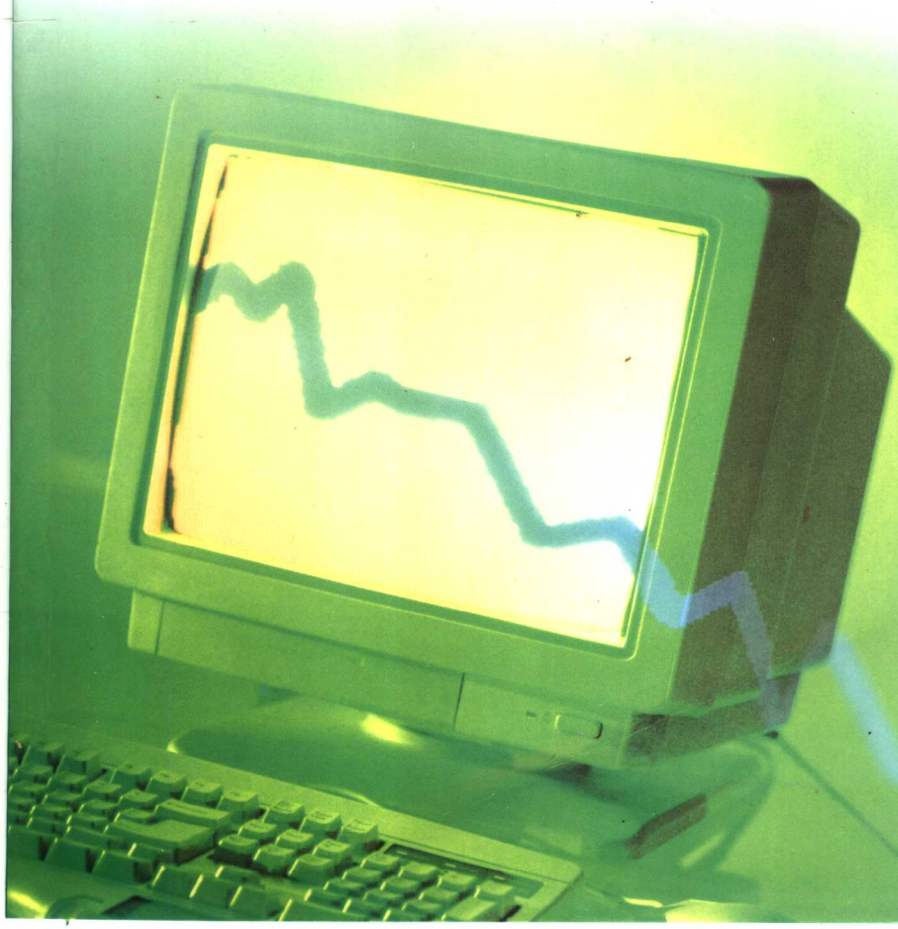
SHUZHI FENXI YU SHIYAN

数值分析与实验

薛毅 编著



北京工业大学出版社

前　言

数值分析是一门理论性很强、应用面很广的课程．特别是随着计算机的普及与发展，数值分析已在各方面得到广泛的应用．我国绝大多数的理工科院校都开设不同层次的数值分析课或计算方法课．但目前的数值分析课的教学也存在一些缺陷，主要问题是将数值分析仅作为一门数学理论课，而对它的应用讲述不够，数值实验不足．这除了硬件设备外，其主要原因是在数值分析的计算中需要用到计算机程序语言，如 C 语言、Fortran 语言等，在编程实现中会遇到的一些计算机知识方面的困难，而这些困难又是数学教师不便于解决的．

Matlab 软件已经相当普及了，数不胜数的各种 Matlab 书籍就能说明这一点．它有着非常强大的数值计算功能，除了基本运算外，还有信息工程、应用数学、控制工程等方面的大约 25 个工具箱（Toolbox）．所谓工具箱，就是求解问题的各种函数，有了这些函数就可以最大限度地避免重复劳动，可以“站在巨人的肩膀上”开展工作．而目前这方面的书籍也存在一个问题，它们告诉读者如何使用功能强大的各种函数，但对求解问题的基本原理几乎不作介绍，使应用者只知其然，不知其所以然．当在解决实际问题中遇到困难时，就不知如何处理了．

本教材的目标是将两者有机地结合，使学习者既知其然，也知其所以然，既学会使用 Matlab 中提供的函数解决问题，又了解这些函数编程的理论基础．当遇到困难时，能够有办法找出解决问题的方案．

要写一本包含 Matlab 所有函数且能达到上述目的的教科书几乎是不可能的，本书也没有打算这样做．本书只是就数值分析中出现的各种方法和与之有关的 Matlab 函数作相应的介绍．

