第 3 章

数 值 计 算

3.1 基 本 运 算

3.1.1 基本运算的条件

加减运算是对矩阵、向量、数都可进行的基本运算；矩阵的乘法需要左边矩阵的列数与右边矩阵的行数必须相等。违反这一条件，MATLAB 会显示红色的出错信息：

Error using * Incorrect dimensions for matrix multiplication。

单击出错信息中的“*”，就可以看到详细说明。

行向量乘以列向量（两者长度相同）等于一个数，等价于两个对应的行（列）向量的内积，这样做不用调用求内积的内置函数。而列向量乘以行向量等于一个矩阵。

两个数可以有“除”与“除以”的运算；矩阵的这两种运算是不可行的。当一个矩阵可逆的时候，乘以它的逆阵，就相当于“除以”这个矩阵。

因为矩阵是 MATLAB 的基本数据单位，MATLAB 把一个矩阵与一个数的加减定义为这个矩阵的每个元素加上或减去这个数，得到一个同样大小的矩阵。如果正希望这样的结果，那么只要在这个矩阵与这个数之间添上“+”号或“-”号即可，而不必像其他一些语言一样，要用循环语句编一段程序来实现。

MATLAB 还对两个同样大小的矩阵定义了“点乘”与“点除”的运算：两者对应每个元素相乘或相除。这对科学计算十分方便，省去了另编程序的麻烦。对一个方阵，MATLAB 还定义了“点乘方”的运算：结果是矩阵的每个元素的乘方所得到的矩阵。这与 MATLAB 所有的内置函数一样，代入一个矩阵实参，它就对此矩阵的每个元素进行同一种运算。

3.1.2 算术运算（符）

MATLAB 中的算术运算符如表 3.1 所示。

表 3.1 算术运算符

运 算 符	输 入 指 令	输 出 结 果
“+”或“-”（加上或减去）	>> a=1+2, b=4 -5	a=3, b=-1
“*”（乘以）	>> c=8 * 2, d=(4+2i) * (3-5i) % i 是虚数单位 $\sqrt{-1}$	c= 16 d= 22-14i
“/”（forward slash：divided by，除以或被…除）	>> u=[8,6, 4; 3,2,1]/2	u= 4.000 3.000 3.000 1.500 1.000 0.500

续表

运　算　符	输 入 指 令	输 出 结 果
“\”（backslash：divided into，除）	>> v= 2\[8,6, 4; 3,2,1] %而[8,6; 3,2]\2 出错！因为分母不能是矩阵	v= 4.000　3.000　3.000 1.500　1.000　0.500
“^”（to the power of，乘方），只适用于方阵，包括一个数字	>> x=2^3 y=[2, 1-3i; 4-2i, -1-6i]^(1/2)	x=8 y= 1+i　1-i 2-0i　1-2i

（1）矩阵（向量）的加减是对应元素相加减，只要尺寸相同即可。

（2）两个矩阵的相乘，并不定义为对应元素相乘，而是左边矩阵的各行向量与右边矩阵的各列向量的内积。所以，两个矩阵能相乘的条件是左边矩阵的列数（即行向量的维数）与右边矩阵的行数（即列向量的维数）必须相等。使用运算符“*”时，必须小心。

（3）向后（反）斜线运算符“\”用在两个数字之间，“a\b”是“a 除 b”，等价于“a 的逆（或倒数）乘以 b”，即 a\b=a^(-1)*b。这与线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 的解 $\boldsymbol{x}=\boldsymbol{A}^{-1}\boldsymbol{b}$（$\boldsymbol{A}^{-1}$ 左乘以 $\boldsymbol{b}$）在形式上是一样的，因此可以用来解线性方程组：MATLAB 的指令为“x=A\b”。同样，可以用内置函数 mldivide，指令为“x= mldivide(A,b)”。

注意，今后为了输出结果少占空间，有时用输入向量、矩阵的格式来输出向量或矩阵。

例如，X2 =［-1；3］的分量之间用分号，表示列向量；若用逗号，则为行向量。

例如：

输 入 指 令	输 出 结 果
>>A=[2,-1;3,-2];　b=[-5;-9]; X1=A\b,　X2=inv(A)*b,　X3= mldivide(A,b) %inv(A) 是 A 的逆矩阵;X1 与 X3 显示小数	X1 =　[-1.0000; 3.0000] X2 =　[-1; 3] X3 =　[-1.0000; 3.0000]

注意，作为数字相除，“a\b”是“a 的逆乘以 b”与“b/a”是“b 乘以 a 的逆”是一样的，因为数字乘法可换，但矩阵乘法（一般）不可换。而且，矩阵不能作分母。所以如果输入“x=b/A”，就会出错。

（4）类似地，向前斜线运算符“/”，也可以用在两个矩阵（向量）之间。“B/A”等价于“B*inv(A)”，都是 $\boldsymbol{B}$ 左乘以 $\boldsymbol{A}$ 的逆，$\boldsymbol{A}$ 必须是可逆方阵，$\boldsymbol{B}$ 的列数必须等于 $\boldsymbol{A}$ 的阶数。同样，可以用内置函数 mrdivide，指令为“x=mrdivide(B,A)”。

例如：

输 入 指 令	输 出 结 果
>> A=[2,-1;3,-2];　B=[-5,-9]; %同上	
x1=B/A,	x1 =　-37.0000　23.0000
x2=B*inv(A),	x2 =　-37.0000　23.0000
x3= mrdivide(B,A),	x3 =　-37.0000　23.0000
x4= mrdivide(A,B)	x4 =　[-0.0094;　0.0283]　%列向量
x5=A*pinv(B)	x5 = x4

注意，计算 x4 的 ***B*** 是个行向量，不存在逆阵，这里实际上用了 ***B*** 的“广义逆”（详见第 8 章）。这个指令等价于计算 x5 的指令，其中，pinv 是用来求广义逆的内置函数。

