLAB 中矩阵的转置通过运算符“'”（单引号）来完成。
例如对上述矩阵 a 的转置矩阵为：

```
>>b=a'
b =
     1     4     9
     2     5     8
     3     6     5
```

4.4 矩阵的幂指数运算

矩阵的指数运算分为按元素进行指数运算和矩阵整体的指数运算。如对上述矩阵 a：

```
>>x=exp(a) %对矩阵 a 中的每个元素 e^x 运算
x =1.0e+003 *
    0.0027    0.0074    0.0201
    0.0546    0.1484    0.4034
    8.1031    2.9810    0.1484
```

```
>>y=a.^2 %对矩阵中的元素分别乘方运算
y =
     1     4     9
    16    25    36
    81    64    25
```

```
>>a^2 %对矩阵 a 整体进行乘方运算，实际等于 a*a
ans =
    36    36    30
    78    81    72
    86    98   100
```

```
>>expm(a) %对矩阵 a 进行幂指数运算 expm(X) = V*diag(exp(diag(D)))/V
ans =1.0e+005*
    2.7117    2.9129    2.7336
    6.2348    6.6973    6.2851
    7.9283    8.5165    7.9923
```

4.5 矩阵的对数运算

矩阵的对数运算通过函数 logm 来实现。例如矩阵 b=[4 1;1 4]，则：

```
>>logm(b) %对矩阵 b 整体进行对数运算
ans =
    1.3540    0.2554
```

```

0.2554    1.3540
>>log(b) %对矩阵 b 中的元素分别进行对数运算
ans =
    1.3863    0
    0        1.3863

```

4.6 矩阵的开方运算

矩阵的开方运算通过函数 `sqrtm` 来实现。例如矩阵 `b=[4 1;1 4]`，则：

```

>>sqrtm(b) %对矩阵 b 整体进行开方运算
ans =
    1.9841    0.2520
    0.2520    1.9841
>>sqrt(b) %对矩阵 b 中的元素分别进行开方运算
ans =
    2    1
    1    2

```

4.7 矩阵运算的一些常用函数

表 2 其它一些矩阵运算的常用函数

函数名	描述
<code>[]</code>	返回空矩阵
<code>zeros</code>	<code>zeros(n)</code> 、 <code>zeros(n,m)</code> 返回全部元素是零的矩阵
<code>ones</code>	<code>ones(n)</code> 、 <code>ones(n,m)</code> 返回全部元素是 1 的矩阵
<code>eye</code>	<code>eye(n)</code> 、 <code>eye(n,m)</code> 返回对角线元素全部为 1 的矩阵
<code>rand</code>	<code>rand(n)</code> 、 <code>rand(n,m)</code> 返回全部元素在 0 和 1 之间的随机矩阵
<code>randn</code>	<code>randn(n)</code> 、 <code>randn(n,m)</code> 返回全部元素服从标准正态分布的随机矩阵
<code>magic</code>	<code>magic(n)</code> 生成 magic 矩阵
<code>pascal</code>	<code>pascal(n)</code> 生成 pascal 矩阵
<code>lu</code>	<code>[l,u,p]=lu(a)</code> 对矩阵 <code>a</code> 行 <code>l,u</code> 分解,返回上三角矩阵 <code>u</code> ,下三角矩阵 <code>l</code> 和变换矩阵 <code>p</code>
<code>eig</code>	<code>[u,v]=eig(a)</code> 求矩阵 <code>a</code> 的特征相量(<code>u</code>)和特征值(<code>v</code>)，有关系式 $a*u=u*v$

4.8 本章练习

1. 创建 3×3 的 `magic` (`A`) 和 `pascal` (`B`) 矩阵，分别存入变量 `A` 和 `B`，并计算 `A+B`、`A-2*B`、`A*A`、以及 `A/B`。
2. 创建矩阵 `a=[7 9 20;11 30 82;19 78 112]`，`b=[1 3 2;4 6 5;7 8 9]`。(1) 计算 `a-2b` 和 `a*b-I`(`I` 为单位矩阵)；(2) `a.*b` 和 `a*b` 有何不同；(3) 计算 `a.^b`、`a./b`、`a/b` 并解释它们的意义。

3. 对形如
$$\begin{cases} x_1+x_2+x_3=6 \\ 2x_1+7x_2+8x_3=40 \\ 3x_1+2x_2+x_3=10 \end{cases}$$
 的线性方程组若用一般的代换法求解，则显得比

较烦琐，但我们可以把它表示为矩阵的形式 $A*X=B$ ，其中 $A=[1 \ 1 \ 1;2 \ 7 \ 8;3 \ 2 \ 1]$ ，

$B=[6;40;10]$, $X=[x_1;x_2;x_3]$, 这样可以通过 $X=A^{-1}*B$ 求解。试对上述方程组用 MATLAB 的矩阵运算求解。

4. 上机练习表 2 中的矩阵运算函数。

第五章 MATLAB 的程序设计

以上讲述的内容都是在 MATLAB 命令窗口中可以直接执行的命令或函数，但要解决一些复杂的问题，仅仅靠这些命令是不够的。还需要将相关的命令有机地组合起来，形成程序，让计算机自动解决一些问题。MATLAB 为解释性程序设计语言，程序中的语句可以边解释，边执行。

5.1 M-file 文件

MATLAB 的程序是以 M 文件的形式存在的，M 文件分命令文件和函数文件。

(1) 命令文件

若要多次重复执行大量的 MATLAB 命令，可以将这些命令存放在一个扩展名为 .m 的文件中，并在 MATLAB 命令窗口中输入该文件名即可以执行这些命令。这就象一个批处理文件一样。

例 5_1：在 MATLAB 的 File 菜单中选取 New，然后选取其中的 M-file 选项，就出现了 M 文件的编辑窗口，在其中输入：

```
% This is a example m-file
x=0:pi/2:2*pi;
y=sin(x)
z=y+2
```

然后，以 test1.m 的名字存放在 MATLAB 目录下的 bin 子目录中，然后在命令窗口中命令提示符后面输入 test1，就可得到下面的运行结果：

```
>>test1
y = 0    1.0000    0.0000   -1.0000    0.0000
z = 2.0000  3.0000    2.0000    1.0000    2.0000
```

(2) 函数文件

利用函数文件可以将一些经常执行的运算程序编辑为函数的形式，在使用时，直接调入该函数名即可，大大减少了重复编写程序代码，精简了程序。函数文件和命令文件的显著差别就是函数文件可带输入输出参数。下面举一个简单的函数文件的例子：

例 5_2：瞬时向河流投放某种示踪剂 ($K=0$)，投放量为 $M=10\text{kg}$ ，纵向弥散系数 $D_x=50\text{m}^2/\text{s}$ ，平均流速 $u_x=0.5\text{m/s}$ ，河流横断面 $A=20\text{m}^2$ ，根据式

$$C(x,t)=\frac{M}{A\sqrt{4D_x\pi}}\exp\left(-\frac{(x-u_x t)^2}{4D_x t}\right)\cdot\exp(-Kt)，\text{计算 } 500\text{m} \text{ 处的示踪剂浓度分布，并求其}$$

最大值。

上述例子的函数文件如下：

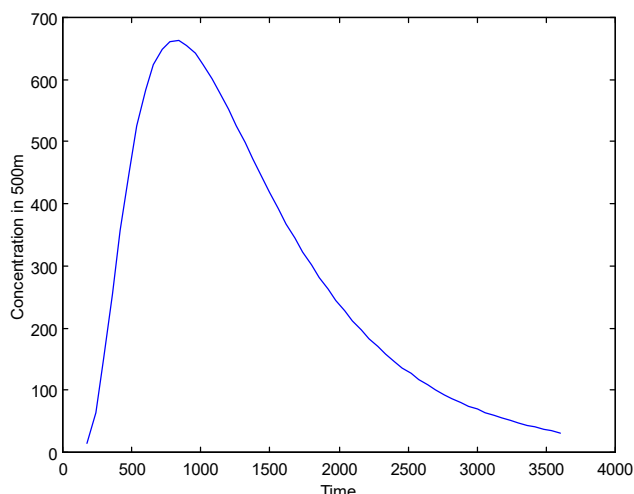
```
function output1=con1(x)
output1=0.0;
m=10000000 %污染物投放量，单位：mg
u=0.5; %水流速度，单位：m/s
a=20; %断面面积，单位：m2
```



```

d=50;%弥散系数, 单位: m2/s
k=0;%降解系数, 单位: 1/d
i=1
for t=180:60:3600;
output1(i)=(m/(a*sqrt(4*d*pi*t)))*e
xp(-((x-u*t)^2)/(4*d*t))*exp(-k*t);
i=i+1
end
t=180:60:3600;
plot(t,output1)
xlabel('Time')
ylabel('Concentration in 500m')

```



其中 con1 是函数名, output1 是输出参数, for.....end 是语句循环操作。Plot 是绘图语句, xlabel 和 ylabel 分别为图形的 x 轴和 y 轴名称。只要在命令窗口中输入 >>con1(500) 即可计算处 500m 处从 t=180s 到 3600s 的浓度分布过程线 (如上图), 可见随时间的变化, 500m 处的浓度达到峰值后有迅速减低。并在 1000S 左右, 500m 处的污染物浓度达到峰值。

由上例可见函数文件的一般格式为:

function [参数输出表]=函数名 (参数输入表)

注释行

函数体

引用定义的函数时, 只要直接输入函数名 (输入参数) 即可。

(3) 变量和常量

MATLAB 中变量必须以字母开头, 后面可以是字母、数字或下划线。变量分局部变量和全局变量。

局部变量的作用域仅仅局限在本函数, 只能被本函数调用。例如在函数文件中定义的变量, 一旦函数执行完毕, 该变量的内存自动释放。例如上述计算污染物浓度的函数, 在我们调用完后, 如果在命令窗口中输入 output1 则会显示 “???Undefined function or variable ‘output1’”。

与局部变量不同, 全局变量一经定义就可以被其它的函数调用。例如我们在命令窗口中输入 >>global output1, 然后运行上述函数 con1(x), 运行完成后再次在命令窗口中输入 output1, 就可以显示出该数组变量的值。

常量, 这是 MATLAB 为特定的常数保留的一些名称, 主要有:

pi 表示圆周率(3.1415926); i 或 j 为基本虚数单位; eps 为系统浮点精确度; realmax 为系统最大的数值; realmin 为系统能表示的最小数值; inf 表示无穷大。我们可以在命令窗口中直接输入上述常量名称来观察它们的值。

5.2 程序流程语句

(1) 赋值语句

分为直接赋值语句和函数调用语句其形式分别为:

直接赋值语句基本格式为：变量名=赋值表达式

函数调用语句的基本格式为：[返回变量列表]=函数名（输入参数变量）

例5_3：对例2定义的函数稍做修改如下：

function [t,c]=con2(x, m) %将距离和投放量作为输入参数

c=0.0;

u=0.5; %水流速度，单位:m/s

a=20; %断面面积，单位: m2

d=50; %弥散系数，单位: m2/s

k=0; %降解系数，单位: 1/d

i=1

for t=180:60:3600;

c(i)=(m/(a*sqrt(4*d*pi*t)))*exp(-((x*u*t)^2)/(4*d*t))*exp(-k*t);

i=i+1

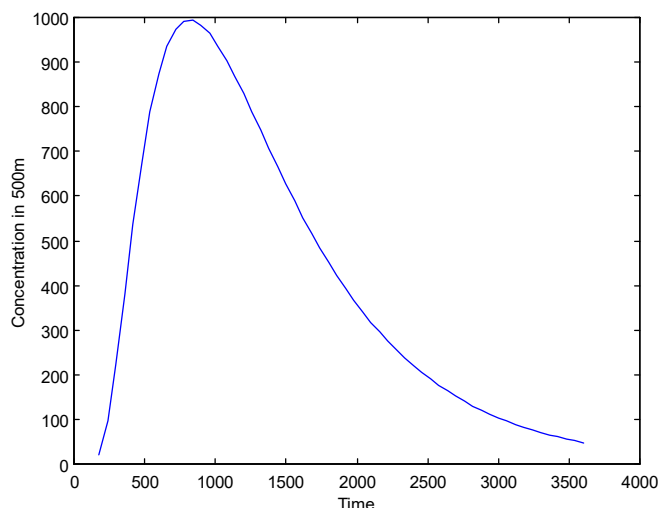
end

t=180:60:3600;

plot(t,c)

xlabel('Time')

ylabel('Concentration in 500m')



这样，我们在应用时直接输入>>[t,c]=con2(500, 15000000)，就可得到当投放量为 15kg 时，500m 处不同时刻及其对应的浓度值（上图）。

(2) 循环语句：

MATLAB 中的循环有两种，分别为 for 循环和 while 循环。

for...end 循环：一般用于已经确定循环次数的情况，其格式：

for 变量名=表达式 %一般以 i=s1:s2:s3 形式表示，当 i>s3 时循环结束
循环体语句

end

例 5_4:

x=zeros(1,10);

for i=1:1:10;

x(i)=i^2;

end

上述语句最先产生一个 1 行 10 列的零矩阵，然后通过循环，对该矩阵进行赋值。

当然和其它计算机语言一样，for 循环可以多层嵌套使用。

例 5_5：下述命令产生一个 5 行 5 列的矩阵

for i=1:1:5;

for j=1:1:5;

a(i,j)=1/(i+j);

end

end

while...end 循环：一般用于事先不能确定循环次数的情况，其格式：
while 条件表达式 %以逻辑判断语句为条件，逻辑判断为假时退出循环。
 循环体语句
end

例 5_6:

```
x=zeros(1,10);
i=1;
while i<=10;
    x(i)=i^2;
    i=i+1;
end
```

在 while...end 循环循环体内可以加入 break 语句，当在循环体内执行到此语句时自动结束循环。

(3) 条件语句

一般有如下几种形式。

If...end 语句，其基本格式为：

if 条件表达式 %若条件为真（非零），则执行条件块语句组，否则跳出条件块语句组
end

下面的例子首先输入判断数，然后根据判断数 x 判断生成的随机数是否大于该判断数。

例 5_7:

```
>>x=input('please input number to judge x=');
please input number to judge x=0.5
>>y=rand(1,1);
>>if y>x
>>s=['given random number is greater than' num2str(x)]
>>else s=['given random number is less than' num2str(x)]
>>end
disp(s)
```

given random number is greater than 0.5

当判断条件为 2 个时应用 **If...else...end** 语句，其基本格式为：

if 条件表达式 %条件为真(非零)，执行语句组 1，否则执行语句组 2
 条件块语句组 1
else
 条件块语句组 2
end

当判断条件大于 2 个时，**If...else...end** 语句采用如下格式：

```

if 条件表达式 1
    条件块语句组 1
elseif 条件表达式 2
    条件块语句组 2
...
else
    条件块语句组 n
End

```

例 5_8:

```

x=input('x=');
y=input('y=');
if x==y
    flag=1;
elseif x>y
    flag=2;
elseif x<y
    flag=3;
end
flag

```

上述例子首先对输入的 x、y 进行大小比较，然后根据情况给出 flag 的值。

(4) 开
关
语
句

其基本格式为

```

switch 开关表达式
    case 表达式 1
        语句组 1
    case {表达式 1,表达式 2,表达式 3.....}
        语句组 2
    ...
    otherwise
        语句组 n
end

```

当开关表达式的值等于某个 case 后面的条件时，程序将转移到该语句组去执行。执行完成后直接跳出循环体，继续执行 end 后面的语句。不再对其它情况进行判断。若要开关表达式满足某个表达式之一时执行某一程序段，则应把这样的开关表达式全部用 {} 括起来，中间以逗号分隔。当所有 case 的情况均为假（等于零）时，则执行 otherwise 后面的语句组 n。

例 5_9:

```

name='JOHN'
switch lower(name)

```

```

case {'john','tom','jack'}
disp('They are all male')
case {'jane','mary','honey'}
disp('They are all female')
case 'henry'
disp('He will come back tomorrow')
otherwise
disp('It is not our student')
end

```

程序的输出结果是 **They are all male。**

5.3 break 和 continue 语句

Matlab 允许 if 语句在循环变量未达到终点之前“跳出”循环。用户可以使用 break 语句命令终止循环达到目的。

例 5_10:

```

for k=1:10
    x=50-k^2;
    if x<0
        break
    end
    y=sqrt(x)
end

```

% The program execution jumps to here

% if the break command is executed

continue 用于将控制传递给下一个迭代，

例 5_11: 以下代码使用了 continu 语句避免计算一个负数的对数。

```

x=[10,1000,-10,100]
y=nan*x;
for k=1:length(x)
    if x(k)<0
        continue
    end
    y(k)=log10(x(k))
end
y

```

其结果是: y=1、3、NaN、2

5.4 错误调试

(1) 使用断点

大部分调试会话都是从设置断点开始的。断点在一个指定处停止执行 M-文件，并且其允许用户在恢复执行前查看或者修改函数工作空间中的值。要设置一个断点，可以将光标放在文本行之中，让后再单击工具栏中的断点图标或者从 Debug 菜单下选择 set/clear breakpoint

菜单项。用户还可以通过右击文本行弹出快捷菜单设置一个断点。文本行左侧红色圆圈指出的是断点设置在这一行。如果选中作为断点的文本行是不可执行的语句，那么就在下一个可执行的命令行处设置断点。

(2) 查找故障

编辑器对于纠正运行时错误非常有用，这是因为他允许用户访问函数工作空间并且检查或修改其中所包含的值。现在，下例将给出单步执行一个示例调试会话。首先创建一个名为 `fun1.m` 的 M 文件，他接受一个输入矢量并且返回矢量中那些大于平均值的数量值。这个文件调用一个名为 `fun2.m` 的 M 文件，`fun2.m` 文件在给定矢量以及其平均值的情况下，计算矢量中值大于平均值的数量。

```
function y=fun1(x)
    avg=sum(x)/length(x);
    y=fun2(avg,x);
```

按以上所示的代码创建 `fun1.m` 文件，并且认为的设置一个故障。让后创建文件 `fun2.m`，其内容如下：

```
function above=fun2(x,avg)
    above=length(find(x>avg));
```

使用一个可以通过手算的简单测试示例。例如，使用矢量 `v=[1,2,3,4,10]`。他的平均值是 4，其中包含一个大于平均值的 10。现在调用函数 `fun1` 来对它进行测试。

```
>>above=fun1([1,2,3,4,10])
above=
3
```

但是正确的答案应该是 1，因此至少有一个函数 `fun1` 和 `fun2` 中的运行有错误。在开始调试时，用户并不确定错误在哪里。插入一个断点的合理位置应该是在 `fun1.m` 中平均值计算的后面。打开 `fun1.m` 的编辑器，并且通过使用工具栏上的 `set breakpoint` 按钮在第三行上 (`y=fun2(avg,x)`) 设置一个断点。行的左边指出了行号。注意：要查看变量 `avg` 的值，用户必须计算 `avg` 值后的任何一行中设置断点。

要运行程序到断点处并检查某一个，首先要通过输入 `fun1([1,2,3,4,10])` 从命令窗口执行该函数。当 M 文件执行到一个断点处暂停时，文本左边的绿色箭头就会指示出将要执行的下一个命令行。通过高亮显示变量名，然后右击弹出快捷菜单，选择其中的 `Evaluate Selection` 菜单项来对 `avg` 的值进行检查，此时用户可看到命令窗口中显示出 `avg=4`。由于这个 `avg` 的值是正确的值，所以错误必定存在于第三行中对 `fun2` 函数的调用，或者存在于 `fun2.m` 文件中。

通过将光标放在命令行上并同时单击 `clear breakpoint` 按钮，可以清除 `fun1.m` 中第三行上的断点。通过单击调试工具栏上的按钮 `continue` 按钮，可以继续执行 M 文件。打开 `fun2.m` 文件，在第 2 行处设置一个断点，查看程序是否将 `x` 和 `avg` 的正确值传递给了函数。在命令窗口中，输入 `above=fun1([1,2,3,4,10])`。高亮显示第 2 行表达式中的变量 `x`：`above=length(find(x>avg));`，右击，从弹出的快捷菜单选择 `evaluate selection` 菜单项，用户就将在命令窗口中看到 `x=4`。这个值是不正确的结果，这是因为 `x` 应该是 `[1,2,3,4,10]`。现在，用相同的方法了解第 2 行中变量 `avg` 的值。用户应该可以在命令窗口中看到 `avg=[1,2,3,4,10]`。这个值是不正确的结果。这是因为 `avg` 应该=4。

所以，在 fun1.m 中第 3 行的函数调用中颠倒了 x 和 avg 的值。这一行正确的代码应该是 y=fun2(x,avg)。经过代码修改后保存，再运行函数，就可以得到正确的结果。

5.5 本章练习

1、别用 for 循环和 while 循环编写程序，计算 $L = \sum_{n=1}^{100} (2n-1)^2$ 。

2、用 MATLAB 语言编写的分段函数 $f(x) = \begin{cases} 0, & x \leq 5 \\ (x-5)/10, & 3 \leq x \leq 10 \\ 1, & x > 10 \end{cases}$ ，并计算 f(7) 的值。

