高等学校教学用书

数值方法教程

刘钦圣　张晓丹　王兵团　编

GAODENG XUEXIAO JIAOXUE YONGSHU

冶金工业出版社

高等学校教学用书

数值计算方法教程

刘钦圣　张晓丹　王兵团　编

北　京
冶金工业出版社
1998

图书在版编目(CIP)数据

数值计算方法教程/刘钦圣等编. —北京:冶金工业出版社
1998.8
高等学校教学用书
ISBN 7-5024-2155-6

Ⅰ.数… Ⅱ.刘… Ⅲ.数值计算-高等学校-教材 Ⅳ.0241

中国版本图书馆 CIP 数据核字(98)第 00778 号

出版人 卿启云 (北京沙滩嵩祝院北巷 39 号,邮编 100009)
冶金工业出版社印刷厂印刷 ;冶金工业出版社发行;各地新华书店经销
1998 年 8 月第 1 版,1998 年 8 月第 1 次印刷
850mm×1168mm 1/32;8.75 印张;230 千字;270 页;1—1500 册
14.70 元

前　言

随着电子计算机应用的普及与发展，科学计算已与科学理论、科学实验并列为现代科学的三大组成部分，因而数值计算方法的内容也愈来愈广泛和丰富了。

本书是为高等院校非计算数学各专业的研究生和高年级本科生学习“计算方法”课程所编写的教材，主要从实用的角度介绍现代科学技术与工程设计中常用的数值方法和理论。对每种数值方法在讲清基本原理的前提下，突出如何构造和分析具体算法，并写出详细的计算步骤或非形式语言程序，辅以典型例题；同时对计算工作量、收敛性、稳定性、误差估计、适用范围以及优缺点等进行简要的论证和评述。目的是帮助读者理解和掌握各个方法的全过程，使读者不仅对方法的原理、构造和应用有较深切的理解，而且能独立地在计算机上实现这些方法，从而为今后应用数值方法解决实际问题打下良好的基础。另外，本书从教学法考虑，力求在叙述上循序渐进，推导上简明易懂，每章前有概述，后有小结，并附有适当数量的习题，书后给出部分习题的参考答案。

全书共分九章，第 1～5 章由刘钦圣编写，第 2～4 章由王兵团编写，第 6～9 章由张晓丹编写，最后由刘钦圣统一修改定稿。为适应不同课时的要求，本书内容具有较大的伸缩性，全部讲授约需 60～72 学时；删去某些章节后，也可适用于 36～52 学时的教学需要。

本书初稿曾在北京科技大学研究生中试用过多次。此次出版前又进行了全面的修改和整理，并经过北方交通大学陈立成教授和北京科技大学孙仁济教授的仔细审阅，他们提出了很多宝贵的意见和建议，谨在此向他们表示衷心的谢意。

由于我们的水平有限，本书的缺点错误在所难免，敬希读者批评指正。

编　者

1997 年 7 月于北京

目　　录

第一章 概 论

本章简要地介绍计算数学的一些基本概念。包括计算数学的对象、应用和发展，计算机中数的浮点运算，误差的基本概念、问题的性态以及算法的数值稳定性等。它们将贯穿到本课程的全部内容中。

§1.1 计算数学引论

在生产斗争和科学试验中，许多现象的定量分析可以归结为求解特定的数学问题。一般说来，这些问题往往具有非常复杂的形式，很难甚至不可能用解析方法求得精确解，只好求助于近似方法求出数值解。在电子计算机出现以前，手工计算需要成年累月，有些问题甚至终生难以解出，而现在利用计算机很多问题都容易解决了。

利用计算机解决实际问题，通常要按以下步骤进行：

(1)建立数学模型。首先将所讨论的实际问题，根据专业知识进行分析，突出主要因素，忽略次要因素，用数学语言描述出来，确定自变量和因变量并建立它们之间的关系。同时，根据某些规则和实际经验简化得出的模型，最后构造成一个适定的数学问题。它可能是一个方程组，一个函数的极小化，一个积分计算或一个微分方程等。