（5）乘方运算符“C^n”，矩阵 ***C*** 只能是方阵。

- 当 n 是正整数时，表示 n 个 ***C*** 连乘。
- 当 n 是负整数时，矩阵 ***C*** 只能是可逆方阵，表示 n 个 ***C*** 的逆连乘。
- “C ^(－1)”等价于“inv(C)”，即 ***C*** 的逆阵。
- 当 $n=0$ 时，无论矩阵 ***C*** 是否是可逆方阵，结果都是单位阵 ***I***。

例如：

输 入 指 令	输 出 结 果
>> C=[2,5;1,3], Y1=inv(C), Y2=C^(-1)	 Y1 =[3　　-5;　-1　　2] Y2 =[3　　-5;　-1　　2]
B1=C^(-2), B2=Y1^2	B1 =[14.0000,　-25.0000; -5.0000,　9.0000] B2 =[14,　-25 ;　-5　　9]
>> D=[2,1;4,2];　detD=det(D),　E= D^0 % 内置函数 det 是求方阵的行列式，见 3.2 节	detD = 0,　E= eye(2)

3.1.3　点乘、点乘方与点除运算

1. 点运算的定义与运算符

MATLAB 还设置了计算两个矩阵，包括两个向量的对应元素相乘或相除，或者一个矩阵（向量）的每个元素的乘方的运算符：在“*”“/”“\”“^”的前面加上点“.”（dot），成为“.*”“./”“.\”“.^”。参加运算的两个矩阵（向量）必须有同样的尺寸。

例如：

输 入 指 令	输 出 结 果
>> x=[-1,5,3-4i]; y=[-2,-4,1+2i]; u=x.*y, v=x./y, w=x.^2,　t=x.\y	 % 以下删去了 4 位或最后 2、3 位全 0 的小数及 0 i u = 2.　-20.　11. + 2. i;　　v = 0.5　-1.25　-1. - 2. i w = 1.　25.　-7. -24. i;　　t = 2.　-0.8　-0.2 + 0.4i

2. 运算的先后次序

MATLAB 的运算次序与数学规定的一样，即

- 有括号的先做括号内的运算，有多层括号（全用小括号）的，先做最里层的。
- 然后是先做（点）乘方，再做（点）乘除，最后做加减。点乘除与乘除出现在同一项时，按先后次序运算，不能用结合律，见下面 B2 和 B2b 的区别。

例如：

输入指令	输出结果
>> C=[2,5;1,3];　D=[2,1;4,2]; B=C * D^0./D + ((C-D). * C).^2,	B = 1.0000　405.0000 9.2500　10.5000
% 以下运算用于检测运算符的优先级别。 B1=D^0,　B2=C * B1./D, B2b=C * (B1./D),　% 注意 B2 与 B2b 的区别	B1 = [1,　0;　0　1] B2 = [1.0000,　5.0000;　0.2500,　1.5000] B2b =[1.0000,　2.5000;　0.5000,　1.5000]
B3=C-D, B4=B3. * C, B5=B4.^2, B6=B2+B5	B3 = [0　4;　-3　1] B4 = [0　20;　-3　3] B5 =[　0　400;　9　9] B6 = B

3.1.4　数值的字符表达和分数表达

一个变量一般用数值表达，但也可以用字符表达或分数表达。例如，可以把圆周率 π 转换为最接近的分数："R＝rats(pi)""format long；R"或分数表达"format rat；R"。

rats 的"单数"rat 是把小数转换为连分数。rats 与 rat 只是用来显示所转换成的分数或连分数的，不能用来参加运算。例如，rats(pi)＋1 会出错。但是，"format rat；pi＋1"能得到正确结果。

例如：

输入指令	输出结果
>> R=rats(pi),　% 字符表达(π 表达为祖冲之的密率) format long; R,　% 字符表达 format rat; R1=pi+1　% 分数表达	R =　'　355/113　' R =　'　355/113　' R1=　468/113

3.2　矩阵的一元运算

矩阵的一元运算有矩阵的转置、求逆、数乘与矩阵的行列式。

3.2.1　矩阵的转置

矩阵的转置运算用符号""实现，即 $\boldsymbol{A}$ 的转置矩阵 $\boldsymbol{A}^{\mathrm{T}}$。在代码中是"A'"。注意，这个""必须是在 MATLAB 的指令窗口或"编辑"(Editor)窗口中，而且是英文环境下打印出来的。它不同于 Word 在英文环境下输入的"'"。

例如：

输入指令	输出结果
>> A=[11,12,13,14; 21,22,23,24; 31,32,33,34],	A =　11　12　13　14 21　22　23　24 31　32　33　34

续表

输入指令	输出结果
B=A'	B = 11 21 31 12 22 32 13 23 33 14 24 34

这样，一个列向量可用一个行向量(它显示出来所占的空间小得多)的转置来实现。

本书以斜黑体小写字母表示列向量。线性代数书中不少把线性方程组表示为 $\boldsymbol{Ax}=\boldsymbol{b}$，其中的 $\boldsymbol{x}$ 是未知量形成的列向量，而 $\boldsymbol{b}$ 则是右端项常数形成的列向量。所以本书的表示法与此一致。

列向量的分量表达式，为少占空间，也写成行向量加转置。例如，$\boldsymbol{x}=(1,2,3,4,5)^{\mathrm{T}}$。赋值语句可以写为“x = [1, 2, 3, 4, 5]'”(最后是一撇“'”)或“x =[1; 2; 3; 4; 5]”(分量之间用分号隔开)。

若 $\boldsymbol{C}$ 是复矩阵，则运算符“'”是共轭转置，即转置以后，再对每个元素取共轭复数。

例如：

输入指令	输出结果
>> C=[1+i, -2, i; 3, -i, -1-2 * i], D=C'	C = 1+i -2+0i 0+i 3+0i 0-i -1-2i D = 1.0000 -1.0000i 3.0000 +0.0000i -2.0000 +0.0000i 0.0000 +1.0000i 0.0000 -1.0000i -1.0000 +2.0000i

