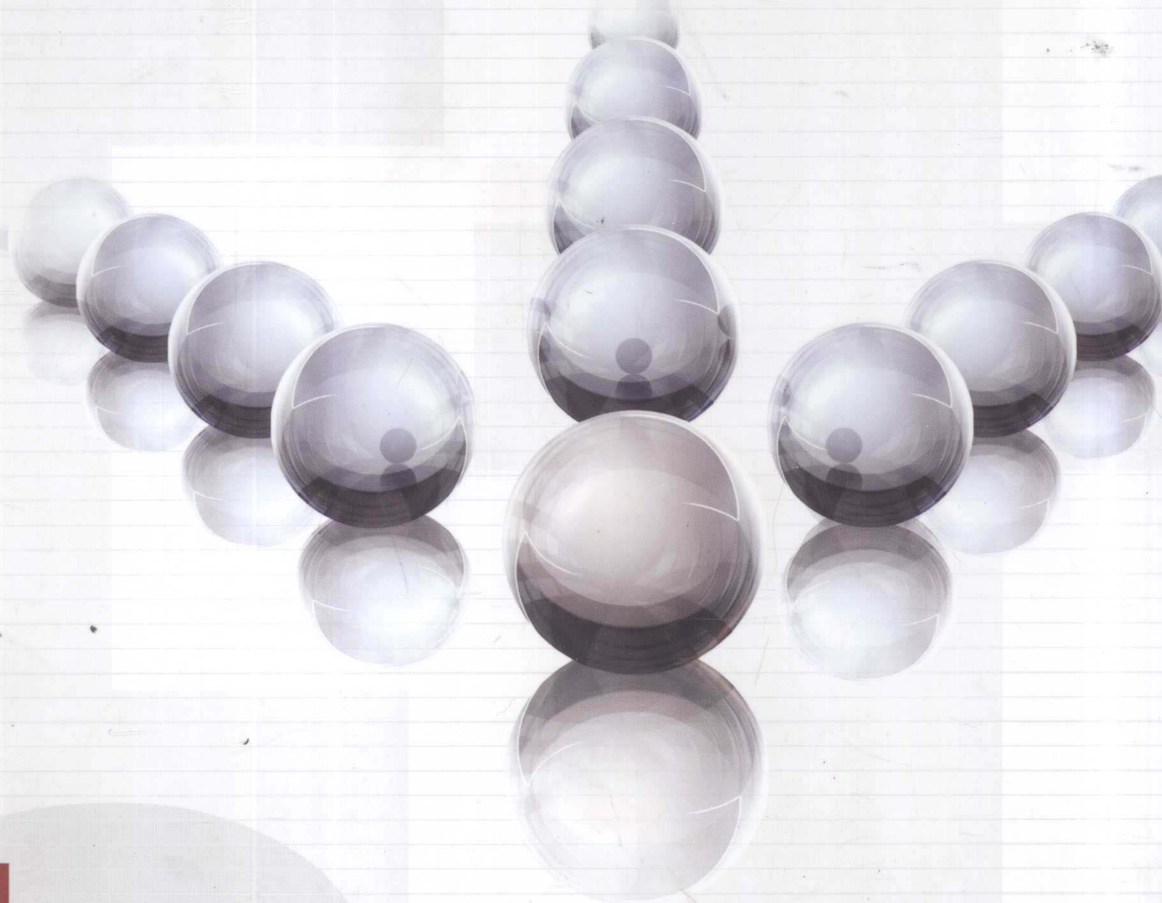

# Fortran 95/2003 科学计算与工程

- 光盘包括12章93个范例代码
- 计算方法和数值分析的参考手册

宋叶志　茅永兴　赵秀杰　编著

清华大学出版社

# Fortran95/2003科学计算与工程

宋叶志 茅永兴 赵秀杰 编著

清华大学出版社
北 京

## 内 容 简 介

科学计算方法是许多科研工作得以展开的前提。本书较为详细地介绍了科学计算与工程中的常用数值方法。全书以 Fortran 95/2003 语言编写而成，全部程序在 Visual Studio 2008 集成 Intel 编译器环境下调试通过。