(2)选择数值方法。建立数学模型之后，根据问题的性质，选取一个或几个数值计算方法。对所选方法，希望有适当的精度、有效性和稳定性。最好在计算之前，先从理论上分析：(a)所用方法能否达到所要求的精确度；(b)方法的计算量会不会太大，程序的实现是不是方便；(c)方法对数据的微小摄动反应是否灵敏。分析清楚后，就可考虑实现所选方法的算法。

(3)编写程序。把算法译成计算机程序。现在的程序一般用高

级语言编写，例如 FORTRAN、PASCAL、BASIC 与 C 语言等等。

(4)上机计算。不妨先用各种各样的数据去检验程序，为了评估计算的质量，将这些试算的结果与已知结果以及问题的基本原理进行对照。若是满意的，再从提高效率、简化及典型化方面来加工程序，使之成为标准化文件；若计算结果不满意，则返回第(2)或第(3)步去复查。

本书的任务主要是讨论上述第(2)步骤的问题，即介绍现代电子数字计算机上常用的数值计算方法。

1.1.1 计算数学的对象

计算数学是一门最古老的数学，人类在原始社会就会用手指、结绳等方法进行数值计算。我国古代在计算数学上曾有辉煌的成就。公元前 14 世纪的商代就基本形成了十进制。春秋战国出现了算筹，以后改进成算盘，相应地产生了珠算方法。北魏的刘徽提出的“割圆术”，计算出圆周率 $\pi=3.1416$，公元五世纪南朝刘宋的祖冲之发展了缀术，并算出更为精确的 π 值，领先于世界 1000 多年。公元前一世纪汉代《九章算术》一书，已经记载有开平方与开立方的算法，所述求解一元二次方程的方法就是 13 世纪欧洲出现的“试位法”；书中还有一章讲述联立一次方程组的解法，实质上就是近代的“消元法”。这比欧洲早出现 1000 多年。

计算数学又是一门很年轻的数学。它是随 1946 年电子计算机的诞生、发展和广泛应用而逐步发展形成的。因此，今天我们所说的计算数学，可以简单地说成是计算机上的数学方法。如果完整而严格地来说明计算数学的对象，不妨定义为：它是研究数值计算方法的设计、分析和有关的理论基础与软件实现的一个数学分支。通常我们把计算数学称为**计算方法**或**数值分析**。其内容大致可分成两大方面：一是连续系统的离散化，一是离散型方程的数值求解。其基本概念包含：误差、收敛性、稳定性、计算量、存贮量以及自适应性等。如果从数学问题的来源或类型来看，则计算数学有如下的分支：数值代数、数值逼近、计算几何、数学物理方程的数值解以及计算概率统计、最优化方法等等。除最后两者外，本书将分别介绍

它们的基本内容。

1.1.2 计算数学的应用与发展

不论在现代自然科学、工程技术或是工农业生产、国防现代化等国民经济各个领域中，各种各样的实际问题，往往需要得到可靠的数值结果，这就离不开计算数学。特别是天体物理、大气研究、分子生物、集成电路设计、现代武器研制、密码分析和天气预报等都离不开大型的数值计算。因此，计算数学的应用是非常广泛的。

随着计算机向微型和巨型化方向的发展，自动化、最优化设计更加普及与广泛，科学研究与工程设计的手段，逐渐由模拟实验向数值计算的方向转变。因为当今计算所取得的成果，其精确可靠性已经接近、达到甚至超过实验结果的水平。而实验往往费人、费时、费钱，并且在某些特殊条件下，是非常艰难和危险的。此时，计算就成为十分关键甚至是唯一可行的手段。因此，自 80 年代以来，科学计算的意义与重要性日益引起人们的重视，它已与科学理论、科学实验并列为现代科学的三个组成部分。一系列与计算数学相关的边缘学科，如计算力学、计算物理、计算化学、计算生物、计算地质与计算经济学等，也相继出现了。

