国外计算机科学经典教材

Scientific Computing with MATLAB

MATLAB 科学计算

(德) Alfio Quarteroni 著
Fausto Saleri
李敏波 译

清华大学出版社

国外计算机科学经典教材

MATLAB 科学计算

(德) Alfio Quarteroni
Fausto Saleri 著

李敏波 译

清华大学出版社

北 京

Alfio Quarteroni, Fausto Saleri
Scientific Computing with MATLAB
EISBN: 3-540-44363-0

北京市版权局著作权合同登记号　图字：01-2004-1049

图书在版编目(CIP)数据
MATLAB 科学计算/(德)夸特罗尼，(德)色拉瑞著；李敏波译.—北京：清华大学出版社，2005.1
书名原文：Scientific Computing with MATLAB
（国外计算机科学经典教材）
ISBN 7-302-09302-4
Ⅰ.M…　Ⅱ.①夸…②色…③李…　Ⅲ.计算机辅助计算－软件包，MATLAB－教材　Ⅳ.TP391.75
中国版本图书馆 CIP 数据核字(2004)第 089269 号

出 版 者：清华大学出版社　　**地　　址**：北京清华大学学研大厦
http://www.tup.com.cn　**邮　　编**：100084
社 总 机：010-62770175　**客户服务**：010-62776969
组稿编辑：曹　康
文稿编辑：侯　彧
封面设计：康　博
版式设计：康　博
印 刷 者：北京市通州大中印刷厂
装 订 者：三河市化甲屯小学装订二厂
发 行 者：新华书店总店北京发行所
开　　本：185×260　**印张**：14.75　**字数**：306 千字
版　　次：2005 年 1 月第 1 版　2005 年 1 月第1 次印刷
书　　号：ISBN 7-302-09302-4/TP・6528
印　　数：1～4000
定　　价：26.00 元

出 版 说 明

近年来，我国高等学校的计算机学科教育进行了较大的改革，急需一批门类齐全、具有国际水平的计算机经典教材，以适应当前的教学需要。引进国外经典教材，可以了解并吸收国际先进的教学思想和教学方法，使我国的计算机学科教育能够与国际接轨，从而培育更多具有国际水准的计算机专业人才，增强我国信息产业的核心竞争力。Pearson、Thomson、McGraw-Hill、Springer、John Wiley 等出版集团都是全球最有影响的图书出版机构，它们在高等教育领域也都有着不凡的表现，为全世界的高等学校计算机教学提供了大量的优秀教材。为了满足我国高等学校计算机学科的教学需要，我社计划从这些知名的国外出版集团引进计算机学科经典教材。

为了保证引进版教材的质量，我们在全国范围内组织并成立了“清华大学计算机外版教材编审委员会”（以下简称“编委会”），旨在对引进教材进行审定、对教材翻译质量进行评审。“编委会”成员皆为全国各类重点院校教学与科研第一线的知名教授，其中许多教授为各校相关院、系的院长或系主任。“编委会”一致认为，引进版教材要能够满足国内各高校计算机教学与国际接轨的需要，要有特色风格，有创新性、先进性、示范性和一定的前瞻性，是真正的经典教材。为了保证外版教材的翻译质量，我们聘请了高校计算机相关专业教学与科研第一线的教师及相关领域的专家担纲译者，其中许多译者为海外留学回国人员。为了尽可能地保留与发扬教材原著的精华，在经过翻译和编辑加工之后，由“编委会”成员对文稿进行审定，以最大程度地弥补和修正在前面一系列加工过程中对教材造成的误差和瑕疵。

由于时间紧迫和能力所限，本套外版教材在出版过程中还可能存在一些不足和遗憾，欢迎广大师生批评指正。同时，也欢迎读者朋友积极向我们推荐各类优秀的国外计算机教材，共同为我国高等学校的计算机教育事业贡献力量。