本书首先介绍通常数值分析中各种数值计算的原理与方法，再给出相应的算法，然后根据这些算法编写相应的 Matlab 程序，这是本书的第一个层次．

通过这一层次的学习，使学生对数值分析的整体有一个充分的了解，可以通过几何直观对算法本质有充分的认识．通过 Matlab 的编程训练（Matlab 只作为计算机语言的训练平台），使学生对数值分析的各种算法有更深入的理解．只有很好地理解算法，才能很好地编程，而很好的编程计算，才会使学生对算法有更深刻的理解．因此，这一层次的目的并不在于编程训练，而是通过 Matlab 的编程训练，使学生能在脱离 Matlab 工作环境时，完成相应的计算工作．如果直接用计算机语言（如 C 语言）完成这一目标，会遇到许多困难，需要让学生预先了解相关的计算机语言，并完成大量的底层工作，这些工作将会干扰学生对算法的了解与训练．

由于有了 Matlab 软件作为辅助工具，可以使学生在原有数值分析教科书的基础上，开展进一步的工作，介绍一些更深入的问题，如将求解非线性方程的数值方法推广到求解非线性方程组是十分方便的．这样做的好处是不必增加过多的计算，便可以清楚、方便、深入浅出地了解算法的道理．这是本书的第二个层次．

这一层次的目的就是开阔学生眼界，如解数值分析中较为深入的课题，像数值积分中的

Lobatto 求积公式，二重、三重数值积分等．通过这一层次内容的学习，为数值分析方法的应用奠定了基础．

本书还重点介绍 Matlab 中与数值分析算法有关的各种函数，利用这些函数可以非常方便、快捷地求解各种复杂的问题．为了便于学生对数值方法应用有一个充分的了解，本书在每章中特别安排了一节案例分析，通过案例分析帮助学生从“算”数学过渡到“用”数学．这是本书的第三个层次．

第三个层次的目的是将数值分析与实际应用相结合，让学生学习数学建模的基本过程，应用数值分析的基本知识，建立相应的数学模型，解决实际问题．这部分的重点是模型的建立、计算结果的分析以及对计算结果的解释等工作，并不强调问题的求解过程，而关于问题的求解可以利用 Matlab 自带的优秀函数和各种工具箱来完成．完成这一层次训练同时还达到本书的另一目的，就是训练学生用 Matlab 提供的各种工具箱求解问题．这部分的训练实际是在完成数学建模的整个过程．

本书各章的排列顺序与作者所编的《数值分析》[1]教材基本相同．第一章为 Matlab 软件简介，主要介绍 Matlab 的基本命令与函数，以及简单的绘图语句．第二章为解非线性方程的数值方法，增加了求非线性方程组的数值方法．第三章为线性方程组的数值解法，增加了对病态方程组求解的讨论．第四章为解线性代数方程组的迭代法，增加了用共轭梯度法求解方程组的内容．第五章为插值方法，增加了高维插值函数的介绍．第六章为函数逼近与数据拟合，增加了非线性最小二乘方法的介绍．第七章为数值积分，增加了 Lobatto 求积法和反常积分的方法．第八章为常微分方程的数值解，增加了微分方程组刚性问题的讨论．第九章为矩阵特征值与特征向量计算，增加了广义特征值与特征向量的计算．

本书各章的结构如下：第一部分是基本方法介绍．简单论述了数值计算所涉及算法的基本原理与方法、相应的几何直观解释，给出了算法的基本性质和基本结论，略去了相关的证明，如读者需要了解这方面更深入的知识，可以参考这一方面的教学参考书，如［1］，［2］，［3］和［4］．这一部分给出了本节的算法和相应的 Matlab 函数（程序），这些函数是作者自编的，其主要目的是给学生编写程序起抛砖引玉的作用．这些函数从功能和适用性等方面来讲，都无法与 Matlab 提供的函数相比，但这些函数对于 Matlab 提供的函数，以及本章所讨论算法的理解还是有很大帮助的．

第二部分是深入讨论的课题．主要讨论一些较为深入的问题，是对前一部分内容的扩充．如在讲完非线性方程的求解后，增加了非线性方程组的求解方法．在讲完求解线性方程组的迭代法后，增加了用共轭梯度法求解的方法．每章增加的内容，基本上是对原问题的扩充．

第三部分是 Matlab 函数．主要介绍 Matlab 提供的、求解本节问题的各种函数．这部分的内容重点是这些函数的使用和参数的调用，为后面求解实际问题作准备，也为今后解决实际问题提供帮助，有关 Matlab 软件更深入的知识，读者可参见相关的 Matlab 书籍，如［5］和［6］．

第四部分是案例分析．通过案例分析，使学生了解本章内容的应用点，如何运用本章知识去解决实际问题，当然，这部分内容基本上都属于综合应用的题目，有些内容可能超出本章的知识范畴．