3、在许多环境空气质量预测和模拟模型中，箱式环境空气质量模型比较简单。其基本假设是：在模拟环境空气中污染物浓度时，可把所研究的空间范围简化为一个尺寸固定的“箱子”，箱子的高度就是从地面计算的混合层高度，箱子的平面尺寸是所研究的区域面积，污染物浓度在箱子内处处相等。该模型（图示）假定箱子的高、长、宽分别为 h (m)、 l (m)、 b (m)，箱子内污染物浓度为 C (g/m³)，污染物本底浓度为 C_0 (g/m³)，污染源源强为 Q (g/m²·s)，污染物衰减速度常数为 K (s⁻¹)，平均风速为 u (m/s)，则箱子内污染物在 Δt 时段内质量的变化为 $\Delta C \cdot lhb$ 。

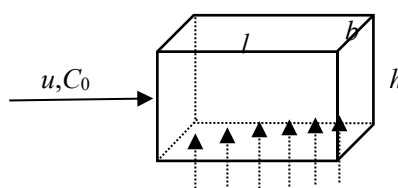


图 单箱模型示意图

当 $t \rightarrow \infty$ 时，箱内污染物平衡浓度 C_p 为：
$$C_p = C_0 + \frac{Q/h - C_0 K}{u/l + K}$$

(1) 试利用上述方程编写单箱模型的平衡浓度计算函数 c=singlebox(C_0, Q, h, l, b, u, k)；

(2) 调用上述函数利用下面资料计算平衡浓度：某山谷地区要建一工业区，其混合层厚度 $h=750m$ ，该地区长 $l=50m$ ，宽 $b=5m$ ，风速 $u=2m/s$ ， $k=0$ （即不考虑 SO_2 衰减）， SO_2 本底浓度 $C_0=0.011mg/L$ 。该工业区建成后计划用煤 $8000t/d$ ，煤的含硫量为 3%， SO_2 的转化率为 85%，用单箱模型估计工业区建成后该地 SO_2 的浓度。（ $Q = \frac{(8000 \times 3\% \times 85\% \times 64/32) \times 10^6}{50 \times 5 \times 60 \times 24 \times 60} = 18.89$ g/m²·s， $C=0.641mg/L$ ）。

$$(dp/dt)lbh = ubh(p_b - p) + lbQ - kplbh$$

$$C = C_0 + \frac{Q/h - C_0 K}{u/l + K} \left[1 - \exp\left(-\left(\frac{u}{l} + K\right)t\right) \right]$$

(3) 然后在循环中调用该函数，分别计算该地区燃煤 2000t/d、4000t/d、6000t/d、8000t/d、10000t/d、12000t/d 时区域 SO_2 的浓度（其它参数不变）。

第六章 MATLAB 绘图

MATLAB 可以将计算得到的数据根据命令要求绘制为二维、三维以至四维图形，这为数据提供了生动的表现形式。需要注意的是 MATLAB 的图形窗口和命令窗口都是独立的窗口。对图形窗口中图形的引用可使用图形窗口菜单中的 `copy figure` 选项。

6.1 二维平面绘图

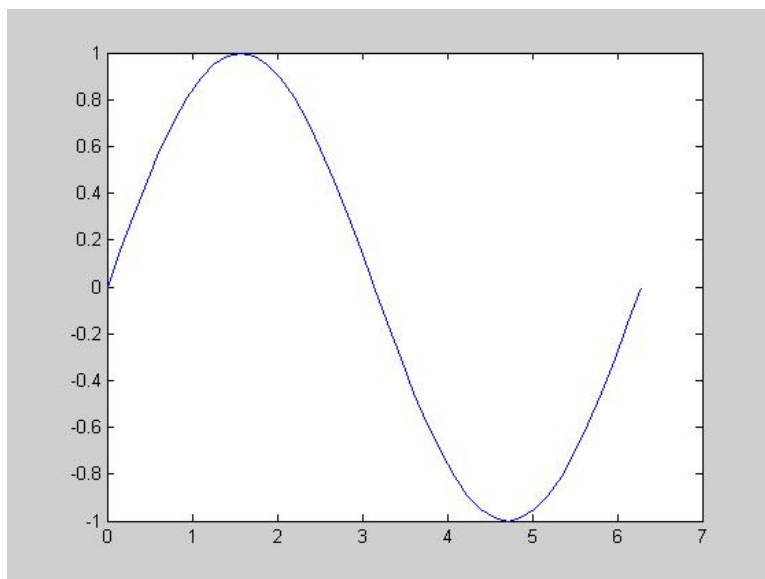
(1) 基本二维绘图语句

`plot` 是二维绘图的基本语句，其它二维绘图函数大多是以 `plot` 为基础构造的。`plot` 的基本格式为：**`plot(x,y)`**。其中 x 为平面直角坐标系中的水平轴数值， y 为平面直角坐标系中的垂直轴数值。

例 6_1:

```
x=0:pi/20:2*pi;  
y=sin(x);  
plot(x,y)
```

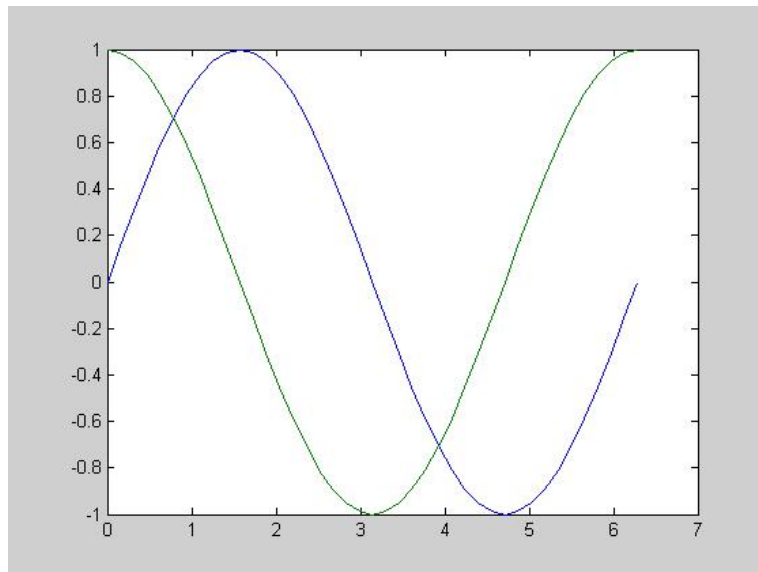
我们立即可以得到下面的图形：



若要在同一坐标系内绘制多个图形，只要将多对数据依次放入 `plot` 函数内即可。例 6_2:

```
x=0:pi/20:2*pi;  
y=sin(x);  
y1=sin(x);  
y2=cos(x);  
plot(x,y1,x,y2)
```

我们立即可以得到下面的图形。



如果 x 和 y 都是矩阵，则以其对应列为对应坐标绘制图形。

我们注意到上面的图形都是实线，这样如果同一坐标系内图形较多的话，很难区分出不同的图形来，其实上述命令只是 `plot` 最简单的操作形式，`plot` 命令含有许多可选项，能够补充上面的不足。

`plot(x,y,s)` 命令可以获得不同颜色和不同符号表示的图形， s 是如下的三种字符
 颜色字符: `b` blue; `g` green; `r` red; `c` cyan; `m` magenta; `y` yellow; `k` black。
 图形点: `.` point; `o` circle; `x` x-mark; `+` plus; `*` star; `s` square; `d` diamond;
`v` triangle(down); `^` triangle (up); `<` triangle (left); `>` triangle (right); `p` pentagram; `h` hexagram;

线型: `-` solid; `:` dotted; `-.` Dashdot; `--` dashed;

例如 `plot(X,Y,'c+:')` 绘制出 cyan 色 dotted 线型的图形并在数据点以“+”号表示。

`plot(X,Y,'bd')` 绘制出 blue 色图形，并在每个数据点加上菱形符号 (diamond)。

关于不同符号、点型、线型可以上机练习。

例 6_3:

```
x=0:pi/20:2*pi;
```

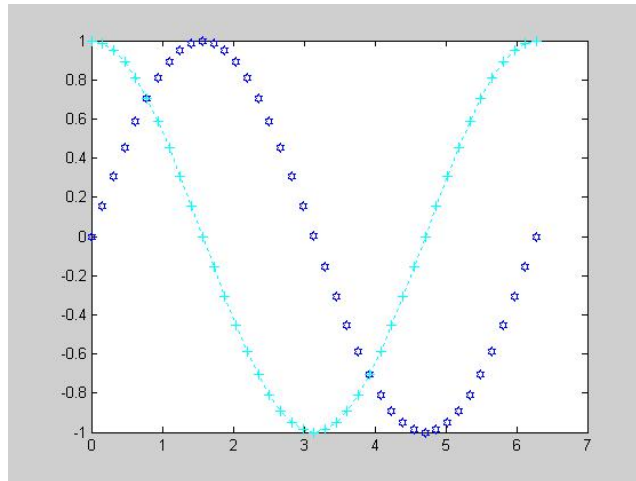
```
y1=sin(x);
```

```
y2=cos(x);
```

```
plot(x,y1,'h',x,y2,'c+:')
```

可得到下面的图形。

```
Plot(x,y,'ro')
```



(2) 其它二维绘图语句

除 plot 函数外，MATLAB 还提供了一些绘图函数。

表 7 其它常用二维绘图函数

函数名	描述	函数名	描述
loglog	x、y 轴均为对常用数刻度	hist	绘制累计图
semilogx	y 轴为对常用数刻度	rose	绘制极坐标累计图
semilogy	x 轴为对常用数刻度	stairs	绘制阶梯图
plotyy	在图形的左右两侧分别建立 y 轴	stem	绘制针状图
bar	绘制长条图	fill	绘制实心图
errorbar	图形加上误差范围	feather	绘制羽毛图
fplot	绘制比较精确的函数图形	compass	绘制罗盘图
polar	绘制极坐标图形	quiver	绘制向量场图