3.2.2 数乘矩阵

数 r 乘矩阵 $\boldsymbol{A}$，直接用“r * A”或用点乘“r. * A”来实现。

3.2.3 方阵的行列式

方阵 $\boldsymbol{A}$ 的行列式，用内置函数 det：“det($\boldsymbol{A}$)”。

例如：

输入指令	输出结果
>> M=[2, -1, 3; -1, 0, 4; -2, 1, 5]; r=det(M)	r = -8

3.2.4 方阵的逆

当方阵 $\boldsymbol{M}$ 的行列式非零时，它的逆矩阵存在，用指令“N=inv(M)”求逆矩阵 $\boldsymbol{N}$，逆矩阵元素用小数表示。指令“rats(N)”或“format rat; N”可把 $\boldsymbol{N}$ 的每个元素都转换为分数。

例如：

输 入 指 令	输 出 结 果
>> M=[2, -1, 3; -1, 0, 4; -2, 1, 5]; N=inv(M)	N =0.5000　-1.0000　0.5000 　0.3750　-2.0000　1.3750 　0.1250　　0　0.1250
N1=rats(N) format rat; N	N1 = 3×42 char 数组 　' 1/2　　-1　　1/2　' 　' 3/8　　-2　　11/8　' 　' 1/8　　0　　1/8　' N =1/2　-1　1/2 　3/8　-2　11/8 　1/8　0　1/8

3.2.5　与矩阵相关的其他数值

1. 矩阵的尺寸

内置函数 size(A) 或 [r,c]= size(A) 返回一个行向量，其元素是矩阵 ***A*** 的行数 r 与列数 c。length (A)返回 ***A*** 的行数与列数中的最大值，等价于 max(size(A))。对于向量，长度仅仅是元素数量。空矩阵的行数、列数与长度为零。

例如：

输 入 指 令	输 出 结 果
>> M=[2, -1, 3; -2, 1, 5];　　%";"结尾的指令不显示 L1=size(M), L2 = length(M),　L3 = max(size(M)), [r,c]= size(M)	 L1 =　　2　　3; L2 = L3 =　　3 r =　　2;　　c =　　3

2. 矩阵的秩

用内置函数 rank()来获得。

例如：

输 入 指 令	输 出 结 果
>> M=[11,12,13,14; 21,22,23,24; 31,32,33,34]; L= rank (M),	 L =　　2

这是因为 ***A*** 的任何两行(列)不成比例，但它们的差是成比例的向量：相邻两列的差都等于[1;1;1]，而相邻两行的差都等于[10,10,10,10]。

3. 矩阵各(行或列)元素的和

用内置函数 sum()来获得。

(1) 矩阵 ***A*** 各列元素的和：sum(A) 或 sum(A,1)，输出一个行向量。

(2) 矩阵 ***A*** 各行元素的和：sum(A,2)，输出一个列向量。

(3) 矩阵 ***A*** 各元素的和：sum(sum(A))，输出一个数。

例如：

输入指令	输出结果
>> A=[1, 2, 3; -2, 1, 2]; R1sum= sum(A), R2sum=(sum(A,1)), Csum=sum(A,2), Asum=sum(sum(A))	 R1sum = -1 3 5 R2sum = -1 3 5 Csum = [6; 1] %列向量 Asum = 7

4. 矩阵的最大值

(1) Y=max(A)：矩阵 ***A*** 各列元素的最大值，输出行向量 ***Y***。

(2) [Y, I]=max(A)：行向量 ***Y*** 同(1)，行向量 ***I*** 是取得同列中最大值的(第1个)元素的行下标。

例如：

输入指令	输出结果
>> A=[1, 2, 3; -2, 5, 2; 3, 5, 3]; Y=max(A), [Y,I]=max(A)	Y = 3 5 3 Y = 3 5 3 I = 3 2 1

(3) C=max(A, B)：矩阵 ***C*** 的每个元素是对应的 ***A*** 与 ***B*** 元素中的最大值。特别地，max(A, a) = max(A, a * ones(size(A)))：a 是常数，a * ones(size(A)) 就是一个与 ***A*** 同样尺寸，每个元素全是 a 的矩阵。

(4) max(A, [], DEM)：“[]”是英文状态下的方括号，其中没有内容，表示“空”。DIM=1时，求各列元素的最大者，与 max(A) 相同；DIM=2时，求各行元素的最大者。

例如：

输入指令	输出结果
>> A=[1, 2, 3; -2, 5, 2; 3, 5, 3]; B=[-2, 0, 3; 1, 2, 3; 4, 5, 6]; C=max(A,B), M1=max(A, 2), M2=max(A, 2 * ones(size(A))) % M2= M1,M2 输出省略	C = 1 2 3 1 5 3 4 5 6 M1 = 2 2 3 2 5 2 3 5 3
[Yc,Ic]=max(A,[],1), % 与 max(A) 等价 % Yc,Ic 两个行向量	Yc = 3 5 3 Ic = 3 2 1
[Yr,Ir]=max(A,[],2) % Yr,Ir 两个列向量	Yr = [3; 5; 5] Ir = [3; 2; 2

5. 矩阵的最小值

所有这方面的指令与求矩阵的最大值的指令类似。只要把内置函数 max() 改为

min(),即把求最大值改为求最小值。

例如:

输 入 指 令	输 出 结 果
>> A=[1, 2, 3; -2, 5, 2; 3, 5, 3]; B=[-2, 0, 3; 1, 2, 3; 4, 5, 6]; X1=min(A,B), X2=min(A, 2), [Z,J]=min(A, [],2) % Z,J 两个列向量	X1 =-2　0　3 -2　2　2 3　5　3 X2 = 1　2　2 -2　2　2 2　2　2 Z = [1;　-2;　3],　J=[1;　1;　1]

3.3　向量的内积与外积

3.3.1　向量的内积

向量的**内积**(inner product)又称点积(dot product),是数量积。两个尺寸一样的行向量 ***a*** 与 ***b*** 的内积用 dot(a,b) 或 a * b' 或 b * a' 或 sum(a. * b) 或 sum(b. * a) 来得到。内置函数 sum() 的参数若为向量,是求各分量的和。

例如:

输 入 指 令	输 出 结 果
>> a=[1,0,-2]; b=[-3,2,4]; d1=dot(a,b),　d2=a * b',　d3=b * a', d4=sum(a. * b),　d5=sum(b. * a) , D1=a' * b, D2=b' * a % D2=D1 的转置	 d1 = d2 =　-11 d3 = d4 = d5 =　-11 D1 =-3　2　4 0　0　0 6　-4　-8 (D2 =　D1 ')

其中,D1 与 D2 都是列向量(左)乘以行向量。线性代数基础好的可以自行证明:一个方阵的秩为 1 的充要条件是它等于一个(非 0)列向量乘以一个(非 0)行向量。两个尺寸一样的列向量 ***a*** 与 ***b*** 的内积也可用 dot(a,b)或用类似于上面的其他方法得到,见习题 X3.10。

3.3.2　向量的外积

向量的**外积**(cross product),也称叉积或向量积。两个 3 维行(列)向量 ***a*** 与 ***b*** 的外积用 cross(a,b) 来得到,向量的外积仍是一个向量。

例如:

输 入 指 令	输 出 结 果
>> a=[1,0,-2]; b=[-3,2,4]; c=**cross**(a,b)	c =　4　2　2

3.4 内置函数与函数值计算

内置函数(Built-in Functions)是MATLAB已经编制好的函数(子程序),前面已经调用了一些内置函数,例如size、length、det、dot、cross、exp等。MATLAB内置了绝大多数常用的数学函数与平面(二维)、立体(三维)的作图函数。要知道有关的函数名及其功能,以及如何使用该函数就需要用到两个非常有用的指令。

绝大多数内置函数的参数都是矩阵(包括向量与数)。在计算函数值的时候,对矩阵的每个元素进行计算。

3.4.1 两个重要搜索指令

1. lookfor +(一个)关键词

当不知道函数名时,要用"lookfor 关键词"去搜索。例如,想寻找计算与对数(logarithm)有关的函数,就可以用指令"lookfor logarithm"去搜索。

输入指令	>> lookfor logarithm
输出结果(列出了部分函数)	
log	Natural logarithm.
log10	Common (base 10) logarithm.
log2	Base 2 logarithm and dissect floating point number.
log	Symbolic matrix element-wise natural logarithm.
log10	Symbolic matrix element- wise common logarithm.
log2	Symbolic matrix element-wise base-2 logarithm.

若要找指数(exponential)函数,使用指令"lookfor exponential"。当然也可以用关键词的一部分,如"exp"去搜索,但会得到很多额外与指数函数无关,但与"exp"有关的结果,例如与"expression"或"exponent"有关的结果。

2. help+函数名

知道函数名时,要用"help **函数名**"去搜索。

从上面搜索结果的说明中,我们知道如果要计算的是自然对数函数值,就用指令"help log",得到下面的结果。

输入指令	输出结果
>> help log	log - 自然对数 此 MATLAB 函数 返回数组 X 中每个元素的自然对数 ln(x)。 Y = log(X) 另请参阅 exp, log10, log1p, log2, loglog, logm, reallog, semilogx, semilogy log 的文档 名为 log 的其他函数

要得到更详细的说明与使用例子，单击带有下画线的字符“log 的文档”。从上面可以看到，指令“help log”还显示了与自然对数有联系的其他函数。例如，常用对数 log10（$\log_{10}$），以 2 为底的对数 log2（$\log_2$），指数函数 exp 以及名为 log 的其他函数等。单击带有下画线的字符可以找到与自然对数有联系的其他函数。

关于如何使用“help”，可用“help help”来查询。

3. 常用的内置初等函数

用指令“help elfun”（elementary function）可找到 MATLAB 中内置的初等函数表。单击表上的函数名，相当于用指令“help 函数名”。

（1）**绝对值函数**：|x|→ abs(x)。x 为复数时，等于 x 的模长。

例如：

输 入 指 令	输 出 结 果
>> x=[-2,　0,　3,　3+4i];　abs(x)	ans =　2　　0　　3　　5

对于两个矩阵是否相等，当矩阵规模较小时，可以显示出来用眼睛检查。但矩阵规模较大时，由于数值计算中不同方法的数值误差，在理论上应该相等的两个矩阵，实际上会有差别，这时可以联合使用 abs 与 sum 来检查这一差别是否在容许范围之内。

例如：

输 入 指 令	输 出 结 果
>>a=[1,0,-2]; b=[-3,2,4]; D1=a'*b; D2=b'*a; DIF=**sum(sum(abs**(D1-D2')))	DIF = 0

（2）幂函数：x^a→ x^a；$\sqrt{x}$ → sqrt(x) 或 x^(1/2)。这里，x 可以是复数。

例如：

输 入 指 令	输 出 结 果（以下删去了小数点后面的所有 0 及 0 i）
>> x=[-4, 4, 3+4i] ; x2=x.^2 ,　　xs=x.*x, xh=x.^(1/2),　xq=sqrt(x)	 x2 = xs　=　16.　　16.　-7. +24i xh = xq　=　0+2i　2.　2+ 1i

（3）三角函数：函数名后＋“d”，即参数 x 的单位用度的标以 *，否则用弧度。

正弦 sin(x)，*sind(x)；余弦 cos(x)，*cosd(x)。

正切 tan(x)，*tand(x)；余切 cot(x)，*cotd(x)。

正割 sec(x)，*secd(x)；余割 csc(x)，*cscd(x)。

例如：

输 入 指 令	输 出 结 果
>> S1=sin([pi/2,-pi/4,pi]), S2= sind([90,-45,180]) ,	S1=1.0000　-0.7071　0.0000 S2=1.0000　-0.7071　0

续表

输 入 指 令	输 出 结 果
T=tan(pi/4), C=csc(pi)	T = 1.0000 C = 8.1656e+15

(4) **反三角函数**：其函数名是在对应的三角函数名前加字母 a，标以 * 的表示输出量的单位为度，否则为弧度。

反正弦 asin(x)，* asind(x)；反余弦 acos(x)，* acosd(x)。

反正切 atan(x)，* atand(x)；反余切 acot(x)，* acotd(x)。

反正割 asec(x)，* asecd(x)；反余割 acsc(x)，* acscd(x)。

例如：

输 入 指 令	输 出 结 果
>> As=asin([1, -sqrt(2)/2, 0]), Asd= asind([1, -sqrt(2)/2, 0])	As= 1.5708 -0.7854 0 Asd= 90.0000 -45.0000 0