第五部分是实验．所谓实验实际上就是留给学生的习题，只不过这部分习题不是用“手

算”完成的，而是需要借助于计算机、自编函数以及 Matlab 提供的相关函数来完成的，并在完成实验后，要求学生写出相应的实验报告．这部分内容对学生掌握该章的内容是非常重要的，也可作为考查学生掌握知识的依据．

本书所编写的 Matlab 函数以及所介绍的 Matlab 函数均在 Matlab 6.5 环境下完成，而且全部程序均运行通过，在其他的 Matlab 环境下可能会出现一些微小的差别，但这些差别不会影响到函数的使用和对算法的理解．读者如果需要作者自编的 Matlab 程序，可以发电子邮件向作者索取，邮件地址：xueyi@bjut.edu.cn．

本书虽然涉及数值分析、计算机知识和 Matlab 软件等多方内容，但本书的立足点是数值分析的实验与应用，因此在编写上自成体系，只要具备初步的计算机知识就能读懂本书的全部内容．

本书是为工科各专业研究生和本科生、应用数学本科生“数值分析实验课”或“数值分析实习”编写的，可用作“数学建模”或“数学实验课”的教材，作为大学生参加全国大学生数学建模竞赛的辅导材料，也可作为“数值分析”或“计算方法”课程的辅助教材或教学参考书．特别是对那些非常关心实际问题的求解与计算的工科研究生、科技工作者和工程技术人员，本书将会为他们提供很大的帮助．

由于受编者水平限制，可能在内容的取材、结构的编排以及课程的讲法上存在不妥之处，我们希望使用本书的老师、同学以及同行专家和其他读者提出宝贵的批评和建议．

在本书出版之际，我们谨向对本书提供过帮助的各位老师和专家，以及给予我们大力支持的北工大出版基金委员会和北工大出版社表示衷心的感谢．

编　者

2004 年 10 月

目　录

第一章　Matlab 软件简介 …… 1
1.1　矩阵与数组运算 …… 1
1.1.1　向量、矩阵的表示 …… 1
1.1.2　矩阵运算 …… 2
1.1.3　数组运算 …… 4
1.1.4　关系运算 …… 6
1.1.5　逻辑运算 …… 7
1.1.6　矩阵运算函数 …… 7
1.1.7　基本函数 …… 9
1.2　控制流语句 …… 10
1.2.1　for 循环语句 …… 10
1.2.2　while 循环语句 …… 11
1.2.3　if 和 break 语句 …… 11
1.3　文件 …… 12
1.3.1　M 文件 …… 12
1.3.2　文件的输入和输出 …… 14
1.4　绘图 …… 15
1.4.1　二维绘图 …… 15
1.4.2　三维绘图 …… 19
1.4.3　图形的保存 …… 21
1.5　实验 …… 22
1.5.1　实验目的 …… 22
1.5.2　实验要求 …… 23
1.5.3　实验步骤 …… 23
第二章　非线性方程求解 …… 25
2.1　非线性方程求解方法 …… 25
2.1.1　二分法 …… 25
2.1.2　迭代法 …… 28
2.1.3　Newton 法 …… 32
2.2　求非线性方程组的解 …… 35
2.3　求解非线性方程根的 Matlab 函数 …… 37
2.3.1　solve 函数 …… 37
2.3.2　fzero 函数 …… 39
2.3.3　fsolve 函数 …… 40
2.4　案例分析——商品的产量与价格问题 …… 42
2.4.1　问题的提出 …… 42
2.4.2　问题的求解 …… 42
2.5　实验 …… 44
2.5.1　实验目的 …… 44
2.5.2　实验要求 …… 44
2.5.3　实验步骤 …… 44
第三章　线性方程组的数值解法 …… 46
3.1　线性方程组的求解方法 …… 47
3.1.1　列主元 Gauss 消去法 …… 47
3.1.2　Gauss-Jordan 消去法 …… 52
3.1.3　矩阵分解方法 …… 56
3.1.4　对称正定矩阵的 Cholesky 分解 …… 60
3.2　求解线性方程组的摄动理论 …… 64
3.2.1　范数 …… 64
3.2.2　矩阵的条件数 …… 67
3.3　病态方程组的求解 …… 69
3.3.1　Hilbert 矩阵 …… 69
3.3.2　求解病态线性方程组出现的问题 …… 70
3.3.3　奇异值分解 …… 71
3.4　案例分析——投入产出模型 …… 72
3.4.1　问题的提出 …… 72
3.4.2　问题的求解 …… 73
3.5　实验 …… 78
3.5.1　实验目的 …… 78
3.5.2　实验要求 …… 78
3.5.3　实验步骤 …… 78
第四章　解线性代数方程组的迭代法 …… 80
4.1　求解线性代数方程组的迭代方法 …… 80
4.1.1　Jacobi 迭代法 …… 80
4.1.2　Gauss-Seidel 迭代法 …… 82
4.1.3　逐次超松弛迭代法 …… 84
4.2　迭代法的收敛性 …… 86
4.2.1　迭代收敛定理 …… 86
4.2.2　迭代收敛速度 …… 88
4.2.3　Jacobi 迭代与 Gauss-Seidel 迭代