全书包括 12 章和 3 个附录。主要内容包括矩阵分解与线性方程组的直接方法、线性方程组的迭代方法、最小二乘法与数据拟合、特征值及特征向量、非线性方程求根、非线性方程组数值解法、插值法、数值微分、数值积分、常见的特殊函数计算、常微分方程（组）的数值方法及应用范例。

本书适合作为大学理工科非数学专业本科生或研究生计算方法、数值分析课程的教材或参考书。因为提供了全部的源代码，对于从事数值分析教学的教师也是一本难得的工具书，还可作为科研与工程技术人员的参考手册。



**图书在版编目（CIP）数据**

Fortran 95/2003 科学计算与工程/宋叶志，茅永兴，赵秀杰编著.一北京：清华大学出版社，2011.2

ISBN 978-7-302-24706-7

I. ①F… II. ①宋…②茅…③赵… III. ①FORTRAN 语言一程序设计 IV.①TP312

中国版本图书馆 CIP 数据核字（2011）第 007584 号

**责任编辑**：夏非彼　夏毓彦
**责任校对**：闫秀华
**责任印制**：杨　艳
**出版发行**：清华大学出版社　　**地　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　编**：100084
**社 总 机**：010-62770175　　**邮　购**：010-62786544
**投稿与读者服务**：010-62776969，c-service@tup.tsinghua.edu.cn
**质量反馈**：010-62772015，zhiliang@tup.tsinghua.edu.cn
**印 装 者**：清华大学印刷厂
**经　销**：全国新华书店
**开　本**：190×260　**印　张**：33.75　**字　数**：864 千字
附光盘 1 张
**版　次**：2011 年 2 月第 1 版　　**印　次**：2011 年 2 月第1 次印刷
**印　数**：1～4000
**定　价**：69.00 元

产品编号：039077-01

# 前　言

科学计算是现代自然科学研究与工程技术的一个基石。可以认为科学计算方法属应用数学的一个分支，然而又不限于数学本身，其内容具有广泛性和一般性，在众多科学中都需要有科学计算方法的支持，而现代科学技术的进步及应用需求又反过来促进科学计算理论与方法的发展。可以这样说，就当今而言，科学计算能力的水平在一定程度上反应了一个国家或地区科学技术的发展水平。

抛开计算机硬件技术的发展不谈，就科学计算领域而言，曾经出现了较多计算机语言如 Fortran、Pascal、C、C++、Basic，及专门的数学软件 MATLAB、Mathimatica、Maple、SciLab 等。对于科学计算而言，建议读者优先学习 Fortran 及 MATLAB 语言，Fortran 在科学计算领域庞大的用户基础是其他语言所无法比拟的，尤其是对于老一辈科学家，对他们而言 Fortran 更是犹如母语一般。就作者经历而言，MATLAB 数值功能虽强大，但一般多用来仿真、模拟与快速分析。对于大型程序，主要是以 Fortran 居多，如数值气象预报程序、航天器轨道计算、许多有限元软件的编写程序等，这些大型程序一般都涉及非常复杂的科学计算方法。近些年，我国大范围普及 C、C++、Java、C# 等语言，造成介绍 Fortran 及其科学计算方面的书籍较少。

目前，我国大学理工科本科生、研究生及博士生普遍开设数值分析、计算方法或科学计算与工程等课程（计算数学系一般代之以函数逼近、数值代数等课程）。而这些教材中，往往没有提供让学生可以参考的程序，这对刚学习完高等数学与线性代数的同学学习科学计算方法造成了一定的困难。对于刚学习完高等数学与代数的同学，往往一开始学习科学计算方法是较为困难的，这困难主要来源于两个方面，一是使用数学工具的手段还不是很成熟，二是刚开始面对数值问题编程时感到不习惯。鉴于此，本书提供了常用数值方法的算法及完整代码。