清华大学出版社

国外计算机科学经典教材

编审委员会

前　言

本书主要介绍科学计算方法，列举了若干类数学问题的数值解法，这些问题利用解析方法一般难以求解。书中主要介绍了如何计算连续函数的零点和积分，如何求解线性系统，如何利用多项式法处理函数逼近，以及如何构造微分方程精确的近似解问题。

为此，第 1 章中主要介绍了计算机在对实数、复数、向量和矩阵进行存储和运算时所遵循的运行规则。

为了使所叙述的内容更加生动和具体，书中自始至终都结合 MATLAB 编程环境来进行阐述。读者会逐渐地熟悉 MATLAB[1]主要的函数、语句和结构。本书对每章中提到的算法都将给出具体程序实现，以便读者对这些算法的理论性能，如稳定性、准确性和复杂性作出实时的定量评估。书中对于在习题和例题中提出的若干问题也给出了解决办法，这些问题多来自于某些特定的实际应用。

在每章的最后，都有一节专门指出未提及的相关内容，同时给出相关参考文献，以便读者进行更深入的理解和学习。

书中会经常提到参考书[QSS00]，它对本书所提到的问题进行了更深入的讲解，并对有关结论进行了理论证明。而关于 MATLAB 更为详尽的介绍，我们向读者推荐［HH00］。本书中出现的所有程序均可从 mox.polimi.it/Springer 站点下载。

本书对读者并无特别要求，读者只需具备微积分的基础知识即可阅读本书。

尽管如此，在第 1 章中还是对微积分和几何学中的基本概念进行了回顾，这些概念在全书中都有广泛的应用。

最后，作者非常感谢海德尔堡 Springer-Verlag 的 Thanh-Ha Le Thi， Springer-Italia 的 Francesca Bonadei 和 Marina Forlizzi 在整个项目中给予的友好合作。同时感谢 Cardiff 大学的 Eastham 教授对全书手稿所做的文字编辑工作，他提出的许多宝贵意见，对全书的多处内容进行了改进。

Alfio Quarteroni，Fausto Saleri

2003 年 5 月于米兰和洛桑

1　MATLAB 是 The MathWorks 公司的商标，公司地址：24 Prime Park Way, Natick MA 01760, 电话：001+508-647-7000，传真：001+508-647-7001。

目 录

第1章 绪 论

本书将系统地应用基本的数学概念，这些概念读者应该已经知道，但可能不会立即回想起它们。

因此本章将主要帮助读者回顾这些概念，并介绍属于数值分析领域的新概念。同时将使用 MATLAB(MATrix LABoratory)工具来帮助阐述这些概念的含义和作用，MATLAB 提供了用于科学计算的可编程、可视化的集成环境。1.6 节对 MATLAB 进行简要的介绍，这些介绍足以使读者读懂书中的后续内容，但我们还是向有兴趣的读者推荐手册［HH00］，该手册对 MATLAB 语言进行了完整的描述和说明。

本章将扼要地介绍微积分学、线性代数学和几何学中的基本概念，不过将采用一种更有利于把它们应用到科学计算之中的方式来介绍它们。

1.1 实数

许多人都知道实数集 **R**，但也许很少人能知道计算机处理实数的方式。一方面，计算机的资源是有限的，因此它只能表示出实数集 **R** 的有限维的子集 F。子集 F 中的数被称为浮点数。另一方面，正如将要在 1.1.2 小节中所讲的，用于描述 F 的性质与用于描述实数集 **R** 的性质是不同的。原因是任意实数 x 原则上都被机器截取了一定位数，这就产生了一个新的数值(称为浮点数)，记为 $fl(x)$，而这个浮点数并不一定要与原来的数 x 保持完全一致。