的进一步讨论 …… 89
4.2.4 逐次超松弛迭代中最优松弛因子的讨论 …… 90
4.3 求解线性方程组的共轭梯度法 …… 93
4.4 实验 …… 95
4.4.1 实验目的 …… 95
4.4.2 实验要求 …… 95
4.4.3 实验步骤 …… 95
第五章 插值方法与数值微分 …… 97
5.1 插值的一般方法 …… 97
5.1.1 Lagrange 插值多项式 …… 97
5.1.2 Newton 插值 …… 99
5.1.3 Hermite 插值 …… 104
5.1.4 分段低次插值 …… 106
5.1.5 三次样条插值函数 …… 110
5.2 有关插值运算的 Matlab 函数 … 116
5.2.1 interpl 函数 …… 116
5.2.2 spline 函数 …… 118
5.2.3 pchip 函数 …… 119
5.2.4 csape 函数 …… 119
5.3 高维插值函数 …… 121
5.3.1 interp2 函数 …… 121
5.3.2 griddata 函数 …… 122
5.3.3 interp3 函数 …… 125
5.3.4 interpn 函数 …… 125
5.4 数值微分 …… 125
5.4.1 数值微分的计算公式 …… 125
5.4.2 Matlab 中关于数值微分的函数 …… 128
5.5 案例分析——估计水塔的水流量 …… 131
5.5.1 问题的提出 …… 131
5.5.2 问题的求解 …… 131
5.5.3 模型的稳定性分析与检验 …… 133
5.6 实验 …… 134
5.6.1 实验目的 …… 134
5.6.2 实验要求 …… 134
5.6.3 实验步骤 …… 134
第六章 函数逼近与数据拟合 …… 136
6.1 函数的最佳平方逼近 …… 136
6.1.1 最佳平方逼近的概念及计算 … 136
6.1.2 最佳平方逼近算法与 Matlab 计算 …… 137
6.2 数据的最小二乘拟合 …… 139
6.2.1 基本概念 …… 139
6.2.2 最小二乘算法与 Matlab 计算 … 140
6.3 数据拟合的 Matlab 实现 …… 142
6.3.1 polyfit …… 143
6.3.2 lsqcurvefit …… 144
6.3.3 lsqnonlin …… 144
6.3.4 nlinfit …… 145
6.4 案例分析——人口增长模型 … 146
6.4.1 Malthus 人口模型 …… 146
6.4.2 Logistic 人口模型 …… 147
6.4.3 案例分析——确定人口增长模型中的参数 …… 147
6.5 实验 …… 149
6.5.1 实验目的 …… 149
6.5.2 实验要求 …… 149
6.5.3 实验步骤 …… 150
第七章 数值积分 …… 151
7.1 Newton – Cotes 求积公式 …… 151
7.1.1 梯形求积公式 …… 151
7.1.2 Simpson 求积公式 …… 153
7.1.3 Cotes 求积公式 …… 154
7.2 自适应步长的求积方法 …… 156
7.2.1 自适应步长梯形公式 …… 156
7.2.2 自适应步长 Simpson 公式 …… 158
7.2.3 Romberg 求积公式 …… 159
7.3 高精度求积公式 …… 161
7.3.1 Gauss 求积公式 …… 161
7.3.2 Lobatto 求积公式 …… 165
7.4 数值积分的 Matlab 实现 …… 166
7.4.1 trapz 函数 …… 166
7.4.2 quad 函数 …… 167
7.4.3 quadl 函数 …… 167
7.4.4 dblquad 函数 …… 167
7.4.5 triplequad 函数 …… 168
7.5 反常积分的数值方法 …… 169
7.5.1 暇积分 …… 169
7.5.2 无穷区间上的积分 …… 169
7.6 实验 …… 172