当然本书并不是仅仅为初学者准备的，有经验的编程用户，对于某些问题也可以直接使用书中的程序或者稍作改动即可拿来就用。

## 内容安排

数值分析本身的内容广义地讲，可以包含很多内容，如偏微分方程、数值最优化等，而这些内容往往都可以单独成为一门学科或专题。通常的共识是数值算法所涉及的范围是直至偏微分方程但不包含偏微分方程的内容，而本书即主要介绍这部分内容。除了本书引子之外，全书共 12 章，其中前 11 章介绍计算方法的一般性内容，最后一章作为范例，仅仅起到示范性作用。

关于章节的安排，我们做了仔细的考虑，按照现在的顺序，保证了阅读时章节之间的依赖性不会对读者造成障碍，又使本书主要按照数值代数和函数逼近的两大知识块的顺序讲解。而如果先讲函数逼近的话，里面部分内容（如样条插值等），需要用到数值代数的知识。

在前 3 章分别介绍了线性方程组的直接方法、迭代方法、最小二次与数据拟合，其中最小二乘部分有些内容很自然地需要线性方程组的知识做基础。第四章特征值问题，不需要依赖于其他章

节，因为其正交变换部分也可以放在该章单独讲解，因此可以放在全书的任何地方，但考虑到数值代数部分的整体性，把它放在了第4章。

第5章非线性方程的解法不需要依赖其他章节的内容，而第6章的非线性方程组的数值方法则需要读者最好理解第5章的内容，同时计算非线性方程组必须具备线性方程组的知识，因此我们把第6章安排在线性方程组与非线性方程之后介绍。因为非线性方程组的应用非常广泛（如非线性回归等领域），我们介绍了较多的并且实用的数值方法，尤其是给出了多种变形的拟牛顿方法，如Broyden、DFP、BFS等，还介绍了拓展收敛域的数值延拓法及参数微分法，这些算法涉及一定的编程技巧，在同类的书籍中较少有如此详细介绍的。我们把非线性最小二乘问题，放到应用范例一章以卫星导航原理为实例介绍。

接下来分别介绍了插值法（第7章）、数值微分（第8章）、数值积分（第9章）、常见的特殊函数计算（第10章）及常微分方程数值方法（第11章）。其中插值法介绍了最重要及最基础的Lagrange、Newton、Hermite插值方法，并把重点放到了样条插值方法。其中B样条因其理论优美、实用性强应该得到重视，而国内许多教材中较少有介绍，给出程序的则更少。关于特殊函数的计算，我们介绍了几种典型的特殊函数，并未求全而罗列更多的计算方法。常微分方程问题介绍了单步法、一般多步法及预测校正方法（PECE），作为基础，我们认为够了，当然在实践中，可能根据具体的工作需求而需要构造更复杂的算法，但基本原理大抵如此。

根据以往读者的需求，他们希望安排一章内容列举一些范例，以说明科学计算方法是有“用武之地”的，实际上我认为这个艰巨的任务，应该交给读者自己。作者知识面有限，只能举一两例做示范性的说明。

为了给编程经验不是很丰富的初学者提供更多的有效帮助，我们特地安排了附录部分讲解了，集成开发环境下编程的最基础知识及程序调试方法。

## 致谢

本书的编写得到了单位领导及动力中心各位老师的帮助和支持，尤其感谢空间飞行器精密定轨课题组首席科学家胡小工研究员对我的帮助。在书的早期编写中，还得到了总参某部陈国平同志及北京航天飞控中心曹建峰博士的帮助。在编写过程中好友徐导，杨建平夫妇、复旦大学数学学院杨卫红副教授及陈清女士、王婷婷夫妇、Tiffany、Phebe Gray博士等也给予作者许多鼓励。北京图格新知公司夏非彼老师为本书的出版付出了辛勤的劳动，作者致以诚挚的谢意。