1.1.1 实数的表示

为了更好地理解集合 **R** 和 F 之间的区别，我们通过几个 MATLAB 实例来说明计算机(例如一台 PC 机)处理实数的方式。至于是采用 MATLAB 语言，还是采用其他的语言，完全根据使用的方便程度来决定。实际上计算结果主要取决于计算机的工作方式，而很少取决于所采用的编程语言。考虑有理数 $x=1/7$，它表示成小数为 $0.\overline{142\ 857}$。由于它的小数位数是无限的，因此它的十进制表示也是无限的。为了得到它的计算机表示，在提示符后输入分数 1/7，得到：

```
>>1/7
ans=
  0.1429
```

得到的结果是一个小数点后只有四位的数，而且它的最后一位与初始数字的对应位不同。如果把输入换成 1/3，我们将会得到 0.3333，此时第四位小数仍然与初始数值的对应位保持一致。这是由于实数在计算机内是四舍五入的。这意味着：首先，只能返回一个固定位数的十进制数；其次，对于最后一位数字来说，如果与其相邻的下一位数字大于或等于 5，那么最后一位数字便加 1。

需要注意的一点是，只使用一个 4 位十进制数去表示一个实数是存在问题的。实际上，数字在内部是用 16 位十进制数来表示的，我们所作看到的结果也只是 MATLAB 多种输出格式中的一种。同样的一个数值，根据其所做的特定格式的说明的不同，便可以得到它的不同表示方式。例如，对于数 1/7 来说，有下列一些可能的输出格式：

format long	输出	0.14285714285714
format short e	输出	1.4286e－01
format long e	输出	1.428571428571428e－01
format short g	输出	0.14286
format long g	输出	0.142857142857143

可以看出一些表示方式比其他几种方式更加接近计算机的内部表示。实际上，计算机通常采用下列方式来存储一个实数：

$$x=(-1)^s\cdot(0.a_1a_2\cdots a_t)\cdot\beta^e=(-1)^s\cdot m\cdot\beta^{e-t},\quad a_1\neq 0 \tag{1.1}$$

其中 s 只能取 0 或 1，β(大于或等于 2 的整数)是计算机采用的底数，整数 m 称为尾数，尾数的长度 t 是所存储数据 a_i 的最大位数，范围介于 0 和 $\beta-1$ 之间，整数 e 称为指数。长型 e 数据采用的表示方法就非常类似于这种形式，其中 e 代表指数，表示指数大小的数位前面加有正负运算符号，写于字符 e 的右侧。式(1.1)中所给数值的小数点位置是不固定的，称之为浮点数。数位 $a_1a_2\cdots a_p$(其中 $p\leqslant t$)称为数 x 的有效数位。

$a_1\neq 0$ 这一条件保证了同一个数值不会出现多种表示形式。例如，如果没有这一限制条件，数 1/10 既可以表示为(以 10 为底)0.1×10^0，也可以表示成 0.01×10^1 等其他形式。

因此，集合 F 可以由底数 β，有效位数 t 的大小和指数 e 的变化范围(L，U)(其中 $L<0$，$U>0$)完全描述确定。此时集合可表示为 $F(\beta, t, L, U)$。例如，在 MATLAB 中集合 F 可以表示为 $F(2, 53, -1021, 1024)$(实际上，以 2 为底的含有 53 个有效数位的数值与 MATLAB 用长型所显示出来的以 10 为底的含有 15 个有效数位的数值是一致的)。

当用集合 F 中的数 $fl(x)$来代替一个不等于 0 的实数 x 时，不可避免地会造成舍入误差。由于满足下式：

$$\frac{|x-fl(x)|}{|x|} \leqslant \frac{1}{2}\varepsilon_M \tag{1.2}$$

所以舍入误差一般都比较小，公式中 $\varepsilon_M = \beta^{1-t}$ 代表距 1 最近的浮点数与 1 之间的差值。注意 ε_M 是 β 和 t 的函数。在 MATLAB 中，可以通过 eps 命令来得到 ε_M，结果为 $\varepsilon_M = 2^{-52} \simeq 2.22 \times 10^{-16}$。需要指出，在式(1.2)中描述的是 x 的相对误差，它比绝对误差$|x-fl(x)|$有着更实际的意义。实际上，后者未考虑 x 量值本身的作用，而前者考虑了这个问题。