(5) 指数函数：e^x → exp(x)；　$a^x = e^{x\ln a}$ → exp(x * log(a))。

例如：

输 入 指 令	输 出 结 果
>> Ex=exp([log(2), 1, -1]), a=2; x= 3; ax=exp(x * log(a)), ax1=a^x	Ex = 2.0000 2.7183 0.3679 ax = 8.0000 ax1= 8

(6) 对数函数：以 e 为底 → log(x)；以 10 为底 → log10(x)；以 2 为底 → log2(x)。这里，x 可以是复数和负数。

例如：

输 入 指 令	输 出 结 果
>> L1=log(-1), AL1= abs(log(-1)), L2=log2(4), L10=log10(0.01), LP=log(1+i)	L1 = 0.0000 + 3.1416i AL1 = 3.1416 L2 = 2, L10 = -2 LP = 0.3466 + 0.7854i

3.4.2 取整的内置函数

不少计算机语言中的变量分为整型与实型。这样整除运算就可以运用通常的除法运算符，只要 a 与 b 在程序一开始的说明语句中都说明为整型变量，得到的结果就是整数。例如，8 整除以 3，得到 2。MATLAB 的变量不分整型与实型，要整除运算不能用通常的除法运算符，但它有内置函数，可以实现同样的目的。它们的实参是矩阵(包括向量和数)，对每个元素取整；矩阵的元素可以是复数，此时它们把此元素的实部与虚部分别

取整。

(1) round(x)：四舍五入取整。

(2) floor(x)：向下取整或向负无穷方向取整。等于数轴上左侧最接近 x 的整数。

(3) ceil(x)：向上取整或向正无穷方向取整。等于数轴上右侧最接近 x 的整数。

(4) fix(x)：截尾取整或向零取整。等于在数轴上朝原点方向最接近 x 的整数。

如表 3.2 所示是对向量 $\boldsymbol{x}$ 的值与以上 4 个取整函数的结果比较。

表 3.2　取整函数结果比较

$\boldsymbol{x}$	[−1.9,	−0.2,	3.4,	5.6,	7.0,	2.4+3.5i]
round(x)	−2	0	3	6	7	2+4i
floor(x)	−2	−1	3	5	7	2+3i
ceil(x)	−1	0	4	6	7	3+4i
fix(x)	−1	0	3	5	7	2+3i

内置函数的参数可以是数值，也可以是表达式，但涉及的变量必须先赋值。

例 3.1　已知年 y 与从元旦到某日的天数 c，求那一天是星期几。例如，2017 年的春节(1 月 28 日)是星期几？其中，$c=28$。

解：令 $x=y-1$，先计算 $s=x+\lfloor x/4\rfloor-\lfloor x/100\rfloor+\lfloor x/400\rfloor+c$，其中，$\lfloor z\rfloor$ 为 z 取整数部分；然后作带余除法 $s\div 7$，余几即为星期几(余数=0：星期日)。

作两个整数 s(被除数)与 t 的带余除法的正确方法是：先求出它的整数部分 n(上述被除数 x，$s>0$，所以用向下取整 floor 或截尾取整 fix 均可)。再求出余数部分 $r=s-nt$。输入指令串：

输 入 指 令	输 出 结 果
>> c=28;　x=2017-1; s=x+fix(x/4)-fix(x/100)+fix(x/400)+c n=fix(s/7),　r=s-n * 7	 s=2533 n=361, r=6, 星期六

如果先用通常的除法求出 $s/7$ 的带小数的值 z，再取它的整数部分 n 与小数部分(分数 6/7 的小数值)b，再作“b * 7”，用 fix 或 floor 转换为整数，必须十分小心。例如，把上面第 3 行的两个指令改为：

输 入 指 令	输 出 结 果
>> z=s/7,　n=fix(z),　b=z-n, d= b * 7,　r1=fix(d)	z= 361.8571,n=361,b= 0.8571 d=6.0000,　　r1=5

实际上，b 的准确值是 6/7，它是循环小数，循环节是 857143。若在“format long”下运行，显示 d =5.99999999999983。从而 r1 的结果会是 5。所以在用 fix 或 floor 取整前应先加一个 0.1，而在用 ceil 取整前应先减去一个 0.1，即

```
r1=fix(d+0.1), r2=floor(d+0.1), r3=ceil(d-0.1)
```

都会得到正确的结果 6。

此外，当 s 与 t 都是正数，而且仅要求余数时，可以用内置函数 rem(s,t) 或 mod(s,t)。详情可用 help 来查询。

如果要的是 s 除以 7 所得余数，程序的最后一行可改为“r＝rem(s,7)”或“r＝mod(s,7)”都会得到正确答案 6。

3.5 随机数的产生

有各种类型的随机数，这里只介绍两种。其一为一致分布(uniformly distributed)的随机数，另一为正态分布(normally distributed)的随机数。其他类型的随机数可用指令 **lookfor random** 来获得详情。

3.5.1 一致分布的随机数

一致分布的随机数由内置函数 rand()产生。

1. 内置函数 rand(m,n)和 rand([m,n])

内置函数 rand(m,n)和 rand([m,n])都得到 $m\times n$ 矩阵，其元素是单位区间 [0,1] 内一致分布的随机数。rand(n) 得到 $n\times n$ 方阵，其元素是单位区间[0,1]内一致分布的随机数。rand(1,1)与 rand()无参数的结果都得到一个在单位区间[0,1]内一致分布的随机数。

例如：

输 入 指 令	输 出 结 果
>> x=rand(1,10) %在 [0,1] 内得到的 10 个随机数	x= 0.0975 0.2785 0.5469 0.9575 0.9649 0.1576 0.9706 0.9572 0.4854 0.8003

与“hist(x)”配合会得到直方图。例如：

```
>>x=rand(1,1000);  %不要忘了加分号!
   hist(x)    %得到直方图如图 3.1 所示
```