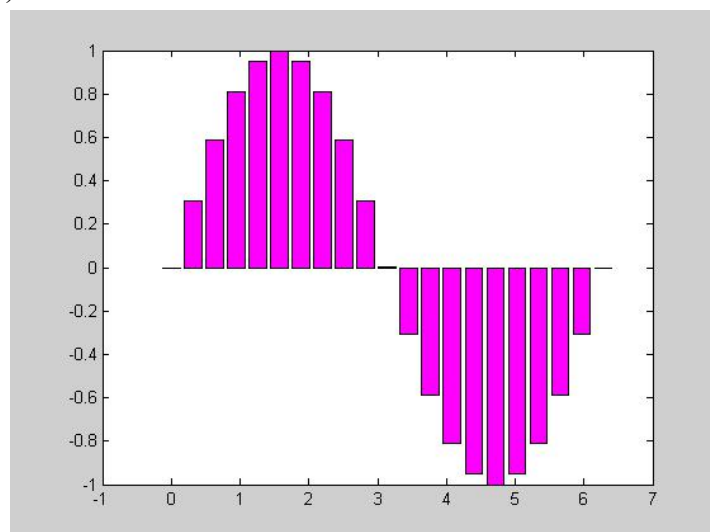
关于上述函数的应用格式与 plot 函数类似，同学可以上机练习，当然也可以在命令窗口中直接输入“help 函数名”来获得该函数的帮助。长期练习和使用后，就可以熟练使用这些函数绘制出各种精美的图形。

下面举一些应用上述函数绘图例子。

例 6_4:

```
x=0:pi/10:2*pi;
```

```
bar(x,sin(x),'m') %下面图形
```



另外对 bar 函数可以有两种使用方式，一种是 group，一种是 stack 格式，这可以通过下面的例子说明。

```
>> x=randn(3,6)
```

```
x =
```

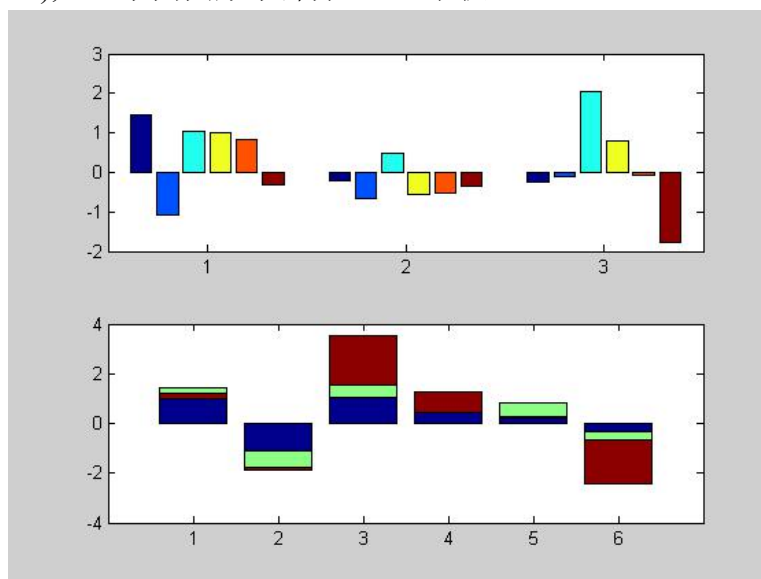
```
    1.4573    -1.0926     1.0522     1.0027     0.8157    -0.3321
   -0.2280    -0.6718     0.4720    -0.5523    -0.5312    -0.3430
   -0.2396    -0.1090     2.0289     0.8026    -0.0888    -1.7586
```

```
>> subplot(2,1,1);
```

```
>> bar(x,'group'); %下面图形上部分 group 分组型
```

```
>> subplot(2,1,2);
```

```
>> bar(x,'stack'); %下面图形下部分 stack 堆积型。
```



errorbar 函数的基本调用形式为：**errorbar(x,y,e)**，x,y 表示横、纵坐标，e 表示误差，误差范围是[y-e,y+e]。

例如，下面命令绘制以标准差为误差范围的误差条图：

```
>> x=0:pi/20:pi
```

```
>> y=sin(x);
```

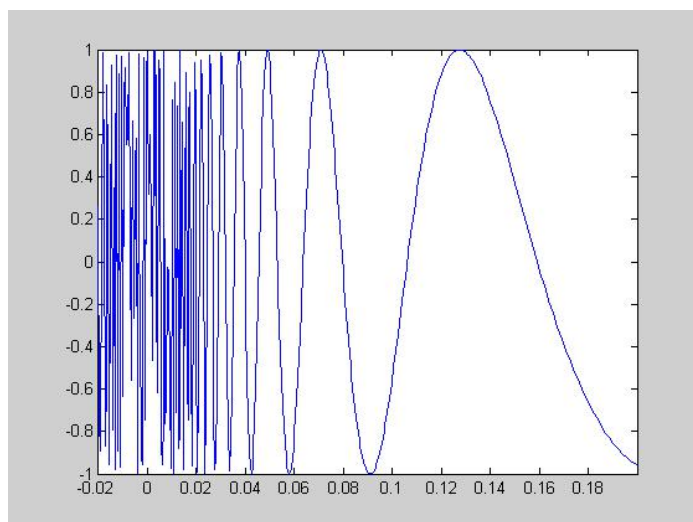
```
>> e=std(y)*ones(size(x)); %std 为求标准差函数
```

```
>> errorbar(x,y,e) %关于图形的形式可上机练习
```

fplot 函数用于在特定取值范围内进行精确绘图，例如：

```
>>fplot('sin(1/x)',[-0.02,0.2]) %下图
```

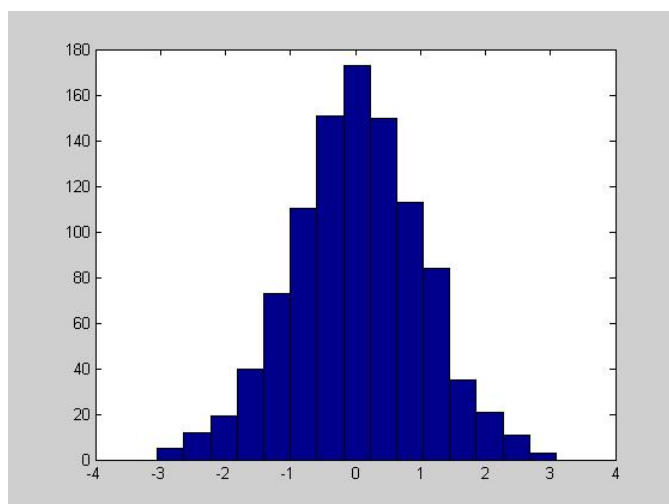
因为在 0 附近，随 x 的变化，函数 $\sin(1/x)$ 的变化幅度比较大，如按一般绘图输出，则在 0 附近曲线有较大抖动，上述命令可以在 [0.04, 0.2] 的绘制范围内精确绘制出函数 $\sin(1/x)$ 的图形。



函数 `hist` 用于显示大量数据的分布情况和统计特性。例如可由上述函数验证 `randn` 产生的数据是否服从正态分布：

```
>> x=randn(1,1000); %产生 1000 个正态分布随机数
>> hist(x,15) %15 代表长条个数
```

从下面产生的图形明显可见所产生的随机数是服从正态分布的。



(3) 二维图形的标注语句

我们注意到上面的图形没有标题，坐标轴也没有标题，这对图形的表示是很不方便的。为此 MATLAB 提供了图形标注语句。

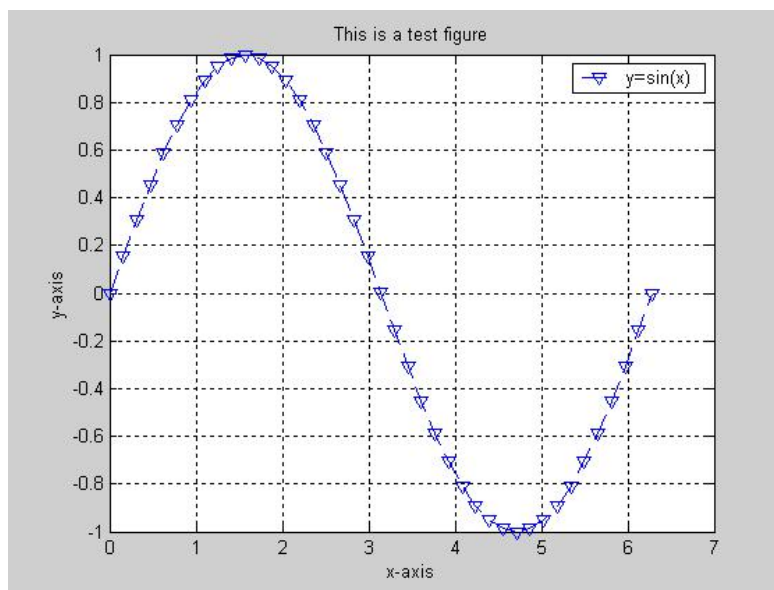
表 8 二维图形的标注语句

函数名	描述
<code>axis([xmin xmax ymin ymax])</code>	调整 x 轴和 y 轴的刻度范围
<code>xlabel('x-title')</code>	给 x 轴加上标题
<code>ylabel('y-title')</code>	给 y 轴加上标题
<code>title('chart-title')</code>	给图形加上标题
<code>legend('legend name')</code>	给图形加上图例
<code>grid on</code>	打开图形网格
<code>grid off</code>	关闭图形网格

例 6_5:

```
>> x=0:pi/20:2*pi;
>> y=sin(x);
>> plot(x,y,'bv--')
>> xlabel('x-axis');
>> ylabel('y-axis');
>> title('This is a test figure');
>> legend('y=sin(x)')
>> grid on
```

这样，我们可以得到下边显示的图形。



(4) 二维图形上点的标注

对二维图形上点的标注是通过命令 `text` 实现的。其基本格式为：**`text(x,y,'字符串')`**。其中 `x,y` 表示标注的位置，字符串是要标注的内容。

例如在上述命令组后面加入 `>>text(0,0,'This is zero point')` 可以在上述图形的零点位置显示 `This is zero point`。

(5) 其它二维图形处理语句

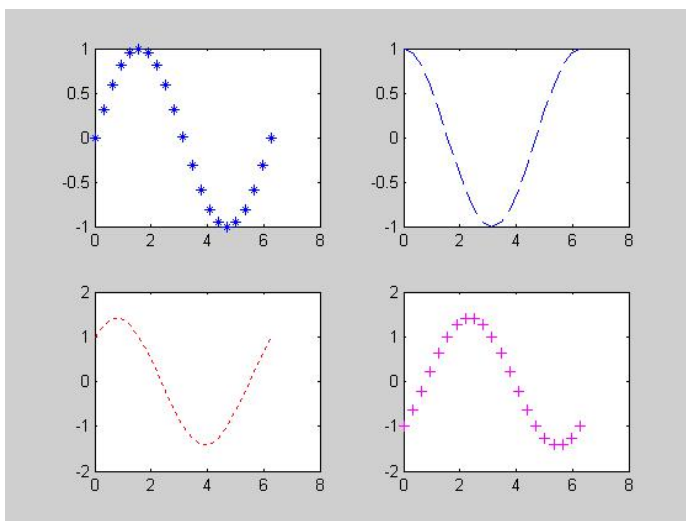
表 9 其它二维图形处理语句

函数名	描述
Hold on	保持当前的图形，后续的图形命令加入到当前的图形中
Hold off	后续命令产生的新图形将原来图形覆盖
subplot(n,m,k)	将命令窗口划分为 $n \times m$ 个图形块，新出的图形放置于第 k 个图形块中