科学计算的物质基础是计算机，其理论基础是计算数学。现代计算能力的大幅度提高，既来自计算机的改进，也来自计算方法的进步。计算方法的发展，促进计算机体系结构的更新，而计算机的更新换代，也对计算方法提出了新的标准和要求。在 1946 年第一台计算机问世以后的几十年里，均按 Von Newmann 的概念结构，即每一时刻只能按一条指令对一个数据进行操作，这种串行处理限制了计算速度的提高。70 年代初，并行处理机的出现和发展，导致了计算方法的一种新的分类。传统的数值方法都是为适应串行处理计算机而研究的，可称为**串行算法**；而新的面向并行处理计算机的数值方法，则可称为**并行算法**。改造现存的有效算法及数值软件包使之适应并行计算机，特别是设计新的高效率的并行算法，已成为计算数学中当前最活跃的新领域。

§1.2 算法及其效率

1.2.1 什么是算法

计算数学的任务主要是用计算机计算出实际问题的数值解。它不满足于一般地讨论已知量与未知量的依存关系，而是要求根据给定的已知量的数据去把未知量的数值计算出来。因此，它关心的是数学中的构造性方法，即不仅说明解的存在，而且要求提供如何求解的具体方法。

由于电子计算机实质上只会做加减乘除等基本运算，研究如何通过计算机所能执行的基本运算，求出各类实际问题的数值解，就是计算数学的根本课题。从给定的已知量出发，经过有限次的基本运算及规定的运算顺序，能够计算出所求未知量的数值解，这样构成的完整的计算步骤，称为**算法**。计算方法的任务和目的，狭义地说，就是研究和构造算法。

有许多问题的解，不可能经过有限次算术运算计算出来，例如计算任意角的三角函数值，求非线性方程 $f(x)=0$ 的根，求一般微分方程的解。这类问题常常采用近似替代的办法，即把它们转化成比较简单的、可用有限次运算求解的问题。用简化问题的解作为原来问题的近似解，这就产生了误差估计和收敛性问题。因此构造和分析算法，必须讨论它的收敛性和误差估计，同时要注意算法的数值稳定性，尽力节省内存和计算工作量。这些都是评价一个算法优劣的标准。当然，要求一个算法十全十美是不现实的，往往是为了改善某一个方面而要降低另一方面作为代价。

1.2.2 算法的计算量

现在我们讨论算法的计算工作量，它是衡量算法好坏的一个重要标准。因为这关系到算法的计算效率问题。为简便计，通常以计算机完成一次浮点加(减)法和乘(除)法所需时间作为一个时间单位，称为**浮点运算**，记作 flop(floating point operations)。我们用所需浮点运算的次数来度量算法的效率。由于大多数算法的加减法次数，大致与乘除法次数相同，而且计算机做加减法比乘除法快

得多。因此，我们实际上仅需统计乘除法的次数，就可以作为算法的计算工作量。

例 1.2.1 计算 x^{255}

按原型计算，需要的计算量 $N=254$flop，如果改用 $x^{255}=x\cdot x^2\cdot x^4\cdot x^8\cdot x^{16}\cdot x^{32}\cdot x^{64}\cdot x^{128}$，只需要 $N=14$flop。