如果把分量数增加到 10 万个，则直方图的所有矩形的顶端都差不多高。

2. 一致分布的随机数

要产生区间 [a,b] 内(10 个)一致分布的随机数(行向量)，则应该用下列语句，其中“*”表示“乘以”的运算符：

```
r = a + (b-a) * rand(1,10);
```

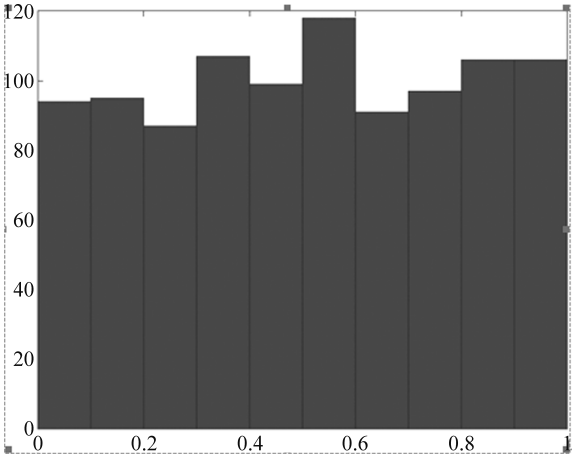

图 3.1　一致分布的随机数的直方图

内置函数 rand 的其他调用方法用指令“help rand”查看详情。

3.5.2　正态分布的随机数

1. 内置函数 randn(m,n) 和 randn([m,n])

内置函数 randn(m,n) 或 randn([m,n]) 都得到 $m \times n$ 矩阵，其元素是标准正态分布的随机数。期望值为 0($\mu=0$)，标准差为 1($\sigma=1$)。正态分布函数的图形是一条钟形曲线，直方图的外包曲线接近钟形曲线。randn(n)得到 $n \times n$ 方阵，其元素是标准正态分布的随机数。

例如：

```
>>xn=randn(1,1000);  %不要忘了加分号!
  hist(xn)    %得到正态分布直方图如图 3.2 所示
```

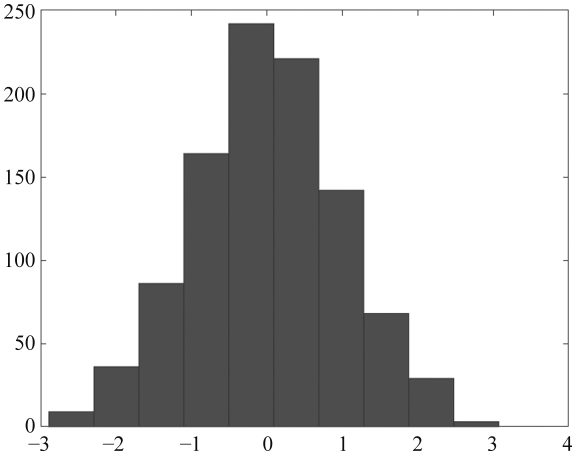

图 3.2　标准正态分布的随机数的直方图

2. 正态分布的随机数

要产生期望值为 a($\mu=a$),标准差为 b($\sigma=b$)的(100 个)正态分布的随机数(行向量)则应该用下列语句:

```
rn = a + b * randn(1,100);
```

3.6 创建和运行 M 文件

前面讲的都是直接在指令窗口中调用内置函数,更必须学会编制程序并保存为 MATLAB 的文件(M 文件)和运行它。

3.6.1 创建函数子程序文件

MATLAB 有一些内置的 function(子)程序,例如 dot、sum、cross 等内置函数,可以被其他的 function 程序或主程序调用。虽然 function 程序并不都包含函数,但一般还是称它们为函数(子程序),以区别于主程序。存放的 M 文件称为“函数文件”。创建一个函数文件的好处就在于我们可以用一个其他文件来调用它。在多个程序中,如果都有一组仅变量名不同而功能一样的语句,或者在一个程序中反复出现一组(仅变量名不同而功能一样的)语句,就可以把这组语句编成函数子程序,那些“变量”就成为名字相同的形式(输入、输出)参数。然后被多个程序调用或被一个程序反复调用。

用“>> help function”以及单击所显示的信息的最底下的“doc_ function”可发现编制函数程序的方法如下。

(1) 函数头行:function [out1, out2, …] = funname(in1, in2, …)。

注意:

- 函数子程序(头行)不能在指令窗口的提示符下输入。
- out1, out2, …是输出(output)形式变量(名)或形式参数,简称输出形参。只有一个变量时,可省去方括号。例如,function y = funname(in1, in2, …)。
- funname 是我们取的函数名字,也是保存时计算机自动取的 M 文件名。
- in1, in2, … 是输入(input)形参。圆括号不能省去;即使没有输入变量,也要输入空的圆括号。例如,function y = Qudr(),其中,Qudr 是函数名。

(2) 函数表达式。

把函数表达式,即因变量(输出变量)与自变量(输入变量)的关系式,写在执行语句中。

例 3.2 下面是一个自编的计算二次多项式值的 MATLAB 函数。

```
%求多项式 ax^2+bx+c 的函数值,x 是自变量,y 是因变量。
%调用格式: y=Qudr(a,b,c,x);
function y=Qudr(a,b,c,x);
y=a * x * x+b * x+c;
return;
```

前面已经提到，“%”后面的是注释(comment)语句(输入“%”后，计算机自动会把 % 与此后的说明变成绿色)，不执行任何计算。一个语句前加上“%”，它就被“屏蔽”了，不再被执行。初学者可以把前两行的说明复制到新建文件中。

1. 新建一个文件

单击“主页”左上角“新建(脚本)”按钮，将在编辑器中新建一个名为“Untitled”的文件，在“编辑器”最右处列出。图 3.3 显示了一个在已有 TRem.m 文件下新建一个文件的状态，此时，“编辑器”里有两个文件，左边是原有打开的 TRem.m 文件，新建文件 Untitled 排在最右面。如果原来没有已打开文件，那么 Untitled 将在“编辑器”最左处列出。如果连续单击“新建(脚本)”按钮，将在“编辑器”右边依次列出“Untitled2”“Untitled3”等。