例 6_6:

```
>> x=0:pi/10:2*pi;
>> subplot(2,2,1);
>> plot(x,sin(x),'*');
>> subplot(2,2,2);
>> plot(x,cos(x),'--');
>> subplot(2,2,3);
>> plot(x,sin(x)+cos(x),'r:');
>> subplot(2,2,4);
```

```
>> plot(x,sin(x)-cos(x),'mC+');
>> plot(x,sin(x)-cos(x),'m+');
```



(6) 图形交互处理

图形交互处理可以使用户用鼠标获得图形上点。这种功能通过 `ginput` 函数来实现。其基本格式：`[x,y]=ginput(n)`。其中 `n` 为选取点的数目，`x` 和 `y` 分别存贮用户选取 `n` 个点的横坐标和纵坐标。例如下述程序段可以从图形 `y=sin(x)` 上获取三个点，并记录在 `x,y` 中。

例 6_7:

```
>> x=0:pi/10:2*pi;
>> plot(x,2*sin(x),'*-');
>> [x,y]=ginput(3)
x = 2.3468  3.2823  5.7661
y = 1.4094 -0.2047 -0.9766
```

若要在利用鼠标点出的点绘制一条折线，则可在上述命令的基础上，输入：

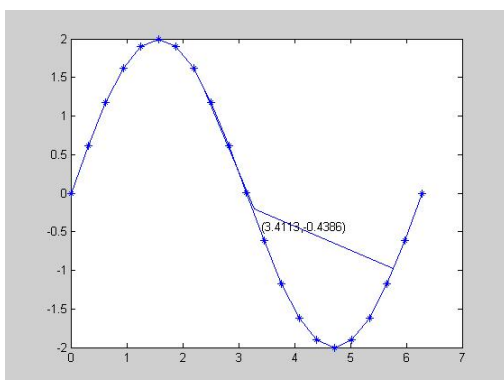
```
>> hold on
```

```
>> plot(x,y)
```

若要在鼠标点取的位置输入相应的坐标值，则可以用如下命令：

```
[x,y]=ginput(1);
>> str=['(num2str(x),num2str(y))']
>> text(x,y,str)
```

另外，若要交互地用鼠标将文本放置在图形上，可以使用 `gtext('字符串')` 命令。同学可以自己加以练习。



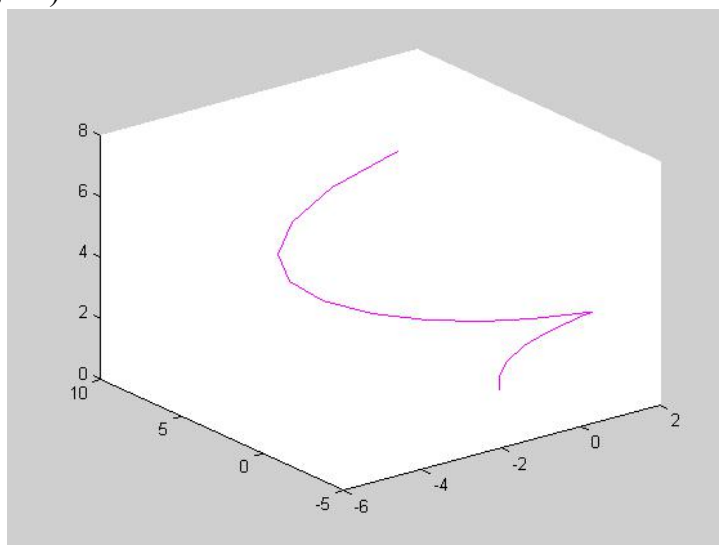
6.2 空间三维绘图

(1) 三维曲线图的绘制

绘制三维曲线图的基本命令是 `plot3`，其调用格式为：**`plot3(x,y,z,选项)`**。`x,y,z` 分别表示空间三维内的 3 个坐标向量，选项可以用来定义线型、颜色、标记等。

例 6_8:

```
>> x=linspace(0,2*pi,20);
>> y=x.*sin(x);
>> z=x.*cos(x);
>> plot3(y,z,x,'m-')
```



(2) 三维曲面网线图的绘制

三维曲面网线图可以用 `mesh` 函数生成。其基本格式为：**`mesh(x,y,z)`**。其中 `x,y,z` 分别为三个控制三维坐标矩阵。图形中的每个点是通过 (x_{ij}, y_{ij}, z_{ij}) 确定的，点与点之间靠线段连接。使用 `mesh` 函数时，一般要靠 `meshgrid` 函数生成网线节点矩阵（当然也可以直接指定网线节点矩阵）。

```
[x,y]=meshgrid(xscope,yscope)
```

例 6_9:

```
>> [x,y]=meshgrid(0:2,0:2)
```

x =

0	1	2
0	1	2
0	1	2

y =

0	0	0
1	1	1
2	2	2

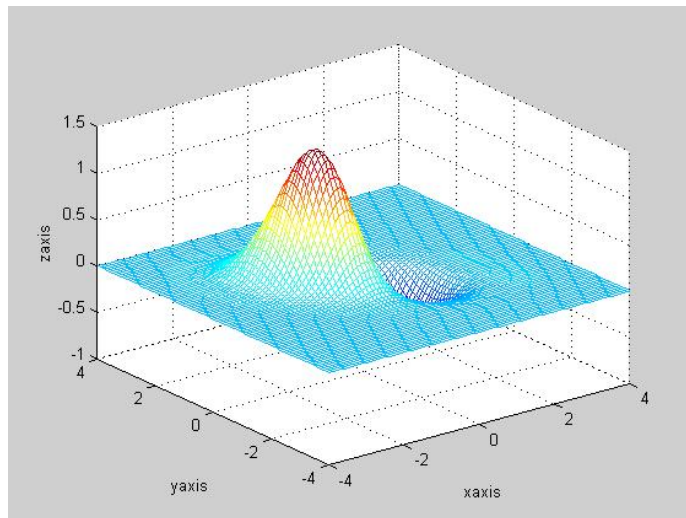
可见上述名，上述命令的作用是产生一个横坐标开始于 0 结束于 2；纵坐标也始于 0 结束于 2 的网格分割，产生的坐标对分别为 (0, 0)、(0, 1)、(0, 2)、(1, 0)、(1, 1)、(1, 2)、(2, 0)、(2, 1)、(2, 2)。

下面给出一例子：绘制二元函数 $z=f(x,y)=(x^2-2*x)\exp(-x^2-y^2-xy)$ 的三维表面图形。

例 6_10:

```
>> [x,y]=meshgrid(-4:0.1:4,-4:0.1:4);  
>> z=(x.^2-2*x).*exp(-x.^2-y.^2-x.*y);  
>> mesh(x,y,z)  
>> xlabel('xaxis')  
>> ylabel('yaxis')  
>> zlabel('zaxis')
```

产生图形如下面所示。



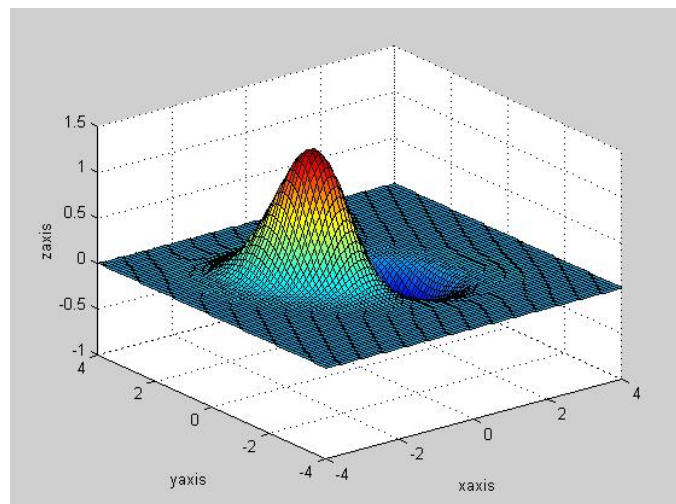
(3) 三维表面图的绘制

这是通过函数 `surf(x,y,z)` 来实现的，其参数定义与 `mesh` 函数相同。`Surf` 函数绘出的是表面图，其图形表面被覆盖。

例 6_11:

```
>> [x,y]=meshgrid(-4:0.1:4,-4:0.1:4);  
>> z=(x.^2-2*x).*exp(-x.^2-y.^2-x.*y);  
>> surf(x,y,z)  
>> xlabel('xaxis')  
>> ylabel('yaxis')  
>> zlabel('zaxis')
```

可见，其图形表面被覆盖。



(4) 等值线图的绘制:

其基本调用格式为: **c=contour(x,y,z)**。

其中 c 为等值线矩阵, 其余参数同前。

例绘制函数 $z=\exp(-x^2-y^2-xy)$ 的等值线。

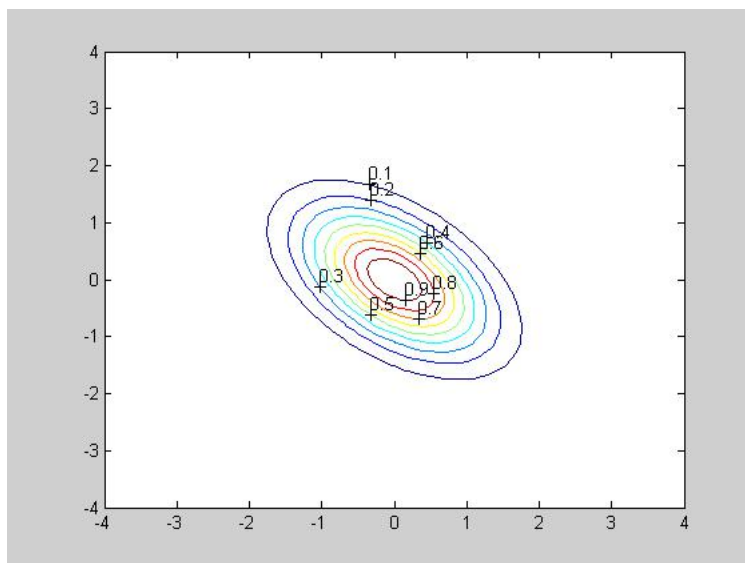
输入命令如下, 例 6_12:

```
>>[x,y]=meshgrid(-4:0.1:4,-4:0.1:4);
```

```
>>z=exp(-x.^2-y.^2-x.*y);
```

```
>>c=contour(x,y,z);%生成等值线
```

```
>>clabel(c)%用等值线矩阵标记等值线, 得到下面的图形。
```