囿于作者水平有限，加上时间仓促，书中肯定有不少疏漏，甚至谬误之处，还望读者不吝指教。作者的电子邮箱为：song.yz@163.com，也可以通过QQ：1556207030进行答疑。

如果有院校教师愿意采用本书作为教材或指定参考书的，可以和清华大学出版社联系或直接与作者联系，编者愿意提供Fortran程序设计或计算方法一定的技术支持。

宋叶志　中国科学院

茅永兴　中国卫星海上测控部

# 引　子

## 1. 科学计算方法有何应用？

至于科学计算法有何应用，这显然不是三言两语可以回答的。这里先对科学计算方法做一通俗介绍。在不严格的情况下，有时我们不去区分计算方法、科学计算、数值分析几个名词的差别。目前，我国理工科院校的计算数学、计算机等专业在本科生就开设了数值分析课程。而绝大部分院校理工科研究生和博士生都开设了计算方法的课程。科学计算对于个人而言，也是从事科研工作或工程技术的一项基本技能。

可以说从现在的航天器飞行、自动控制、大型桥梁设计等都离不开科学计算技术的支持。如果没有卫星飞行我们又如何能看上卫星电视？如何能使用卫星定位？可以说身边的许多事物都在直接或者间接的与科学计算有联系。现在许多社会学领域也离不开科学计算，比如医学统计方法、传染病动力学、证券金融时间序列分析等都需要依赖一定的数值方法实现。

当今社会科技日益发展，而许多高新技术都离不开科学计算的支持。上世纪中叶以来，计算机软硬件的发展为科学计算提供了有利条件。目前，各国科技领域都普遍非常重视科学计算理论与技术的发展。

科学计算具有一般性的理论意义，并不局限于某具体工程技术，而许多具体的科学实践领域反过来对进科学计算的发展提出了新的课题和需求，从而两者互相促进。

## 2. 该选择哪种编程语言？

现今，在使用的计算机高级语言种类繁多。对于编程经验不是很多的人而言，一个很自然的问题是：“我该学习哪种编程语言?”

我们给出直接的回答是，没有统一的答案。具体选择哪种语言是因人而异，因需求而异同，没必要厚此薄彼。

一般而言每种语言都有某一些领域比较突出。比如 C 语言编写操作系统很出名，现在的 C#编写界面非常方便，SQL 语言编写数据库程序，R 语言、SAS 编写统计程序等。

对于有意从事科研或工程技术的读者，建议优先学习 Fortran 和 MATLAB，最好两种语言都能比较熟练的掌握。

Fortran 从诞生就一直是科学计算领域最重要的语言之一，拥有庞大的用户群体。第一版的 Fortran 是由 IBM 在上世纪 50 年代为自己的 704 计算机开发的。在那之前，基本所有的计算机程序都是通过手工编写的机器语言，非常容易出错。Fortran 的出现可以说是科学计算领域的一次革命，从此程序员能像写数学公式似的编写程序。

到目前为止 Fotran 已经经过数次较大的修改，不过由于良好的兼容性，一般不太会给程序员带来较大的麻烦。现在使用较多的是 Fortran90 之后的版本，包括 95 版和 2003 版。但是 Fortran77 依然有相当多的人在使用。目前，国内外众多科研机构的许多大型程序都是用 Fortran 编写的。

## 3.计算方法是否局限于编程实现？

一个简洁的回答：当然不是。

计算方法更重要之处，在于提供一整套能够武装人的大脑思维的分析理论与方法，使之可以灵活运用于实践问题。而如果购买计算方法的书籍，仅仅是为了获得一套可执行的代码，这只实现

了作者编写本书目的的一部分价值。为了表明算法的重要性，下面举一个简单的例子。

【问题】仅仅给你一张纸和笔，让你计算 $\sqrt{447}$ ，要求精确到小数点后 4 位有效数字。当然，如果你需要还可以提供给你只具有加、减、乘、除功能的小计算器。读者可以先不看解答，考虑一下这个问题，如果你仅仅有笔和纸，不一定要求你算出来，但是你要给出一个方法，要求是按照这个方法原则上是你可以算出来。