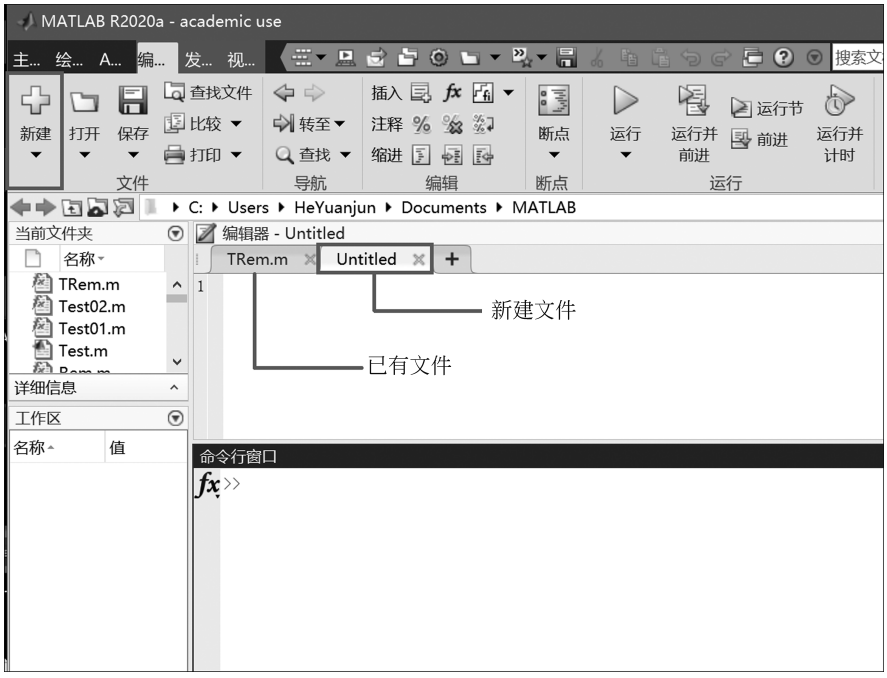

图 3.3　新建一个文件

2. 保存一个文件

“Untitled”“Untitled2”“Untitled3”等是系统给出的默认文件名，可以在保存时将文件名改成设计者指定的名字存放。如图 3.4 所示是将“Untitled”保存为“Qudr.m”文件。

如果保存成功，“编辑器-Untitled”下一行原来的“Untitled”会自动改为“Qudr.m”。

如果把例 3.2 的两行说明也输入到 Qudr.m 文件中(输入到 function 的前面或后面都可以)，在命令窗口中输入“help Qudr”后，两行说明就会显示。

3. 打开已存在的 M 文件

要打开(比如想修改)一个已存在的 M 文件时，可单击左上角的“打开”图标(图 3.5)，系统会在“最近使用的文件”中列出最近的文件，选择其中需要打开的文件即可。

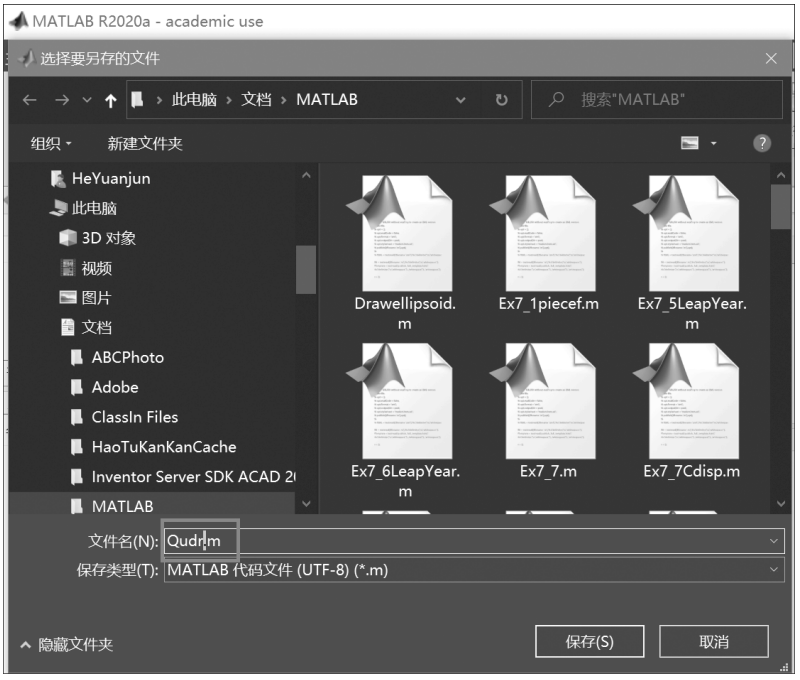

图 3.4　保存一个文件

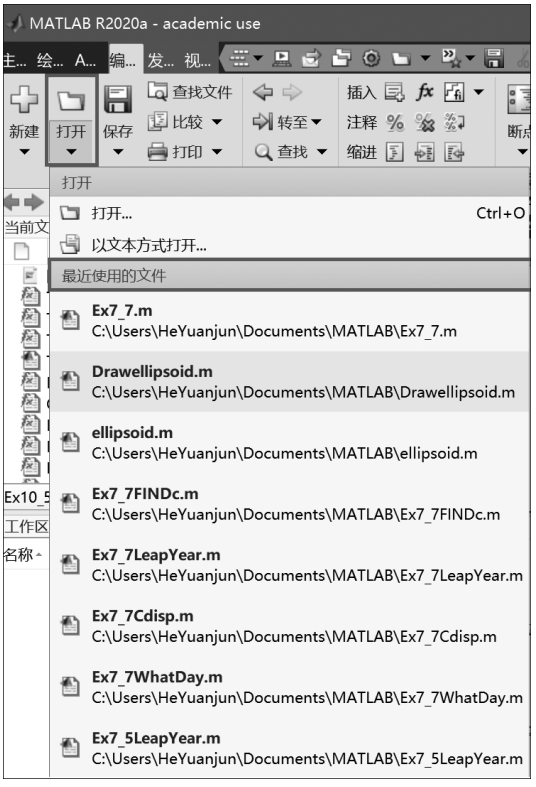

图 3.5　打开已有的文件

3.6.2　运行 M 文件

1. 运行函数子程序

有以下两种方法来运行函数子程序。

(1) 在命令行(指令)窗口中输入与前面“函数头行”相对应的指令。

```
[OUT1, OUT2, …] = funname(IN1, IN2, …)
```