另外, 关于 mesh 函数和 surf 函数还有一些特殊用法。例如用函数 meshc 可同时绘制出曲面网格线和等高线, meshz 函数可同时绘制出曲面和零平面图。同样函数 surfc 可同时绘制出表面图和等高线, surf1 函数可绘制出带光照效果的表面图。读者可对上述函数进行练习。

当然 MATLAB 中的绘图函数远远不止这些, 希望读者通过这些绘图函数的应用, 可以触类旁通地使用其它函数, 这里只是起到抛砖引玉的作用。

6.3 本章练习

1、选择合适步长绘制下列函数的图形：（1） $\ln\frac{1-x}{1+x}, x \in (-1,1)$ ；（2） $\sqrt{\cos(x)}, x \in [-\pi/2,\pi/2]$ 。

2、绘制函数 $z=\frac{1}{2\pi}\exp\left\{-\frac{1}{2}(x^2+y^2)\right\}$ 的三维表面网格图 mesh，表面图 surf 等值线图 countour，根据该曲面，练习函数 surf1、meshc、meshz、surfc 的用法。

3、在 $15km^2$ 的空间范围内，在一定空间坐标的取样点测试沉积物中的某重金属元素含量 (mg/kg) 如下表所示。表 10

序号	坐标 $x(m)$	坐标 $y(m)$	含量 $u(mg/kg)$	序号	坐标 $x(m)$	坐标 $y(m)$	含量 $u(mg/kg)$
1	2	3	1.9	9	8	11	1.3
2	2	10	2.3	10	10	8	1.2
3	2	13	1.1	11	11	13	1.4
4	4	1	2.6	12	12	3	1.7
5	5	8	2.2	13	12	6	1.8
6	5	14	1.8	14	12	10	1.2
7	7	3	3.5	15	15	13	1.0
8	7	6	3.1				

经拟合，发现元素含量数值(z)及坐标位置(x,y)符合如下三次趋势面方程。

$z=-1.288+1.133x+0.968y-0.077x^2-0.124xy-0.105y^2+0.00036x^3+0.005x^2y+0.0034xy^2+0.0035y^3$

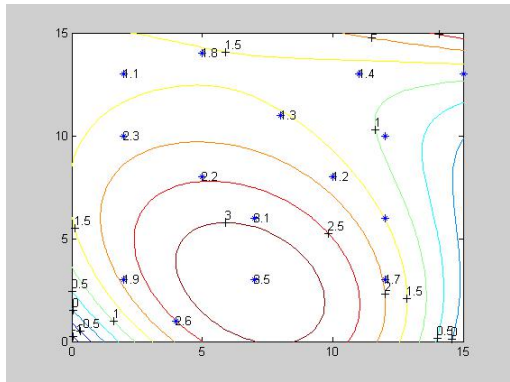
（1）试在 $x \in [0,15], y \in [0, 15]$ 的平面内选择合适步长绘制上述曲面的等值线；

（2）将上表中的点及其对应的重金属元素含量值加入到上面绘制的等值线中。

即最后得出如右图所示的图形（其中*号旁数字为上表中的数字，+号旁的数字则表示等值线的值）。

提示：考虑使用下面的命令与函数

```
hold on, plot(x,y,'*')
d=1.9 2.3 1.10 2.60 2.20 1.80 3.50 3.10 1.30
1.20 1.40 1.70 1.80 1.20 1.00
for i=1:1:15;
text(a(i),b(i),num2str(d(i)));
hold on
end
```



第七章 MATLAB 的数值分析功能

在科学研究中。经常需要进行大量的数学运算，这些运算一般来说难以通过手工的方法精确和快速的进行，需要借助计算机进行运算。在一般的计算机语言中，没有内置专门的数学计算函数，对许多数学计算需要单独编制程序完成，而在 MATLAB 中则有大量的内置数学运算函数，通过调用这些函数可以快捷地实现运算过程。

7.1 函数、插值和曲线拟合分析

(1) 多项式

在数据分析和科学计算的很多领域都牵涉到多项式的使用，所以在 MATLAB 中有一些关于多项式计算的函数。

① **多项式的根**：在 MATLAB 中，多项式是由一个行向量表示的，并且多项式的系数按照降序排列。如多项式 $x^4-15x^3+25x^2+8$ 在 MATLAB 中用向量表示为： $p=[1 \ -15 \ 25 \ 0 \ 8]$ 。

对多项式，在 MATLAB 中可以用 roots 函数直接求出其根，这就简化了我们在一般的数学运算中一元高次方程的根的过程。例如对上述多项式，其方程形式可表示为： $x^4-15x^3+25x^2+8=0$ ，我们输入：

```
>>r=roots(p)
```

立即可得到 $r=13.0860 \quad 2.0781 \quad -0.0820+0.5362i \quad -0.0820-0.5362i$ ，即两个实根和两个虚根。

对得到的多项式的根，可以通过 ploy 函数得到相应的多项式。例如：

```
>>p=poly(r)
```

```
p=1.0000 -15.0000 25.0000 -0.0000 8.0000
```

立即得到原来的多项式向量。

②**多项式的四则运算**：由于 MTLAB 中多项式是用行向量定义的，所以多项式的加减运算与行向量的加减运算过程完全相同。但要注意，多项式的加减运算必须是相同阶数的运算，对低阶多项式需要用零补充相应高阶项，然后再相加减。

例如 $x^4-15x^3+25x^2+8$ 与 x^3+5x+3 相加减过程为：

```
p1=[1 -15 25 0 8];
```

```
p2=[0 1 0 5 3];
```

```
p=p1+p2
```

```
p=1 -14 25 5 11
```

即多项式 $x^4-14x^3+25x^2+5x+11$

关于多项式的乘法运算，是通过卷积函数 conv 来进行的。例如多项式 $x^4-15x^3+25x^2+8$ 与 x^3+5x+3 相乘过程为：

```
>>p1=[1 -15 25 0 8];
```

```
>>p2=[0 1 0 5 3];
```

```
>>p=conv(p1,p2)
```

```
p=0 1 -15 30 -72 88 75 40 24
```

即得到多项式： $x^7-15x^6+30x^5-72x^4-88x^3+75x^2+40x+24$

一些特殊情况下需要多项式进行除法运算，这由函数`[q,r]=deconv(a,b)`来实现，其中 `q` 表示除后的商，`r` 表示余数，上式表示多项式 `a` 除以 `b`。例如：

```
>> [q,r]=deconv(p,p1) %表示多项式 p 除以 p1，余数正好为 0
q = 0      1      0      5      3
r = 0      0      0      0      0      0      0      0
```

③

hanshu 的导数

：对多项式的导数是通过函数 `polyder` 函数来实现的，主要有以下几种格式：

`k=polyder(a)` %返回多项式 `a` 的一阶导数 `k`

`k=polyder(a,b)` %返回多项式 `a` 和 `b` 乘积的一阶导数 `k`

`[q,d]=polyder(a,b)` %返回多项式 `a/b` 的导数，`q` 为导数分子，`d` 为导数分母

例如对多项式 $x^4-15x^3+25x^2+8$ 与 x^3+5x+3 ，输入下列命令：

```
>> p1=[1 -15 25 0 8];
>> p2=[0 1 0 5 3];
>> k=polyder(p1)
k = 4 -45 50 0
>> d=polyder(p1,p2)
d = 7 -90 150 -288 264 150 40
```

④

求值函数

：这是由 `polyval` 实现的，其基本格式为：

`k=polyval(p,x)` %求多项式 `p` 在 `x` 点处的值。例如求多项式 x^3+5x+3 在 $x=1,2,3$ 处的值：

```
>> p2=[1 0 5 3]
>> c=polyval(p2,[1,2,3])
c = 9 21 45
```

(2) 求函数的零点

所谓函数零点，就是函数 $f(x)$ 等于零时，函数中自变量 x 的值。这由函数 `fzero` 来实现，其基本格式为：`z=fzero('fun',x)`。`fun` 表示函数的形式，用单引号括起来。`x` 表示函数中的自变量的取值位置，`z` 则表示函数等于零时自变量在 x 附近的值。若 x 为二维向量，则寻找在 x 指定的区间内的零点。例如，求函数 $\sin(x)$ 等于零时自变量在 3 附近的值可以按如下操作：

```
>> z=fzero('sin',3) %寻找 3 附近的 sin(x) 零点
z = 1.416
>> x=[-pi,pi]
>> z=fzero('sin',x) %寻找 (-3.14, 3.14) 区间内的 sin(x) 零点
z = 0
```

(3) 求函数的极值

实际科研工作中经常碰到求函数的极大值极小值的问题，一般是通过求函数的导数，然后令其等于零解得此时自变量的值，再将此自变量值代入原函数中求得其极值。这种手工方法不但烦琐，而且有时候一些复杂函数的导数很难求得，这使得手工求导数变得不可行。在 MATLAB 中函数的极值是通过数值方法求得的，这使得对所有函数的极值都可以求得。MATLAB 中函数的极值是通过函数 `fminbnd` 或 `fminsearch` 实现的，其基本形式有：

`x=fminbnd('fun',x1,x2)` %返回函数 `fun` 在区间 $(x1,x2)$ 的极小值

`x=fminsearch('fun',[x0,y0])` %返回多元函数 `fun` 以初始值 $(x0,y0,z0)$ 开始的极小值

例如，求函数 $f(x)=x^3-2x-5$ 在区间 $(0, 2)$ 的极小值。

首先定义 M 文件 f.m 如下：

```
function y=f(x)
```

```
y=x^3-2*x-5
```

然后调用函数 $x = \text{fminbnd}('f',0,2)$ 可得到 $y=-6.088$, $x=0.8165$

fminsearch 可以求多元函数的极值。例如求函数 $f(x)=100(x_2-x_1)^2+(1-x_1)^2$ 的极小值。首先定义函数 fl2:

```
function y=fl2(x)
```

```
y=100*(x(2)-x(1))^2+(1-x(1))^2
```