【解答】这是作者以前在某高校 BBS 上看到的一个问题，然后我给了提问者一个公式，即

$$x_1 = \frac{1}{2}\left(x_0 + \frac{447}{x_0}\right)$$

并给他介绍了公式使用方法，方法如下：

给定一个初始的你认为最接近的“猜想”值 $x_0$，然后把这个猜想的值带入等式右边，算出来 $x_1$ 。然后再把 $x_1$ 当作 $x_0$，带入等式右边，算出新的 $x_1$，如此反复循环。可以把最后一次的 $x_1$，当作最终结果，即可以得到很高的精度。上面的公式加使用方法，已经构成了完整的计算方法。

实际上，可以把上述公式等价于下述迭代格式，即

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{447}{x_n}\right), n = 1, 2, \ldots$$

下面介绍一下以上公式的使用过程。

步骤一：$\sqrt{447}$ 等于多少，你不清楚。但是 $21^2=441$ 你应该知道。这样你就令 $x_1=21$。

步骤二：由上面的公式，可以算出

$$x_2 = \frac{1}{2}\left(21 + \frac{441}{21}\right) = 21.142857142857142$$

步骤三：再把 $x_2 = 21.142857142857142$ ，带入到公式中，可以求得

$$x_3 = \frac{1}{2}\left(x_2 + \frac{441}{x_2}\right) = 21.142374517374517$$

这时，可以就把 $x_3$ 作为最后结果。$\sqrt{447}$ 的准确结果 21.142374511865974…。我们才作 2 次复合的四则运算，精度已经达到小数点后 8 位有效数字。

在以上计算过程中，仅仅是利用了加、减、乘、除法运算，并未采用更“高级”的运算方法。

有人疑问说：“我没想起来 $21^2=441$”。如此的话，$20^2=400$，这个总应该清楚的吧。在上面的公式中，就令 $x_1=20$。

$$x_2 = \left(20 + \frac{447}{20}\right) = 21.175$$

$$x_3 = \frac{1}{2}\left(x_2 + \frac{447}{x_2}\right) = 21.142399645808737$$

$$x_4=\frac{1}{2}\left(x_3+\frac{447}{x_3}\right)=21.142374511880917$$

$$x_5=\frac{1}{2}\left(x_4+\frac{447}{x_4}\right)=21.142374511865974$$

$$\vdots$$

可见，我们做了几次复合四则运算之后，精度已经达到 10 几位有效数字，远远超过我们的预期精度需求。

至于以上公式是怎么来的，这正是学习数值方法之后应该具备的最基本能力，说出来最简单不过了。既然是求 447 开根号的问题，就可以建立 $x^2$=447，这样一个方程（负根舍弃），剩下的工作，就是非线性方程一章中的内容了。

以上只是一个简单的范例，事实上关于数值算法的很多内容，早在计算机发明之前就已经被科学家所发现并运用了，从许多算法的名字也可以看出，牛顿迭代法、拉格朗日插值法、高斯积分法等等。高斯在曾就用手算实现最小二乘法而达到参数估计的目的。这些先贤，即便是没有数字计算机，依然是第一流的数值分析专家。

在数字计算机发明之前对于非线性微分方程有摄动分析解法，同样也有人用手工的方法进行微分方程的数值计算，当然这样的代价是相当高昂的。总之，算法是数值方法的灵魂，而编程仅是一个实现算法的过程。

这里没有意图要说明计算机编程不重要，而是同时强调编程方法和算法理论都是很重要的。

### 4. 学习计算方法需要准备哪些基础知识？

本书读者的起点是具备微积分（数学分析）、线性代数、矩阵论和概率统计部分知识、泛函分析少量知识。泛函分析的知识不是必须的，适当的时候我们会交代一下，当然如果是用泛函分析的语言去叙述，很多问题会更简洁明了，并更具备概括性。总之，要想学习好科学计算方法，需要对微积分、矩阵等知识比较熟悉。另外一点是，假定读者已经掌握了编程方法。