0 不属于集合 F，因为如果 0 属于集合 F，则在式(1.1)中 $a_1=0$，此时必须对它进行单独处理。此外，由于 L 和 U 都是有限数，所以那些绝对值极大或极小的数值都不能表示出来。如果精确地描述，F 集合中最小的和最大的正实数分别表示为：

$$x_{\min} = \beta^{L-1}, \quad x_{\max} = \beta^{U}(1-\beta^{-t})$$

在 MATLAB 中，可以利用 realmin 和 realmax 函数来分别得到它们，结果如下：

$$x_{\min} = 2.225\ 073\ 858\ 507\ 201 \times 10^{-308}$$
$$x_{\max} = 1.797\ 693\ 134\ 862\ 315\ 8 \times 10^{+308}$$

遇到比 $x_{\min}$ 还小的正数时会输出下溢信息，同时把这个数按 0 处理或采用特定方式处理(可参照［QSS00］第二章)。对于那些比 $x_{\max}$ 还大的数，会给出上溢信息，并把它存储在变量 Inf 中(计算机用 Inf 变量来表示无穷大+∞)。

集合 F 中的元素在 $x_{\min}$ 处更为密集，在接近 $x_{\max}$ 处密集度减小。实际上，集合 F 中与 $x_{\max}$ 距离最近的数(在 $x_{\max}$ 的左侧)和与 $x_{\min}$ 最近的数(在其右侧)分别是：

$$x_{\min}^{-} = 1.797\ 693\ 134\ 862\ 315\ 7 \times 10^{+308}$$
$$x_{\min}^{+} = 2.225\ 073\ 858\ 507\ 202 \times 10^{-308}$$

可见 $x^{+}_{\min} - x_{\min} \simeq 10^{-323}$，而 $x_{\max} - x^{-}_{\max} \simeq 10^{292}$(!)。但从式(1.2)可以看出这两种情况下的相对距离都是很小的。

1.1.2　浮点数的运算

由于 F 集合只是集合 **R** 的一个特定的子集，所以在集合 **R** 的运算中适用的方法并不完全适用于浮点数的代数运算。具体来讲，交换律对于加法运算($fl(x+y)=fl(y+x)$)与乘法运算($fl(xy)=fl(yx)$)仍然适用，但是其他规则如结合律和分配律已不再适用了。而且 0 也不再是惟一的。实际上，当把变量 a 赋值为 1 并执行下面

的表达式时：

```
>>a=1; b=1; while a+b~=a; b=b/2; end
```

可见，循环每执行一次，只要此时 a 与 b 相加之和不等于 a，变量 b 便会被除以 2。如果按通常的方式对一个实数进行这种操作，这个程序将永远不会停止，而在本例中，程序在运行一定次数以后便会结束，并返回 b 的值：1.1102e－16=ε_M/2。因此我们至少可以看到：虽然 b 此时并不等于 0，但却有 a+b=a 成立。这种情况之所以能够发生，原因就在于集合 F 是由相互独立的数值所构成的；当把两个数 a 和 b 相加时，其中 b<a，并且 b 小于 ε_M 时，总会得到 a+b 等于 a 的结果。

当上溢或下溢的情况发生时，结合律便不再适用了。设 a=1.0e+308，b=1.1e+308，c=－1.001e+308，用下面两种方式分别来进行加法运算，可以得到：

$$a+(b+c)=1.0990e+308,\quad (a+b)+c=Inf$$

这个例子清楚地说明了当两个符号相反、绝对值相近的数值进行加法运算时会出现什么情况。此处得到的结果是错误的，这种情况被称为有效数位的丢失或消去。例如在计算表达式((1+x)－1)/x(很明显，对于任意 $x\neq0$，结果应为 1)的结果时：

```
>>x=1.e－15; ((1+x)－1)/x
ans=
  1.1102
```