然后调用函数 $>>x=\text{fminsearch}('fl2',[1.2,1])$ %以 $(1.2, 1)$ 为初值的极小值。

得 $y=6.590533196313579\text{e-}009$, $x=0.99999991652183 \quad 1.00000269180851$

(4) 插值

插值是对数据点之间函数的估计方法。在环境生态学的研究中，经常碰到这样的情况：对某一区域在许多点位取样测试某种污染物的含量，当要估计该区域未取样点的污染物含量时，可以根据已经取样点的测试情况来估计。这就是插值的具体运用。

- ① **一维插值**：在 MATLAB 中一维插值是通过函数 `interp1` 实现的，其基本格式为： **$y_i=\text{interp1}(x,y,xi)$** 或 **$y_i=\text{interp1}(x,y,xi,'method')$** 。其中 x 、 y 分别为已知的数据点向量。 xi 表示所需插值位置向量， y_i 表示返回的在所插值位置插值后的向量。**'method'**表示所用插值的方法。一般有以下几种：**'nearest'**—表示最近点插值；**'linear'**—表示线性插值方法（默认的方法）；**'spline'**—样条插值；**'cubic'**—立方插值。下面举例说明上述函数应用

例 7_1:

在某一空间方向上有 10 个采样点，间隔 10m，则得到位置向量为： $x=[0 \ 10 \ 20 \ 30 \ 40 \ 50 \ 60 \ 70 \ 80 \ 90]$ 。在 10 个采样点分别得到土壤中元素砷的含量（mg/kg）向量为 $y=[0.4 \ 0.3 \ 0.4 \ 0.5 \ 0.55 \ 0.65 \ 0.7 \ 0.8 \ 0.83 \ 0.9]$ ，试估算在位置 5m，15m，25m 处的土壤元素砷含量。输入下面命令，得到 $y1i$ 和 $y2i$ 及下面的图形。

```
>>x=[0 10 20 30 40 50 60 70 80 90];
```

```
>>y=[0.4 0.3 0.4 0.5 0.55 0.65 0.7 0.8 0.83 0.9];
```

```
>>xi=[5 15 25];
```

```
>>y1i=interp1(x,y,xi,'spline') %样条插值
```

```
y1i = 0.3109    0.3390    0.4579
```

```
>>y2i=interp1(x,y,xi) %线性插值
```

```
y2i = 0.3500    0.3500    0.4500
```

```
>>plot(x,y,'-m');
```

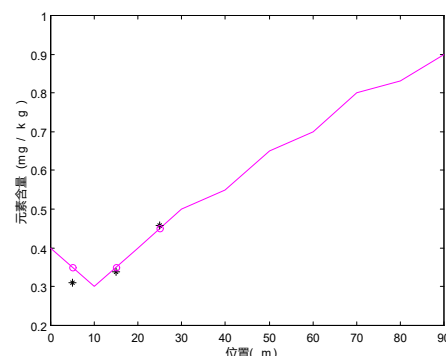
```
>>hold on
```

```
>>ylabel('元素含量 (mg/kg) ');
```

```
>>xlabel('位置(m)');
```

```
>>plot(xi,y1i,'k*');
```

```
>>plot(xi,y2i,'mo');
```



可见样条插值的结果和线性插值的结果是不一样的。

② **二维插值**：二维插值的思想同一维插值是一样的。只是二维插值是发生在平面上，而一维插值是在某线性方向上。二维插值的函数为 `interp2`，其调用格式为： **$zi=\text{interp2}(x,y,z,xi,yi)$** 或 **$zi=\text{interp2}(z,xi,yi)$** 或 **$zi=\text{interp2}(x,y,z,xi,yi,'method')$** 。

其中 z 为平面上已知点的值组成的矩阵, x,y 是与 z 同维的已知点在平面上的横坐标和纵坐标矩阵。 xi,yi 则为要插值点的平面位置坐标, zi 为返回的插值点数值。
例 7_2:

在某湖泊内设置采样网格进行取样, 取样测试数据见下表。要求预计在平面坐标(120, 130)、(200, 210)、(150, 240)处的湖泊水污染物含量。

表 11 湖泊水样中某污染物含量分析结果(mg/L) (括号中数字可表示为坐标点)

东西方向(x) 南北方向(y)	100m	150m	200m	280m	350m
100m	8.2(100,100)	8.1(150,100)	8.0(200,100)	8.2(280,100)	8.4(350,100)
200m	7.9(100,200)	6.3(150,200)	6.1(200,200)	6.5(280,200)	8.1(350,200)
300m	8.5(100,300)	8.3(150,300)	8.2(200,300)	8.5(280,300)	8.6(350,300)

上述问题可以用平面二维插值来解决。由题意可知:

```
z = 8.2000 8.1000 8.0000 8.2000 8.4000
     7.9000 6.3000 6.1000 6.5000 8.1000
     8.5000 8.3000 8.2000 8.5000 8.6000
```

```
x=[100 150 200 280 350], y=[100 200 300]
```

```
xi=[120 200 150 ], yi=[130 210 240 ]
```

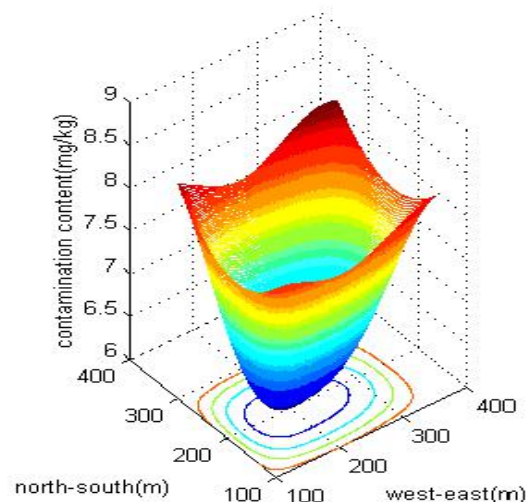
根据上述数据, 输入下面命令:

```
>> x=[100 150 200 280 350];
>> y=[100 200 300];
>> z=[8.2 8.1 8.0 8.2 8.4;7.9 6.3 6.1 6.5 8.1;8.5 8.3 8.2 8.5 8.6];
>> xi=[120 200 150 ];
>> yi=[130 210 240 ]; %定义 yi 为行向量
>> zi=interp2(x,y,z,xi,yi) %默认的 xianxing 插值计算
zi =7.8900 6.3100 7.1000 %在三个插值点得到湖泊水污染物含量
>> zi=interp2(x,y,z,xi,yi,'cubic') %用 cubic 插值方法计算
zi = 7.5538 7.0100 7.1610 %可见和上述的插值结果差别较大
另外建议比较与体会如下命令与上述命令的区别:
>> yi=[130; 210; 240 ];%定义 yi 为列向量
>> zi=interp2(x,y,z,xi,yi) %插值结果为三行数字, 且对角线上数字与上述插值结果
相同 (考虑这种差别是由于 yi 的变化引起的)
```

```
zi =7.8900 7.4300 7.5600
     7.3760 6.3100 6.5000
     7.7240 6.9400 7.1000
```

利用二维插值的这种形式, 可以将平面插值的结果用图形表示出来。仍然以上述数据为例。输入如下命令:

```
>> x=[100 150 200 280 350];
>> y=[100 200 300];
>> z=[8.2 8.1 8.0 8.2 8.4;7.9 6.3 6.1 6.5 8.1;8.5 8.3 8.2 8.5 8.6];
>> xi=100:1:400;%定义多个插值横坐标
>> yi=100:1:400;%定义多个插值纵坐标
>> yi=yi'; %将 yi 定义为列向量
```




```
>> zi=interp2(x,y,z,xi,yi,'cubic');
>>meshc(xi,yi,zi)%以图形表示出插值结果，用 surf 命令也可以
>>xlabel('west-east(m)')
>>ylabel('north-south(m)')
>>zlabel('contamination content(mg/kg)')
```

由图形可以形象地看出，污染物在中心处的浓度比较低，而在边缘比较高。这种插值的趋势和表 11 中的原始数据分布趋势是一致的。

(5) 曲线拟合

所谓曲线拟合就是将一系列的测试数据拟合为一定的函数形式。在 MATLAB 中曲线的多项式拟合是通过函数 `polyfit` 实现的。其调用格式为：

a=polyfit(x,y,n)。其中 **x,y** 为要拟合的数据，**n** 为拟合多项式的次数，**a** 为输出参数。若拟合得到的多项式为 $y=a_1x^n+\dots+a_{n-1}x^2+a_nx+a_{n+1}$ ，则 $a=[a_1,a_2\dots a_n,a_{n+1}]$ 。

例 7_3:

某企业经济发展数据和污染物排放量数据如表 12，试拟合其间的关系。

表 12

年份	1995	1996	1997	1998	1999
经济收益（万元）	765	826	873	942	1032
污染物发生量(万 t)	20.4	31.6	50.8	76.2	97.3

输入下述命令：

```
>>x=[765 826 873 942 1032];
>>y=[20.4 31.6 50.8 76.2 97.3];
>>a1=polyfit(x,y,1) %进行 1 次拟合
a1 =0.3035 -214.1553 %得到多项式为 y=0.3035x -214.1553
>>a2=polyfit(x,y,2) %进行 2 次拟合
A2=-0.0000 0.3515 -235.5032 %得到多项式为 y=-0.0000x^2+0.3515x-235.5032
```

上述二次拟合得到多项式的二次项近似等于 0，这是由于原始数据几乎呈直线分布（可以用 `plot` 命令验证），所以当然不会拟合出曲线来。

7.2 积分和微分

(1) 数值积分

MATLAB 中的积分方法有很多，这里只介绍使用较多的辛普森积分函数。基本调用格式为：**q=quad('fun',a,b)**。**fun** 为积分函数的形式，**a,b** 分别为积分区间的上下限，表示从 **a** 到 **b** 积分，**q** 为返回积分的结果。

例如某函数形式 $fun(x) = 4 \int_0^{\pi/2} (8755 \sin^2 x + 6810 \cos^2 x)^{1/2} dx$ ，试对之求解。

首先需要编写如下形式的 `fun.m` 文件

```
function y=fun(x)
y=sqrt(8755*sin(x).^2+6810*cos(x).^2);
```

在 MATLAB 命令窗口中可输入下面的命令：

```
>>q=quad('fun',0,pi/2) 对函数 fun 在 0-pi/2 范围内求积分
q=138.4375 %立即得到结果
```

(2) 数值微分

MATLAB 中的微分函数是 `diff`，其基本形式为： **$y=\text{diff}(x)$** 或 **$y=\text{diff}(x,n)$** 。其中 x 为向量时，计算元素间的差分，返回的 y 向量比 x 向量少一个元素。当 x 为矩阵时，返回矩阵列之间的差分矩阵。 n 为可选项，表示计算差分次数，即导数的阶数。