编写本书目的之一，是考虑到许多初学者，因为没有足够的数值程序范例，入门就很困难，更无从谈提高了。对于没有入门的读者，可以先拷贝光盘上的程序，在自己计算机上试运行，然后修改相关参数等看计算结果，逐步入门。当然这并不是建议大家都不需要编写程序，而直接拿书中的程序去应付自己的工作需求。要想真正学好科学计算方法，还是需要自己动手编写一定量的代码。这样才能对数学原理又更深刻的认识和体会。

### 5.书中程序是否有 Bug？

当然有 Bug，这是毫无疑问的。

Bug 本意是臭虫、缺陷、损坏等意思。现在在较大的计算机程序中，经常会隐藏着的一些可能是未被发现的缺陷、问题或漏洞统称为 Bug。早期的计算机是由许多庞大且昂贵的真空管组成，并利用大量的电力来使真空管发光。一次科研试验中，一只小虫子 Bug 钻进了一支真空管内，导致整个计算机无法工作。研究人员费了相当长时间，总算发现原因所在，把这只小虫子从真空管中取出后，计算机又恢复正常。后来，Bug 这个名词就沿用下来，表示电脑系统或程序中隐藏的错误、缺陷、漏洞或问题。

与之相对应，人们将发现 Bug 并加以纠正的过程叫做 Debug，有时候我们就称为调试，意思

是捉虫子、杀虫子。

一般而言，程序员并不试图一次性把程序写得完全正确，而是通过反复的“Debug”排除错误。关于如何 Debug 是一个技巧性很强的工作，也是编程所应该必须具备的技能，在附录部分，我们介绍了常见的 Debug 方法。

回到本段的问题，虽然我们对代码做了多次编译、测试，但是我相信里面一定还有不少 Bug。即便是目前世界上最优秀的数值方法库，他们也不敢说自己的程序没有 Bug。像 MATLAB 这样专业成熟的商业软件，现在每年发布两次软件中存在的 Bug。所以，虽然我们的程序读者可以直接使用，但是我们仍然希望读者了解计算方法。

在这一点上，许多人喜欢以“黑箱子”来形容这件事，即用户并不需要知道“黑箱子”是如何实现的，仅仅想知道进去什么，可以出来什么。我们的观点是，我们提供一套我们自己编写的“黑箱子”，声明一点，我们做的可能不一定太好，但是我们试图把“黑箱子”打开，让读者自己从中寻找需要的东西。

鉴于以上的考虑，在编程时，我们对代码的注释是比较详细的，这为读者理解程序提供了方便。

### 6. 什么是面向对象？如何使用书中的程序？

我们的程序主要以 Fortran 90/95/2003（以下简称 F90/95/2003）编写，对于 F2003 我们持保留的态度，不过多的涉及其中的语法，因为其中的一些语法在部分编译器上还不支持。

如果是 Fortran 77（以下简称 F77）的用户，使用部分程序可能需要做较大的改动。F90/95 在 F77 一些容易犯错的地方，虽然允许使用，但是不建议使用，而我们编写时则回避了新版本建议舍弃的语法。另外新版本有一些新的语法，在老版本上是不支持的，如新版大大加强了数组（某种意义下相当于矩阵）的处理方式。

全书采用模块化方法编写，这里的模块化不仅仅是一种语法功能，且可以认为是一种面向对象的思想。F2003 则支持了更为丰富的面向对象功能，如前面所言，我们对这些语法持保留态度，不一味的求新。