例 1.2.2 设 A,B,C,D 分别是 10×20、20×50、50×1、1×100 的矩阵，试按不同算法求矩阵乘积 $E=ABCD$。

解 根据矩阵乘法的结合律，采用下列三种算法：

(1)$E=[(AB)C]D$

(2)$E=A[B(CD)]$

(3)$E=[A(BC)]D$

它们的计算量分别是

(1)$N=11500$flop

(2)$N=125000$flop

(3)$N=2200$flop

显然算法(3)的效率最高。

由于算法的效率，反映的是计算工作量问题，因此由算法的效率可以判定这个算法是否可行。例如用 Cramer 法则求解 n 元线性方程组，要计算 $n+1$ 个行列式，还要进行 n 次除法。按行列式定义，计算一个 n 阶行列式的计算量约为 $(n-1)(n!)$flop，因此，求解 n 阶线性方程组的总计算量是

$$N=[(n+1)(n-1)(n!)+n]\text{flop}$$

当 $n=20$ 时，N 约为 9.707×10^{20}，如果在每秒 1 亿次运算的计算机上计算，也需要 $t=9.70728\times10^{12}$ 秒，即使每天 24 小时不停的运算，也需要 30 多万年，显然，这个算法对高阶线性方程组是毫无实用价值的。

1.2.3 非形式语言

在本节的最后，我们介绍一种非形式语言(INFL)作为描述算法的工具。因为本书的任务是介绍计算机上常用的数值计算方法，每章每节都归结成构造和分析求解各种问题的具体算法上来，我

们的目的是要帮助读者理解和掌握各个算法的计算全过程，并能在计算机上实现。为此，需要有一种语言来描述算法。通常的程序设计语言，如 BASIC、FORTRAN、PASCAL 和 C 语言等，是描述算法的精确语言，可以直接在计算机上应用。但这类高级语言太完整详细，反而掩盖了算法的思路，因而在计算方法教程中不宜采用。我们介绍一种比较接近数学的非形式语言 INFL 来描述算法，它的特点是使用一些数学公式和符号(这在一些程序设计语言中是不允许的)，也使用一些自然语言。INFL 描述的算法将起着计算方法同计算机程序之间的桥梁作用。这种语言没有也不可能有严格的定义，我们将通过一些例子和以后各章中用 INFL 描述的算法实例，来说明它的一些特点。这对学过任何一种计算机程序语言的读者，不需什么说明就能看懂，并且借助它就可以顺利地用来编写出计算机语言程序。

例 1.2.3 设二矩阵 $A\in R^{l\times m}$，$B\in R^{m\times n}$，试写出它们的乘积 $C=AB$ 用 INFL 描述的算法。

解 设它们的元素分别用相应的小写英文字母表示，于是可写出所求的 INFL 程序：

算法 1.2.1

1 For $i=1,2,\cdots,l$

1.1 For $j=1,2,\cdots,n$

1.1.1 $c_{ij}=\sum_{k=1}^{m}a_{ik}b_{kj}$

例 1.2.4 讨论计算 n 次多项式 $p(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_1x+a_0$ 的算法，并写出其 INFL 程序。

解 如果先算各项的值然后相加求和，这种算法的浮点运算较多，共需 $N=\frac{1}{2}n(n+1)$flop；如果把 $p(x)$ 改写成

$$p(x)=x(x(x(\cdots(xa_n+a_{n-1})\cdots)+a_2)+a_1)+a_0$$

那么，上式可写成递推公式

$$p_n=a_n$$

$$p_k=xp_{k+1}+a_k,k=n-1,n-2,\cdots,1,0$$

这是著名的 Horner 算法(实际上此法最早由我国宋代数学家秦九韶提出,比 Horner 早六个世纪,因此,我们应称它为秦九韶算法)

算法 1.2.2 (秦九韶算法)

1 $a_n \Rightarrow p$

2 For $k=n-1,n-2,\cdots,1,0$

2.1 $xp+a_k \Rightarrow p$

它的运算量为 n flop。

例 1.2.5 求二次方程 $ax^2+bx+c=0$ 的实根。

解 此问题有两种算法:

算法 A: $x_{1,2}=\dfrac{-b\pm\sqrt{b^2-4ac}}{2a}$

算法 B: $x_1=\dfrac{-b-\mathrm{sgn}(b)\sqrt{b^2-4ac}}{2a}$

$$x_2=\frac{c}{ax_1}$$

下面我们将知道算法 B 是一个稳定算法,现写出其 INFL 程序

算法 1.2.3

1 $b^2-4ac \Rightarrow d$

2 If$d<0$ then 给出信息"无实根";stop

3 $(-b-\mathrm{sgn}(b)\sqrt{d})/(2a) \Rightarrow x_1$

4 $c/(ax_1) \Rightarrow x_2$

§1.3 机器数系

众所周知,数学分析中的数值范围是实数系,严格的极限理论是在实数理论的基础上建立起来的。实数系是一个连续系统,它是一个无限的、稠密的、连续的集合。实数的运算服从一般的运算律。而计算方法中的数是适应电子计算机上使用的机器数系,它是一个有限的离散集合,其分布是不均匀的,而且,在机器数的运算中,一般的运算律也并非都成立。我们在计算方法中是用这个残缺不全的机器数系,又在不完全满足一般运算律的条件下进行运算的,

因此我们先介绍机器数系的产生和特点。

在数值计算中，常把实数表示成浮点形式。例如 54.673，0.008402，−3.417 分别表示成 0.54673×10^{2} 或 0.054673×10^{3}；0.8402×10^{-2}，-0.3417×10^{1}

这样，一个数的数量级就一目了然。小数点的位置决定于后边 10 的指数，称为**阶码**。这种允许小数点位置浮动的表示方法，称为**数的浮点表示**，它由两部分组成

尾数部：	54673	8402	3417
定位部：	10^{2}	10^{-2}	10^{1}

定位部由阶码 2，−2，1 来确定。一般，任何一个实数 x 都可表示成

$$x=\pm0.a_1a_2a_3\cdots\times10^{c} \qquad (1.3.1)$$

这里，$0\leqslant a_1,a_2,a_3,\cdots\leqslant9$，$c$ 是整数。

在表达式(1.3.1)中，x 的小数点位置取决于阶码 c，例如 $54.673=0.54673\times10^{2}=0.054673\times10^{3}$ 可见阶码 c 变化时，小数点的位置随之浮动，所以叫做浮点表示，(1.3.1)中的 $a=\pm a\cdot a_1a_2a_3\cdots$ 是 x 的小数部分，称为**浮点表示的尾数**。

如果限定(1.3.1)中 $a_1\neq0$，即小数部分满足 $0.1\leqslant|a|<1$，则(1.3.1)称为**规格化的浮点表示**。

例如：0.54673×10^{2}，0.8402×10^{-2}，$\pi=0.31416\times10^{1}$

同样，我们可以定义以 β 为基的 t 位数的数系中数 x 的规格化浮点表示

$$x=\pm(0\cdot d_1d_2\cdots d_t)\times\beta^{c} \qquad (1.3.2)$$

这里 $\beta\geqslant2$ 是整数，通常取 $\beta=2,8,10,16$，又 $0\leqslant d_i\leqslant\beta-1$，且 $d_1\neq0$；t 是计算机的字长、阶码 c 是带有符号的整数，它有固定的下限 L 和上限 U，即 $L\leqslant c\leqslant U$。L,U 和 t 是由计算机的硬件所决定的常数。

这种由具体计算机限制的浮点表示的数的全体组成的集合，记作 $F(\beta,t,L,U)$，称为**机器数系**。对于不同型号的计算机，β,t,L,U 往往各不相同。可以证明集合 $F=F(\beta,t,L,U)$ 中共有 $2(\beta-1)\beta^{t-1}(U-L+1)+1$ 个规格化浮点表示的数(简称机器浮点数)，

它们是一个离散的分布不均匀的有限数集。在用计算机进行计算时，我们实际上只能对 F 中的实数做四则运算，而且 F 中的数对四则运算并不是封闭的，就是说 F 中任意二数的和、差、积、商不一定仍在 F 中。