其中,OUT1, OUT2, …是输出实参,可以与输出形参 out1, out2, …不同名,但要一一对应;IN1, IN2, …是输入实参,可以与输入形参 in1, in2, …不同名,但也要一一对应。输入实参,可用赋值语句,也可以把数值直接写在 funname 后的圆括号中。

例 3.3　下面是一个简单的执行 Qudr.m 函数子程序的例子。注意大小写(Y 与 y)是不同的变量。

输 入 指 令	输 出 结 果
>> a=1; b=2; c=3; d=1; y=Qudr(a,b,c,d)	y=6
>> Y= Qudr (1,2,3,1)	Y=6

(2) 也可以把上述两行指令串各自写成 M 文件(称为主程序)并分别取名为 Test.m 与 Test0.m,然后在指令窗口中输入主程序的名字(省去扩展名“.m”),即可调用(运行)函数子程序 Qudr.m。

例 3.4　下面列出了这两个主程序与调用指令以及输出结果。

输 入 指 令	主程序文件	输 出 结 果
>>Test	a=1; b=2; c=3; d=1; y=Qudr(a,b,c,d)	y=6
>>Test0	Y= Qudr (1,2,3,1)	Y=6

2. 运行主程序

除了上面在指令窗口中输入主程序(其中可以没有调用函数子程序的语句)的名字来运行主程序,还可以在“运行界面”上来运行。

通过“打开”功能,找到想要运行的 M 文件(例如“Test.m”),将在“当前文件夹”中出现该文件(如果显示找不到该文件,则需要设置路径),用右键单击它,在弹出菜单中单击“运行”,就会在“命令行窗口”(数值运算)或弹出的 Figure 窗口(图形输出)中输出结果(图 3.6)。

3.6.3　创建调用函数的 M 文件与输入数据

函数子程序 Qudr.m 被其他函数子程序或主程序调用时,其中的赋值语句要改为输入语句。这样就不必在改变变量值的时候,去修改调用语句。下面介绍几种方法来输入

图 3.6　运行 M 文件及其结果

数据。

(1) 应用内置函数 input('prompt')与键盘输入数据。

例 3.5　建立一个名为"Test1.m"用键盘输入数据的 M 文件：

```
%这是一个应用内置函数 input('prompt')与键盘输入数据
%并调用函数子程序 Qudr.m 的测试程序
a=input('a=');  b=input('b=');  c=input('c=');  x=input('x=');
y = Qudr (a,b,c,x)      %调用函数子程序 Qudr.m
```

写在单引号中的提示(prompt)字符串 'a=','b=','c=' 与 'x=' 会逐个显示在指令窗口中,紧接着后面用键盘输入该变量的值,这个值赋给"="号前的变量。

以下是指令窗口中输入和输出的内容。在输入指令栏,"a=""b=""c=""d="是系统自己输出的,其他都是操作者输入的。最后在"输出结果"栏得到主程序 Test1 的最后一行调用语句"y=Qudra(a,b,c,x)"(后面没有分号)后的结果。

输 入 指 令	输 出 结 果
>> Test1 a=2 b=-1 c=3 x=-2	y = 13

(2) 向量输入法：把上面的 4 个数作为一个向量的元素。

例 3.6　以下的 Test2.m 文件中，只有一句输入语句，赋给向量 ***p***。

```
%Name:  Test2.m,  改为输入向量 p=[a,b,c,x]
%调用函数子程序 Qudr.m 的测试程序
p=input('p=[a,b,c,x]=');
y=Qudr(p(1), p(2), p(3), p(4));
```

这里向量 ***p*** 有 a，b，c 与 x 四个分量。在提示字符串后要输入“[2，－1，3，－2]”。必须用方括号表示向量，各分量之间用逗号或空格来分隔。此时，a＝p(1)＝2，b＝p(2)＝－1，c＝p(3)＝3，x＝p(4)＝－2。所以，实参是 ***p*** 的 4 个分量，用它们代入 4 个形参。指令窗口上显示的全过程为：

输 入 指 令	输 出 结 果
>> Test2; p=[a, b, c, x]=[2, -1, 3, -2] >> y　% Test2.m 中调用语句后有分号！	 %“p=[a, b, c, x]=”是指令窗口显示的 y =13

(3) 用指令“load ＋ 输入文件名”来输入数据。

例 3.7　打开“文本文件”，新建一个名为“Test3Data.txt”(存为 **txt**)的输入文件，其中与输入数据有关的第 2 行内容为“2，－1，3，－2”(或用空格分隔)，而第 1 行是以“%”开头的说明，写上是为了以后能知道这个文件的功能，输入指令不会读入，见图 3.7。

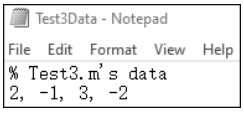

图 3.7　名为“Test3Data.txt”的数据文件

这 4 个数值是以输入文件名(去掉扩展名)为向量名(即 Test3Data)的 4 个分量。然后创建 M 文件 Test3.m(第 2 个语句最后无分号时才会自动输出 y 的值)。

```
%Name:  Test3.m,  用 load Test3Data.txt 输入向量 Test3Data
%调用函数子程序 Qudr.m 的测试程序
load Test3Data.txt
y=Qudr(Test3Data (1), Test3Data (2), Test3Data (3), Test3Data (4))
```

注意，Test3Data.txt 与 Test3.m 要存放在同一个子目录(文件夹)中。运行时，在命令窗口中输入“>> Test3”，马上得到答案：y＝13。

用这种方法，在需要根据不同数据来计算函数值的时候，只要把有关的数据文件复制为 Test3Data.txt，然后运行 Test3.m。