面向对象的好处之一是可以进行数据封装，我们考虑到大部分用户编程习惯仍然是结构化程序设计方法，所以我们尽量只是采用面向对象的形式，而实际上我们的过程都是可以单独拿出来使用。当然，少部分程序，我们作为范例把变量放在模块中，如果有必要的话只需要把这些变量复制到对应的函数中即可。有些章节的程序，可能需要依赖其他章节的介绍的程序，我们把所依赖的函数拷贝到当前章节的程序中，这样做的一个坏处是增加了少量的篇幅，好处是保证每一章节的独立性与完整性。

因为目前国内大部分用户使用的系统依然是 Windows，我们主要的编程环境是采用 Visual Studio 2008+Intel 11 编译器。目前 Visual Studio 2010 还不支持 Intel 编译器的集成。当然我们的大部分程序在 Compaq Fortran 6.6 这个经典 IDE 上也能编译通过，只是有些少量语法 Compaq Fortran 还不支持，需要略做修改，在适当的地方我们做了介绍。部分程序在 Sun 公司的 Solaris Unix F90 编译器编译测试过。需要说明的是集成开发环境本身对 Fortran 语言是没有影响的，但是一个好的集成开发环境为程序员提供非常高效的工程条件。

# 目 录

# 第 1 章

# 矩阵分解与线性方程组的直接方法

自然现象大部分表现为非线性，而很多情况下，我们对于非线性问题是通过线性化来处理，故而在众多学科中，线性问题依然是最基础最重要的内容之一。本章将介绍线性方程组的数值方法。线性方程组的数值方法是数值计算的一个基础内容，在非线性方程组、微分方程边值问题、样条逼近等许多领域中，都需要用到线性方程组的处理方法。

线性方程组主要分为直接方法与迭代方法。本章介绍直接方法，同时介绍与之相应的常见的矩阵分解方法，关于迭代法将在下一章中介绍。

## 1.1 三角方程组

### 1. 实验基本原理

线性方程组的计算方法有多种，针对不同特点的方程采用不同的方法计算效率往往是不同的。线性方程组中最简单的是对角形方程与三角形方程，其中对角形方程几乎都不需要计算机，只要拿笔和纸进行简单的四则运算，就可以得出结果。三角形方程也比较简单，仅仅是一个回带的过程，即便是没有学过数值分析的读者，只要略具备代数知识即可自行给出算法。

然而很多解方程的方法都是先把一般性方程化成三角形方程，这样便可以方便地处理。因此，计算三角形方程便是数值线性代数的一个较简单而又基础的问题。

三角形方程分为上三角形和下三角形两种，其计算方法基本相同，仅仅是次序不同而已。

对于上三角形方程，计算步骤为：

$$\begin{cases} x_n = \dfrac{b_n}{a_{nn}} \\ x_i = \dfrac{b_i - \sum\limits_{k=i+1}^{n} a_{ik} x_k}{a_{ii}}, i = n-1, n-2, \cdots, 1 \end{cases}$$

对于下三角形方程，计算步骤为：

$$\begin{cases} x_1 = \dfrac{b_1}{a_{11}} \\ x_k = \dfrac{b_k - \sum\limits_{i=1}^{k-1} a_{ki} x_i}{a_{ii}}, \quad k = 2, \cdots, N \end{cases}$$

下面通过实验介绍上、下三角形矩阵的计算。

## 2. 实验目的与要求

- 理解上、下三角形方程组的计算流程。
- 能够编程实现三角形方程组的计算。

## 3. 实验内容与数据来源

计算上三角方程组

$$\begin{pmatrix} 2 & 1 & 3 & 4 \\ 0 & 3 & 2 & 5 \\ 0 & 0 & 7 & 3 \\ 0 & 0 & 0 & 2 \end{pmatrix} \boldsymbol{x} = \begin{pmatrix} 50 \\ 49 \\ 53 \\ 12 \end{pmatrix}$$

计算下三角方程组

$$\begin{pmatrix} 2 & 0 & 0 & 0 \\ 1 & 3 & 0 & 0 \\ 1 & 5 & 1 & 0 \\ 3 & 2 & 1 & 4 \end{pmatrix} \boldsymbol{y} = \begin{pmatrix} 10 \\ 23 \\ 39 \\ 43 \end{pmatrix}$$