可见结果很不准确，相对误差竟大于 11%！

再看另一个数值相消的例子，计算下列函数：

$$f(x)=x^7-7x^6+21x^5-35x^4+35x^3-21x^2+7x-1 \tag{1.3}$$

在横轴上区间[$1-2\times10^{-8}$，$1+2\times10^{-8}$]内均匀分布的 401 个点上，得到的是如图 1.1 所示的杂乱曲线(实际上该运算就是$(x-1)^7$，它是连续的，在 x=1 这样小的邻域内等于零函数)。在 1.4 节，会给出描绘曲线所用的命令。

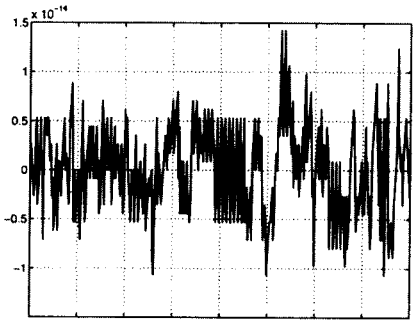

图 1.1　由消去误差导致的函数(1.3)的波动曲线

最后需要注意的一个比较有趣的问题是：在集合 F 中并没有包括那些含义不明确的表达形式，如 0/0 或∞/∞，遇到这类表达形式，将会给出提示信息 NaN(not a number，在 MATLAB 中用 NaN 来表示)，对此声明通常的数值计算规则对它们并不适用。

注 1.1 尽管舍入误差通常都很小，但如果它们在长型和复数型的运算规则中不断累加时，也可能导致灾难性的后果。有两个真实的事例足以说明问题，一个是发生于 1996 年 6 月 4 日的阿里娅娜火箭的爆炸事故，这次事故就是由于控制台上计算机的溢出导致的；另一个是 1991 年海湾战争期间，曾有一枚美国的爱国者导弹落入了美军军营，这也是由于计算导弹飞行轨道时产生的舍入误差引起的。

另一个例子倒没有多少灾难性(但同样会引起不少麻烦)，它就是下面的序列：

$$z_2=2,\quad z_{n+1}=2^{n-1/2}\sqrt{1-\sqrt{1-4^{1-n}z_n^2}},\quad n=2,\ 3,\ \cdots \tag{1.4}$$

当 n 趋于无穷时，序列将收敛于 π。当利用 MATLAB 来计算 z_n 时，π 和 z_n 之间的相对误差在前 16 次迭代运算中是逐渐减少的，但当迭代运算继续时，由于舍入误差的影响，相对误差会逐渐增加(见图 1.2)。

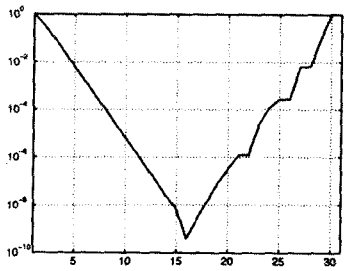

图 1.2 相对误差$|\pi-z_n|/\pi$ 与 n 的对数曲线

参见习题 1.1~1.2。

1.2 复数

复数集用 **C** 来表示，复数的形式为 $z=x+iy$，其中 $i=\sqrt{-1}$ 是虚数单位(此时有 $i^2=-1$)，而 $x=\mathrm{Re}(z)$和 $y=\mathrm{Im}(z)$分别是 z 的实部和虚部。复数在计算机上通常采用实数对的形式来表示。

在 MATLAB 中，变量 i 和 j 如果没有经过重新定义，则它们代表虚数单位。要引入一个实部为 x，虚部为 y 的复数，只需写成 x+i*y；也可以采用另一种方式，使用命令 complex(x, y)来实现。再来看一下复数 z 的三角表示法(或称极坐标表示法)：

$$z = \rho e^{i\theta} = \rho(\cos\theta + i\sin\theta), \tag{1.5}$$

其中 $\rho=\sqrt{x^2+y^2}$ 是复数的绝对值(可通过 abs(z)命令来得到)，θ 为相角，它是在复平面(x，y)上向量 z 与 x 轴之间的夹角。θ 可以通过键入 angle(z)命令来得到。于是式(1.5)可以表示为:

abs(z)*(cos(angle(z))+i *sin(angle(z)))