7.6.1 实验目的 …… 172
7.6.2 实验要求 …… 172
7.6.3 实验步骤 …… 172
第八章 常微分方程的数值解 …… 174
8.1 单步方法 …… 174
8.1.1 Euler 方法 …… 174
8.1.2 Runge-Kutta 方法 …… 178
8.1.3 单步法的收敛性 …… 180
8.1.4 单步法的稳定性 …… 182
8.2 线性多步法 …… 184
8.2.1 线性多步法的一般公式 …… 184
8.2.2 Adams 外推公式 …… 184
8.2.3 Adams 内插公式 …… 185
8.2.4 预报—校正公式 …… 185
8.3 一阶微分方程组和高阶微分方程的数值计算 …… 187
8.3.1 一阶微分方程组 …… 187
8.3.2 高阶微分方程 …… 188
8.3.3 刚性方程组 …… 189
8.4 微分方程（组）初值问题计算的 Matlab 实现 …… 190
8.4.1 非刚性问题的计算 …… 190
8.4.2 刚性（stiff）方程组的计算 …… 193
8.4.3 odeset 函数 …… 195
8.5 案例分析——Lanchester 作战模型 …… 197
8.5.1 问题的提出 …… 197
8.5.2 模型建立 …… 197
8.5.3 硫黄岛战役 …… 197
8.6 实验 …… 200
8.6.1 实验目的 …… 200
8.6.2 实验要求 …… 200
8.6.3 实验步骤 …… 200
第九章 矩阵特征值与特征向量的计算 …… 203
9.1 求解特征值与特征向量的数值方法 …… 203
9.1.1 幂法 …… 203
9.1.2 反幂法 …… 205
9.1.3 Jacobi 方法 …… 207
9.2 求矩阵特征值的 Matlab 实现 … 211
9.2.1 求矩阵的特征值与特征向量 … 211
9.2.2 求矩阵的广义特征值与特征向量 …… 212
9.3 矩阵特征值的应用——循环赛排名次问题 …… 213
9.3.1 问题的提出 …… 213
9.3.2 问题的分析与求解 …… 213
9.4 实验 …… 214
9.4.1 实验目的 …… 214
9.4.2 实验要求 …… 215
9.4.3 实验步骤 …… 215
参考文献 …… 216

第一章　Matlab 软件简介

Matlab 的含义是矩阵实验室（matrix laboratory），是美国 MathWork 公司于 1984 年推出的一套高性能的数值计算和可视化软件，它集数值分析、矩阵计算、信号处理和图形显示于一体，构成一个方便的、界面友好的用户环境．到目前为止，它已发展成为国际上最优秀的科技应用软件之一．

1.1　矩阵与数组运算

Matlab 软件最基本的语句是矩阵与数组运算，它可以非常方便地完成向量、矩阵的各种运算，如向量的加法与数乘，矩阵的加法、数乘、乘法、除法（求逆）和乘方等运算以及数组的点乘、点除等运算．

1.1.1　向量、矩阵的表示

1. 行向量

```
>>x = [-1 0 2]      %中间用空格将数据分开，也可以用逗号分开
x =                 %显示输入结果
    -1     0     2
```

在 Matlab 软件中，%后的语句是说明语句．

2. 列向量

```
>> y =[3 8 2]'              %'表示转置
y =
    3
    8
    2
```

3. 矩阵

```
>> A = [1 2 3; 4 5 6; 7 8 0]      %; 表示换行
A =
    1    2    3
    4    5    6
    7    8    0
```

4. 转置

```
>> B = A'
B =
    1    4    7
```

```
     2     5     8
     3     6     0
```

1.1.2 矩阵运算

在 Matlab 中，通常意义上的运算（与线性代数运算的意义相同）称为矩阵运算．矩阵运算包括

+ 加　-减　*乘　/右除　\左除　^乘幂

在数值运算中左除与右除是一样的，如 1/4 与 4 \ 1，其结果均为 0.25，但对于矩阵运算，结果则大不相同．

1. 加减法运算

加减法运算的基本要求是两矩阵（或向量）的维数必需相同．如

```
>> C = A + B
C =
     2     6    10
     6    10    14
    10    14     0
```

但矩阵 ***A*** 可以与一个数字相加，如

```
>> A + 2
ans =
     3     4     5
     6     7     8
     9    10     2
```

相当于每个元素 +2，对于向量也有同样的结果．如

```
>> x + 2
ans =
     1     2     4
```

2. 乘法运算

矩阵乘法（*）运算，有数乘运算，即纯量与矩阵相乘，如

```
>> 2 * A
ans =
     2     4     6
     8    10    12
    14    16     0
```

矩阵与矩阵相乘，如

```
>> D = A * B
D =
    14    32    23
    32    77    68
    23    68   113
```

当然两矩阵相乘需要满足线性代数中矩阵乘法的运算规则.

3. 矩阵求逆

inv () 函数是矩阵求逆运算函数，如

```
>> inv (A)
ans =
    -1.7778    0.8889   -0.1111
     1.5556   -0.7778    0.2222
    -0.1111    0.2222   -0.1111
```

表示矩阵 $\boldsymbol{A}$ 的逆矩阵 $\boldsymbol{A}^{-1}$. 因此

```
>> I = inv (A) * A
I =
    1.0000    0.0000         0
    0.0000    1.0000         0
    0.0000         0    1.0000
```

是单位矩阵.