## 4. 函数调用接口说明（如表 1-1、表 1-2 所示）

表 1-1　uptri 函数调用接口说明

| 输出参变量 | 数据类型 | 变量说明 |
|---|---|---|
| X | REAL*8(N) | 上三角方程的解 |
| **输入参变量** | **数据类型** | **变量说明** |
| A | REAL*8(N,N) | 上三角系数矩阵 |
| B | REAL*8(N) | 右向量 |
| N | INTEGER | 方程组的维数 |

表 1-2　downtri 函数调用接口说明

| 输出参变量 | 数据类型 | 变量说明 |
|---|---|---|
| X | REAL*8(N) | 下三角方程的解 |

| 输入参变量 | 数据类型 | 变量说明 |
|---|---|---|
| A | REAL*8(N,N) | 下三角系数矩阵 |
| B | REAL*8(N) | 右向量 |
| N | INTEGER | 方程的维数 |

## 5. 程序代码

```
module tri_eq
!----------------------------------------module coment
!  Version     :  V1.0
!  Coded by    :  syz
!  Date        :  2010-4-8
!------------------------------------------------------
!  Description : 用于解上、下三角形线性方程组的回带方法模块
!
!------------------------------------------------------
!  Contains    :
!      1.   solve  方法函数
!      2.
!------------------------------------------------------
contains
subroutine uptri(A,b,x,N)
!---------------------------------subroutine  comment
!  Version   :  V1.0
!  Coded by  :  syz
!  Date      :  2010-4-8
!------------------------------------------------------
!  Purpose   :  上三角方程组的回带方法
!               Ax=b
!------------------------------------------------------
!  Input  parameters  :
!      1.   A(N,N)系数矩阵
!      2.   b(N)右向量
!      3.   N 方程维数
!  Output parameters  :
!      1.  x  方程的根
!      2.
!  Common parameters  :
!
!-----------------------------------------------------
implicit real*8(a-z)
integer::i,j,k,N
real*8::A(N,N),b(N),x(N)
x(N)=b(N)/A(N,N)
!回带部分
do i=n-1,1,-1
   x(i)=b(i)
  do j=i+1,N
   x(i)=x(i)-a(i,j)*x(j)
  end do
   x(i)=x(i)/A(i,i)
end do
end subroutine uptri
```

```
subroutine downtri(A,b,x,N)
!---------------------------------subroutine  comment
!  Version    :  V1.0
!  Coded by  :  syz
!  Date      :  2010-4-9
!-----------------------------------------------------
!  Purpose   :  下三角方程组的回带方法
!               Ax=b
!-----------------------------------------------------
!  Input  parameters  :
!      1.   A(N,N)系数矩阵
!      2.   b(N)右向量
!      3.   N 方程维数
!  Output parameters  :
!      1.  x  方程的根
!      2.
!  Common parameters  :
!
!----------------------------------------------------
implicit real*8(a-z)
integer::i,j,N
real*8::A(N,N),b(N),x(N)
x(1)=b(1)/a(1,1)
do k=2,N
  x(k)=b(k)
  do i=1,k-1
     x(k)=x(k)-a(k,i)*x(i)
  end do
  x(k)=x(k)/a(k,k)
end do
end subroutine downtri
end module tri_eq
!##############################################################
module driver
!----------------------------------------module coment
!  Version      :  V1.0
!  Coded by     :  syz
!  Date         :
!-----------------------------------------------------
!  Description :  驱动函数模块
!
!-----------------------------------------------------
!  Parameters  :
!     1.
!     2.
!  Contains     :
!     1.       dir_main  驱动函数入口
!     2.       dri_up    当读到关键字 uptri 时启动该函数
!     3.       dri_down  当读到关键字 downtri 时启动该函数
!-----------------------------------------------------
!  Post Script :
!     1.
!     2.
!-----------------------------------------------------
contains
```