通过 compass(z)命令可以得到一个或多个复数的极坐标形式，其中 z 既可以是单一的复数，也可以是一个复向量，该向量的元素为复数。例如键入下列语句:

```
>>z=3+i*3;compass(z);
```

可以得到如图 1.3 所示的图表。

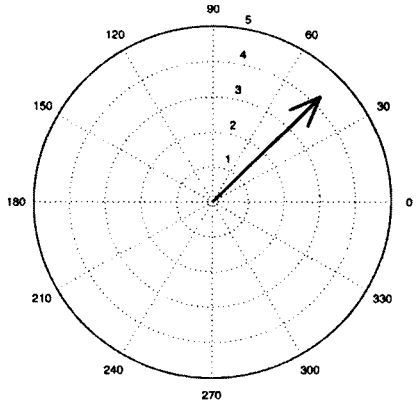

图 1.3 MATLAB 的 compass 命令的输出

对于任意给定的复数 z，可以利用 real(z)命令来求得实部，利用 imag(z)命令求得虚部，并可以利用 conj(z)函数来方便地得到 z 的复共轭 $\bar{z}=x-iy$。

在 MATLAB 进行所有的运算之前都暗含着一个假设：所有操作数和结果一律按复数对待，因此有时会有一些令人意外的现象出现。例如，如果利用 MATLAB 命令(-5)^(1/3)来求-5 的立方根，此时得到的结果并不是-1.7099…，而是复数 0.8500+1.4809i(设符号^代表幂指数运算)。实际上，所有以 $\rho e^{i(\theta+2k\pi)}$(其中 k 为整数)这一形式表示的数与 z 是难以区分的。计算 $\sqrt[3]{z}$ 可以得到 $\sqrt[3]{\rho}e^{i(\theta/3+2k\pi/3)}$，也就是下列三种不同形式:

$$z_1 = \sqrt[3]{\rho}e^{i\theta/3},\quad z_2 = \sqrt[3]{\rho}e^{i(\theta/3+2\pi/3)},\quad z_3 = \sqrt[3]{\rho}e^{i(\theta/3+4\pi/3)}$$

ATLAB 将从其中选择一个作为输出，选择的方法是在复平面上从实轴开始逆

时针旋转扫描，最先遇到的那个值便作为结果输出。由于 $z=-5$ 的极坐标表示形式是 $\rho e^{i\theta}$，其中 $\rho=5$，$\theta=-\pi$，则三个根分别为(图 1.4 表示出了三个根在 Gauss 平面上的分布)：

$$z_1=\sqrt[3]{5}\,(\cos(-\pi/3)+i\sin(-\pi/3))\simeq 0.8550-1.4809i$$
$$z_2=\sqrt[3]{5}\,(\cos(\pi/3)+i\sin(\pi/3))\simeq 0.8550+1.4809i$$
$$z_3=\sqrt[3]{5}\,(\cos(-\pi)+i\sin(-\pi))\simeq -1.71$$

可见 MATLAB 选择了第二个根作为输出(参见图 1.4)。

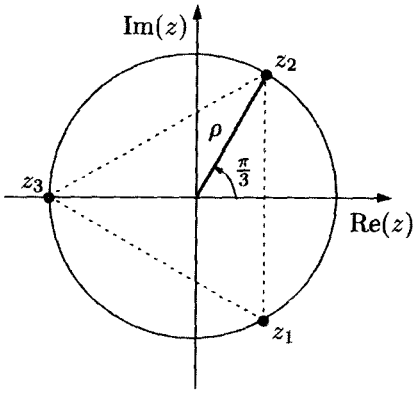

图 1.4　实数 -5 的立方根在复平面上的表示

最后，由式(1.5)可得 Euler 公式：

$$\cos(\theta)=\frac{1}{2}\left(e^{i\theta}+e^{-i\theta}\right),\quad \sin(\theta)=\frac{1}{2i}\left(e^{i\theta}-e^{-i\theta}\right) \tag{1.6}$$