4. 矩阵除法运算

```
>> b = [1 1 1]';                % 输入向量 b
>> x = A \ b                    % 左除运算
x =
   -1.0000
    1.0000
    0.0000
```

表示方程组 $\boldsymbol{Ax} = \boldsymbol{b}$ 的解，即 $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$. 因此，

```
>> x = inv (A) * b
```

将得到同样的结果.

如果矩阵不是方的，inv () 运算无意义，但矩阵除法仍有它的数学背景. 对于超定方程组，矩阵除法表示最小二乘解，如

```
>> X = [1 1 1 1 1 1 1 1 1 1; 1 0 2 0 3 1 0 1 2 0]';
>> y = [16 9 17 12 22 13 8 15 19 11]';
>> beta = X \ y
beta =
   10.2000
    4.0000
```

即 $\boldsymbol{\beta} = (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}$. 等价于 $\boldsymbol{\beta} = \boldsymbol{X}^{+}\boldsymbol{y}$，其中 $\boldsymbol{X}^{+}$ 表示 $\boldsymbol{X}$ 的 + 号广义逆.

5. 乘幂运算

矩阵的乘幂 (^) 运算与通常矩阵乘幂的概念相同，如

```
>> D = A^2                      % A^2 表示 A * A
D =
```

30 36 15
66 81 42
39 54 69

即 $\boldsymbol{D} = \boldsymbol{A}^2$.

1.1.3 数组运算

通常将向量或矩阵的点运算称为数组运算（加减运算无点运算），其点运算有

.* 点乘 ./ 点右除 .\ 点左除 .^ 点乘幂

1. 数组的输入

有两种方法构造一维数组，第一种方法，用“:”构造一维等差数组，其格式为

数组 = 初值 : 增量 : 终值

如

```
>> a = 0 : 3 : 10
a =
    0     3     6     9
```

当增量值为 1 时，增量值可省缺，如

```
>> b = 1 : 5
b =
    1     2     3     4     5
```

第二种方法，用 linspace () 函数构造一维等差数组，其格式为

数组 = linspace (初值，终值，等分点数)

如

```
>> c = linspace (0, 10, 4)
c =
    0    3.3333    6.6667    10.0000
```

等分区间数 = 等分点数 - 1.

2. 数组加减法运算

数组的加减法运算与矩阵的加减法运算相同，就是通常意义下的加减法运算，如

```
>> x = [1 2 3];                    %增加；后，计算机就不显示输入结果
>> y = [4 5 6];
>> x + y                           % 两数组的维数必须相同
ans =
    5     7     9
```

3. 数组乘法运算

数组的乘法运算，也叫点乘（.*）运算，它表示两数组在同一位置上的元素相乘，如

```
>> z = x.* y                       % 两数组的维数必须相同
z =
    4    10    18
```

即 $z_i = x_i \cdot y_i$，其中 $z = (z_1, z_2, z_3)$，$x = (x_1, x_2, x_3)$，$y = (y_1, y_2, y_3)$.

对于一维数组 x，y，矩阵的乘法运算（x * y）无意义.

对于矩阵 A，B，尽管矩阵的乘法运算（A * B）可能有意义，但也可作点乘（.*）运算（为了避免与矩阵乘法混淆，仍称为数组的乘法），如

```
>> C = A .* B                % 两矩阵必须有相同的维数
C =
     1     8    21
     8    25    48
    21    48     0
```

它表示两矩阵同一位置上的元素相乘，即 $c_{ij} = a_{ij} \cdot b_{ij}$，其中 $C = (c_{ij})$，$A = (a_{ij})$，$B = (b_{ij})$.

注意，在数组的乘法运算中，点乘（.*）必须看成一个整体的运算符号，不能将“.”与“*”分开.

4. 数组除法运算

数组的除法运算，也叫点除（.\ 或 ./）运算，两者都表示同一位置上的元素相除，但意义上有差别. 如

```
>> z = x .\ y
z =
    4.0000    2.5000    2.0000
```

点左除表示 $z_i = y_i / x_i$，其中 $z = (z_1, z_2, z_3)$，$x = (x_1, x_2, x_3)$，$y = (y_1, y_2, y_3)$.

```
>> z = x ./ y
z =
    0.2500    0.4000    0.5000
```

点右除表示 $z_i = x_i / y_i$，其中 $z = (z_1, z_2, z_3)$，$x = (x_1, x_2, x_3)$，$y = (y_1, y_2, y_3)$. 从上述运算可以看到，数组的左除运算与右除运算意义是不同的.

与点乘运算相同，点除（./，.\），以及后面将讲到的点乘幂（.^）也必须看成一个整体的运算符号.