例如>>`x=1:6`

`x=1 2 3 4 5 6`

>>`y1=diff(x) %计算 x 的 1 阶差分`

`y1=1 1 1 1 1`

>>`y2=diff(x,2) %计算 x 的 2 阶差分`

`y2=0 0 0 0`

7.3 线性方程组求解

可以通过把线性方程组化为 $AX=B$ 的形式，其中 A 表示 $n \times n$ 的系数矩阵, B 表示 $n \times 1$ 的常数矩阵, X 表示 $n \times 1$ 的未知数矩阵。这样可以通过矩阵求逆解线性方程组。即 $X=\text{inv}(A)*B$ 。

7.4 回归分析

很多时候，一些变量之间存在着相关关系。这种关系很难用确定的函数形式表现出来，例如田间农药使用量与区域地表水环境中该农药的浓度水平之间、人的身高与腿长之间、降水量与河流输沙量之间等。但这种关系可以用回归分析技术进行出来。由于回归分析的计算量很大，所以一般不适用于手工计算，而 MATLAB 则为此提供了强大的计算功能。

(1) 多元线性回归分析

多元线性回归分析的目的是建立预报变量组 (x 矩阵的列) 与响应变量 y 之间的一种定量关系。

多元线性回归的一般结构为： $y=xB+e$ ，其中 y 为 $n \times 1$ 阶的观察值向量， x 为 $n \times p$ 阶的回归矩阵， B 是 $p \times 1$ 阶的系数向量， e 表示 $n \times 1$ 随机误差项。

该问题的解是一个向量 $b=B=(x'x)^{-1}xy$ 。

对多元回归是通过函数 `regress` 来实现的，其基本形式有：

`b = regress(y,x)` 返回基于观测值 y 和回归矩阵 x 的最小二乘拟合 B 的结果

`[b,bint,r,rint,stats] = regress(y,x,alpha)` 返回参数中 b 表示拟合方程的系数向量， $bint$ 表示参数向量 b 的 95%置信区间，残差 r 以及每个残差的 95%置信区间 $rint$ ，而矢量 $stats$ 则依次给出回归的复相关系数 R^2 、统计量 F 和回归概率 p 。**`alpha`** 为预先给定的置信水平，若 $p < 1-\text{alpha}$ 则回归模型才能成立，否则不成立。

下面通过一个例子来说明 `regress` 函数的用法。

例 7_4:

`load moore %调用 MATLAB 内置变量 moore`

`moore =`

x1	x2	x3	x4	x5	y
1125	232	7160	85.9	8905	1.5563
920	268	8804	86.5	7388	0.8976
835	271	8108	85.2	5348	0.7482

1000	237	6370	83.8	8056	0.716
1150	192	6441	82.1	6960	0.313
990	202	5154	79.2	5690	0.3617
840	184	5896	81.2	6932	0.1139
650	200	5336	80.6	5400	0.1139
640	180	5041	78.4	3177	-0.2218
583	165	5012	79.3	4461	-0.1549
570	151	4825	78.7	3901	0
570	171	4391	78	5002	0
510	243	4320	72.3	4665	-0.0969
555	147	3709	74.9	4642	-0.2218
460	286	3969	74.4	4840	-0.3979
275	198	3558	72.5	4479	-0.1549
510	196	4361	57.7	4200	-0.2218
165	210	3301	71.8	3410	-0.3979
244	327	2964	72.5	3360	-0.5229
79	334	2777	71.9	2599	-0.0458

>>x=[ones(size(moore,1),1) moore(:,1:5)]; %产生的矩阵 x 第一列全部为 1，后面 5 列则为矩阵 moore 中的前 5 列

>>y=moore(:,6);

>>[b,bint,r,rint,stats]=regress(y,x);

b=-2.1561

0.0000

0.0013

0.0001

0.0079

0.0001

可见拟合的方程为 $y = -2.1561 + 0.000x_1 + 0.0013x_2 + 0.0001x_3 + 0.0079x_4 + 0.0001x_5$ 。

stats =0.8107 11.9890 0.0001

可见 $R^2=0.8107$, $F=11.9890$, $p=0.0001$ (表明回归系数出现全部为零的几率很小，所以模型可以接受)。

(2) 非线性回归

非线性回归的一般结构为: $y=f(B,x)+e$ ，其中 y 为 $n \times 1$ 阶的观察值向量， x 为 $n \times p$ 阶的输入变量矩阵， B 是 $p \times 1$ 阶的系数向量， e 表示 $n \times 1$ 随机误差项。

非线性回归是通过函数 `nlinfit` 来实现的，其基本形式为：

b =nlinfit(x,y,'model',beta0) 返回由 model 确定的非线性函数的系数，**beta0** 为回归系数的初值。 X 一般是一个矩阵，其列是每个变量的观测值， y 是因变量，是列向量，元素个数等于矩阵 x 的行数。

例 7_5:

非线性方程 $rate = \frac{\beta_1 x_2 - x_3 / \beta_5}{1 + \beta_2 x_1 + \beta_3 x_2 + \beta_4 x_3}$ 表示污染物的化学降解速率 $rate$ 和

污染物含量（x）之间的关系。测试数据如下：

reactants =

x1	x2	x3
470	300	10
285	80	10
470	300	120
470	80	120
470	80	10
100	190	10
100	80	65
470	190	65
100	300	54
100	300	120
100	80	120
285	300	10
285	190	120

rate=[8.55 3.79 4.82 0.02 2.75 14.39 2.54 4.35 13 8.5 0.05 11.32 3.13]^T

然后定义下面的函数：hougen

```
function yhat=hougen(beta,x)
```

```
b1 = beta(1);
```

```
b2 = beta(2);
```

```
b3 = beta(3);
```

```
b4 = beta(4);
```

```
b5 = beta(5);
```

```
x1 = x(:,1);
```

```
x2 = x(:,2);
```

```
x3 = x(:,3);
```

```
yhat = (b1*x2 - x3/b5)./(1+b2*x1+b3*x2+b4*x3);
```

>>beta=[1 0.05 0.02 0.1 2.00] %定义参数 β_1 — β_5 的初值。

>>b= nlinfit(reactants,rate,'hougen',beta)

b=1.2526

0.062776

0.040048

0.11241

1.1914

即 $\beta_1=1.2526$, $\beta_2=0.06277$, $\beta_3=0.040048$, $\beta_4=0.1124$, $\beta_5=1.1914$ 。

7.5 本章练习

1、利用 MATLAB 的多项式功能(`roots` 函数), 计算 25℃时, 正常降水的 pH 值 (5.65)。提示: 清洁大气中一般只考虑 CO₂ 溶解于水造成的酸度, 即有平衡:



$\text{H}_2\text{O} \xrightleftharpoons{k_w} \text{H}^+ + \text{OH}^-$ 。25℃时, $k_H = 3.36 \times 10^{-7} \text{ mol}/(\text{L} \cdot \text{Pa})$, $k_1 = 4.47 \times 10^{-7}$, $k_2 = 4.68 \times 10^{-11}$, $k_w = 1.0 \times 10^{-14}$ 。 $P_{\text{CO}_2} = 33 \text{ Pa}$, $[\text{H}^+] = [\text{OH}^-] + [\text{HCO}_3^-] + 2[\text{CO}_3^{2-}]$ 。

$$[\text{H}^+] = \frac{K_w}{[\text{H}^+]} + \frac{K_1 K_H P_{\text{CO}_2}}{[\text{H}^+]} + \frac{2K_2 K_1 K_H P_{\text{CO}_2}}{[\text{H}^+]^2}。$$

2、有函数 $f(x) = x^4 - 2x^3 - x^2 - 2x + 7$, 求其在 $[-4, 4]$ 区间内的最小值, (-1.0959)。

(提示: `x=-4:1:4;y=x.^4-2*x.^3-x.^2-2*x+7;plot(x,y);grid on;F='x^4-2*x^3-x.^2-2*x+7';X=fminbnd(F,-4,4)%X=1.9013`

`syms xx; Vmin=subs(xx^4-2*xx^3-xx^2-2*xx+7,xx,X)% Vmin=-1.0959`

`VminFig=ginput(1)`

`VminFig=1.9637 -0.9693`)。

3、有一些数据, 比如人口增长数据, 污染物发生量增长数据等具有一定的递增或递减的性质, 利用这些性质以及插值函数的功能可以预测某一时间内数据的变化情况。现假设某地区污染物发生量从 1955 到 2000 年 (每隔 5 年) 的污染物发生量 (万吨/年) 是 75.995 91.972 105.711 123.203 131.669 150.697 179.323 203.212 226.505 249.633。若依此趋势发展, 试推测 1994 年该地区污染物发生量。

4、已知某地区一些地点的土壤中实测某种污染物含量 (表 12), (1) 试利用插值作出其污染物含量的等值线图 (用 `contour` 命令) (2) 将这种污染物含量插值结果转换立体曲面图 (`mesh` 或 `surf` 命令)。可以练习使用不同的插值方法得到图形的平滑性, 体会不同插值方法的精度。

X(m) Y(m)	100	200	300	400	500	600	700
100	1.13	1.25	1.28	1.23	1.04	0.89	0.65
150	1.32	1.45	1.42	1.40	1.02	0.74	0.76
200	1.39	1.50	1.50	1.40	1.30	1.11	0.85
250	1.48	1.20	1.10	1.35	0.85	1.20	1.01
300	1.49	1.20	1.10	1.55	1.45	1.65	1.10
350	1.47	1.55	1.60	1.55	1.64	1.60	1.20
400	1.50	1.50	1.55	1.51	1.60	1.30	1.25

5、根据下表数据，拟合如下方程：

$$z=1.2202+1.1967x-0.4836y-0.2948x^2+0.0804xy+0.0124y^2+0.0174x^3+0.0054x^2y-0.0121xy^2+0.0066y^3$$

(其中，z 表示元素含量，x,y 表示坐标)

序号	x/m	y/m	z/mg/kg	序号	x/m	y/m	z/mg/kg
1	1	1	1.8	9	1	9	3.5
2	1	2	1.4	10	2	8	3.4
3	2	3	1.5	11	3	7	2.4
4	2	5	1.6	12	2	8	1.8
5	4	6	1.7	13	3	5	2.4
6	4	7	2.6	14	6	4	1.7
7	5	8	2.7	15	7	2	1.6
8	5	9	3.4	16	8	3	1.8

(提示：[b,bint,r,rint stats]=regress(y,X),X0=1, X1=x, X2=y,X3=x².....X9=y³)。

第八章 总结