1.3　矩阵

设 n 和 m 是正整数，一个 m 行 n 列的矩阵由 $m\times n$ 个元素 a_{ij} 组成，其中 $i=1,\cdots,m$，$j=1,\cdots,n$，用下面的阵列来表示：

$$\boldsymbol{A}=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \tag{1.7}$$

矩阵 $\boldsymbol{A}$ 可简写为 $\boldsymbol{A}=(a_{ij})$。若 $\boldsymbol{A}$ 中元素为实数，可记为 $\boldsymbol{A}\in\mathbf{R}^{m\times n}$，如果为复数，可记为 $\boldsymbol{A}\in\mathbf{C}^{m\times n}$。

n 维方阵即是 $m=n$ 时的矩阵。从矩阵中单独提出一列元素，则称之为列向量，而从矩阵中单独提出一行元素称为行向量。

为了在 MATLAB 中引入矩阵，需要逐行地写入所有元素，给出表示矩阵的符号，同时行与行之间要互相间隔开。例如键入命令

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

得到

```
A=
    1    2    3
    4    5    6
```

它是一个 2×3 的矩阵，矩阵的元素就是输入的数值。零矩阵 0 即是矩阵中所有元素 $a_{ij}=0$(其中 $i=1, \cdots, m, j=1, \cdots, n$)的矩阵；一个 $m\times n$ 阶的矩阵可以用 MATLAB 命令 zeros(m, n)得到。键入命令 eye(m, n)可构造一个对角阵，这个对角阵主对角线的元素都为 1，其余元素都为 0。

一个 $m\times n$ 矩阵 $\boldsymbol{A}$ 的主对角线是由元素 $a_{ii}(i=1, \cdots, \min(m, n))$所构成的对角线。特别地，执行命令 eye($n$)(即 eye($n$, n)的简化形式)可以得到一个主对角线均为 1 的 n 维方阵，称为单位阵，记为 $\boldsymbol{I}$。

定义下列运算：

1. 如果 $\boldsymbol{A}=(a_{ij})$，$\boldsymbol{B}=(b_{ij})$是 $m\times n$ 矩阵，$\boldsymbol{A}$ 与 $\boldsymbol{B}$ 加法运算定义为：$\boldsymbol{A}+\boldsymbol{B}=(a_{ij}+b_{ij})$；

2. 矩阵 $\boldsymbol{A}$ 与一个实数或复数 λ 之间的乘法运算定义为 $\lambda\boldsymbol{A}=(\lambda a_{ij})$；

3. 两个矩阵进行乘法运算，要求这两个矩阵有相邻的公共维，即如果矩阵 $\boldsymbol{A}$ 是 $m\times p$ 的，矩阵 $\boldsymbol{B}$ 是 $p\times n$ 的，p 为正整数，则可以进行乘法运算。相乘后的矩阵 $\boldsymbol{C}=\boldsymbol{AB}$ 是一个 $m\times n$ 的矩阵，其元素

$$c_{ij}=\sum_{k=1}^{p}a_{ik}b_{kj}, \quad \text{其中} i=1, \cdots, m, \quad j=1, \cdots, n$$

下面给出一个矩阵加法和乘法运算的例子：

```
>>A=[1 2 3; 4 5 6]; B=[7 8 9; 10 11 12]; C=[13 14; 15 16; 17 18];
>>A+B
ans=
8    10   12
14   16   18
  >>A*C
ans=
94    100
229   224
```

需要注意，如果对两个维数不匹配的矩阵进行操作，MATLAB 将给出诊断信息。