对于矩阵 A，B，也有点除（./，.\）运算，为避免与矩阵除法混淆，仍称为数组的除法.

注意，尽管 x，y 都是一维数组，但它们之间仍能作矩阵的除法（/或\）运算. 如

```
>> z = x / y
z =
    0.4156
```

其运算相当于 $z = (x * y') / (y * y')$，即 z 是方程组 $x = zy$ 的最小二乘解. 而

```
>> z = x \ y
z =
         0         0         0
         0         0         0
    1.3333    1.6667    2.0000
```

即 z 是方程组 $xz = y$ 的最小范数解.

5. 数组乘幂运算

数组的乘幂运算，也叫点乘幂（.^）运算，即对每个数组中的元素作乘幂运算，如

```
>> z = x .^ 2
z =
     1     4     9
```

表示数组 $\boldsymbol{x}$ 的每个元素的平方，即 $z_i = x_i^2$，其中 $z = (z_1, z_2, z_3)$，$\boldsymbol{x} = (x_1, x_2, x_3)$.

```
>> z = 2 .^ y
z =
    16    32    64
```

表示 $z_i = 2^{y_i}$，其中 $z = (z_1, z_2, z_3)$，$\boldsymbol{y} = (y_1, y_2, y_3)$.

```
>> z = x .^ y
z =
     1    32   729
```

表示 $z_i = x_i^{y_i}$，其中，$z = (z_1, z_2, z_3)$，$\boldsymbol{x} = (x_1, x_2, x_3)$，$\boldsymbol{y} = (y_1, y_2, y_3)$.

由于 $\boldsymbol{x}$,$\boldsymbol{y}$ 是一维数组，因此相应的矩阵乘幂运算（x^2, 2^y, x^y）均无意义.

对于矩阵 $\boldsymbol{A}$,$\boldsymbol{B}$，也有点乘幂（.^）运算，为避免与矩阵乘幂运算混淆，仍称为数组的乘幂运算. 如

```
>> G = A .^ 2
G =
     1     4     9
    16    25    36
    49    64     0
```

表示 $g_{ij} = a_{ij}^2$，其中 $\boldsymbol{G} = (g_{ij})$，$\boldsymbol{A} = (a_{ij})$.

```
>> G = A .^ B                    % 两矩阵必须有相同的维数
G =
     1    16  2187
    16  3125  1679616
   343  262144     1
```

表示 $g_{ij} = a_{ij}^{b_{ij}}$，其中 $\boldsymbol{G} = (g_{ij})$，$\boldsymbol{A} = (a_{ij})$，$\boldsymbol{B} = (b_{ij})$.

1.1.4 关系运算

对于矩阵、向量和数组有如下关系运算：

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

如果关系成立返回值则为 1，否则返回值为 0. 如

```
>> a = [-1 2 4; 5 4 -8];
>> c = a > 0
c =
     0     1     1
     1     1     0
```

用关系运算，有时对表达某些函数关系会更加方便．如在 Matlab 中输入两个相同长度的数组 x 与 y，其中 x 的分量是从 -1 到 1，每个分量之间的间隔为 0.2，y 的分量满足

$$y_i = \begin{cases} x_i^2 & x_i \geqslant 0 \\ 0 & x_i < 0 \end{cases}$$

通常的输入方法为

```
x = -1:0.2:1;    y = max(0, x).^2;
```

也可以用关系运算输入

```
x = -1:0.2:1;    y = (x.^2).*(x>=0);
```

第二种输入方法在组合条件比较多的情况下，使用起来更加方便．

1.1.5 逻辑运算

在条件语句中有如下逻辑运算

& 与　| 或　~ 非

A & B　条件 A 与条件 B 同时成立，则为真（返回值为 1）．

A | B　条件 A 或条件 B 成立，则为真（返回值为 1）．

~ A　与条件 A 相反的条件成立，则为真（返回值为 1）．

如输入

```
a = [-1 2 4; 5 4 -8];
b = [-2 1 -1; 3 -1 2];
c = (a > 0) & (b > 0)          % a 与 b 同时大于 0 的元素
d = (a > 0) | (b > 0)          % a 与 b 至少一个大于 0 的元素
e = ~ (a > 0)                  % 即 a 小于等于 0 的元素
```

得到

```
c =
     0     1     0
     1     0     0
d =
     0     1     1
     1     1     1
e =
     1     0     0
     0     0     1
```

1.1.6 矩阵运算函数

1. 求行列式的值(det())

det()函数的功能是求矩阵行列式的值．如

```
>> A = [1 2 3; 4 5 6; 7 8 0];      % 构造一个矩阵
>> det(A)                          % 计算行列式
```

```
ans =
    27
```

2. 三角分解(lu())

三角分解也称为 LU 分解，将矩阵 A 分解成一个单位下三角阵和一个上三角阵的乘积，即 $A = LU$. 在 Matlab 中由函数lu()完成此工作，它对矩阵 A 采用选取主元的 LU 分解. 如

```
>> [L, U] = lu(A)                % 列主元 LU 分解.
L =
    0.1429    1.0000         0
    0.5714    0.5000    1.0000
    1.0000         0         0
U =
    7.0000    8.0000         .0
         0    0.8571    3.0000
         0         0    4.5000
```

这里得到的矩阵 L 并不是下三角矩阵，这是因为在计算中采用了列选主元的结果，将得到的矩阵作适当的行交换，就可以得到下三角阵，关于lu()函数的详细说明请见第三章.

3. 正交三角分解(qr())

正交三角分解也称为 QR 分解，将矩阵 A 分解成一个正交阵 Q 和一上三角阵 R 的乘积，即 $A = QR$. qr()函数完成 QR 分解的工作.

```
>> A = [1 2 3; 4 5 6; 7 8 9; 10 11 12];
>> [Q, R] = qr (A)                % QR 分解
Q =
   -0.0776   -0.8331    0.5444    0.0605
   -0.3105   -0.4512   -0.7709    0.3251
   -0.5433   -0.0694   -0.0913   -0.8317
   -0.7762    0.3124    0.3178    0.4461
R =
  -12.8841  -14.5916  -16.2992
         0   -1.0413   -2.0826
         0         0    0.0000
         0         0         0
```

4. 奇异值分解(svd())

所谓将矩阵的奇异值分解就是将矩阵 A 分解成正交阵 U、对角阵 S 和正交阵 V 的乘积，即 $A = USV^{T}$.

```
>> [U, S, V] = svd(A)             % 奇异值分解.
U =
    0.1409    0.8247    0.5477    0.0078
    0.3439    0.4263   -0.7361    0.3977
    0.5470    0.0278   -0.1709   -0.8190
    0.7501   -0.3706    0.3593    0.4134
```

```
S =
   25.4624         0         0
         0    1.2907         0
         0         0    0.0000
         0         0         0
V =
    0.5045   -0.7608    0.4082
    0.5745   -0.0571   -0.8165
    0.6445    0.6465    0.4082
```

5. 特征值分解(eig())

特征值分解也称为谱分析，即求矩阵 A 的特征值和相应的特征向量.

```
>> A = [1 2 3; 4 5 6; 7 8 9];
>> [Q, D] = eig(A)                    % 特征值分解
Q =
    0.2320    0.7858    0.4082
    0.5253    0.0868   -0.8165
    0.8187   -0.6123    0.4082
D =
   16.1168         0         0
         0   -1.1168         0
         0         0    0.0000
```

其中，D 的对角元素 d_{ii} 是矩阵 A 的第 i 个特征值，Q 的第 i 列是对应第 i 个特征值的特征向量.

1.1.7 基本函数

1. 基本初等函数

sin 正弦　cos 余弦　tan 正切　abs 求绝对值

sqrt 开方　exp 指数　log 对数

在 Matlab 的函数运算中，函数的变量可以是数、向量（数组）和矩阵. 如果自变量是数、向量（数组）和矩阵时，对应的因变量也是数、向量（数组）和矩阵.

2. 与矩阵有关的常用函数

(1) norm (). 求矩阵或向量的 2 - 模 (2 - 范数). 如

```
>> A = [1 2 3; 4 5 6; 7 8 9];
>> norm(A)
ans =
    16.8481
```

(2) cond (). 求矩阵的条件数. 如

```
>> cond(A)
ans =
```

3.7740e+016

(3) rank ( ). 求矩阵的秩. 如

```
>> rank(A)
ans =
     2
```

(4) zeors ( ). 生成零矩阵或零向量. 如生成一个1行3列的零矩阵（向量）

```
>> a = zeros(1, 3)
a =
     0     0     0
```

(5) ones ( ). 生成元素为1的矩阵或向量. 如生成一个1行3列元素为1的矩阵（向量）

```
>> b = ones(1, 3)
b =
     1     1     1
```

(6) eye ( ). 生成单位矩阵或单位向量. 如生成3×3阶单位矩阵

```
>> I = eye(3)
I =
     1     0     0
     0     1     0
     0     0     1
```

(7) size ( ). 求矩阵（数组）的大小（维数）. 如

```
>> size(A)
ans =
     3     3
```

表示矩阵 A 是3×3阶的.

(8) length ( ). 求向量（数组）的长度（维数）. 如

```
>> length(b)
ans =
     3
```

表示向量 b 是3维的.

1.2 控制流语句

1.2.1 for循环语句

for循环语句格式为

```
for  循环变量=初值:增量:终值   % 初值开始，终值结束
     语句                      % 循环体中的执行语句
end                            % 循环结束
```