如果初始数据或计算后的数据不属于 F，则计算机将按照不同的情形处理数据。设 m 与 M 分别是机器浮点数系 $F(\beta,t,L,U)$ 中的最小正数与最大正数。若 $x\notin F$，但 $m\leqslant|x|\leqslant M$，则可用 F 中最接近 x 的浮点数近似地表示 x，记作 $\mathrm{fl}(x)$。目前的计算机分舍入机和截断机两种，前者是按四舍五入原则取 x 的前 t 位数作为 $\mathrm{fl}(x)$，后者是直接取 x 的前 t 位数作为 $\mathrm{fl}(x)$。例如，$\pi\notin F(10,5,-77,77)$，则 $\mathrm{fl}(\pi)=0.31416\times10^{1}$（舍入机），$\mathrm{fl}(\pi)=0.31415\times10^{1}$（截断机）。若 $x\notin F$，且 $|x|>M$ 或 $|x|<m$，则机器无法接受这种数，分别称为**上溢**或**下溢**，统称**溢出**。

例 1.3.1 设 $F(2,3,-1,2)$ 中有二数 $x_1=0.100\times2^{2}$，$x_2=0.110\times2^{2}$，求 $x_3=x_1\cdot x_2$

解 $x_1\in F,x_2\in F$

$$x_3=(0.100\times2^{2})\times(0.110\times2^{2})=0.110\times2^{3}\notin F$$

此时，$x_3\notin F$，且 $|x_3|>M$（$\because$ 阶码 $3>U=2$）故发生上溢。

例 1.3.2 设 $F(2,3,-1,2)$ 中有二数 $a=0.110\times2^{0}$，$b=0.111\times2^{0}$，求 $c=a+b$

解 $c=a+b=(0.110\times2^{0})+(0.111\times2^{0})=0.1101\times2^{1}$

此时 $c\notin F(2,3,-1,2)$，故取其 F 中的近似数

$$\mathrm{fl}(c)=0.110\times2^{1}$$

§1.4 误差的基本概念

前面已经提到过误差，本节对误差的基本概念将作进一步的讨论。因为计算方法中的数值，特别是计算出的数值结果一般都是准确值的近似值。准确值与近似值之间存在的差异就是所谓的**误差**。人们总希望所得近似计算结果有令人满意的精确度，因此，估计误差是计算方法的最基本的一项任务。

1.4.1 误差的来源

产生误差的原因是多方面的，主要地可归结如下四种来源：

(1)模型误差　从实际问题提出数学模型时，为使数学模型尽可能的简单，以便于分析和计算，往往要忽略许多次要的因素，这样，即使数学模型能准确求解，也与原来实际问题的真解有所差异，这样产生的误差，称为**模型误差**或**描述误差**。

(2)观测误差　在求解的问题中，一般都含有从观测或实验得到的数据，由于仪器本身的精度有限或某些客观因素会带来一定的误差，这种误差称为**观测误差**或**数据误差**。

(3)截断误差　某些数学问题难以直接求解，往往要通过近似替代，简化成较易求解的问题。例如常用收敛的无穷级数的前 n 项来代替无穷级数进行计算，这种简化引起的误差称为**截断误差**或**方法误差**。

(4)舍入误差　由第§1.3节的讨论得知，计算机的数系 $F(\beta,t,L,U)$ 内只包含有限的一部分离散的实数，而在实际计算所得的实数不一定属于 F，必须按一定舍入规则，近似表示成机器中的浮点数，这种近似引入的误差，称为**舍入误差**或**计算误差**。

上述四种误差都会不同程度的影响计算结果的准确性。但模型误差和观测误差不是计算数学工作者所能单独解决的，需要会同有关学科的科学工作者协同研究。因此，在计算方法教程中，我们主要研究截断误差和舍入误差。本节将进一步讨论舍入误差的一些内容。

1.4.2 误差与有效数字

我们先给出绝对误差与相对误差的定义。

定义 1.4.1　设 x^* 为准确值，x 是 x^* 的一个近似值，$e=x^*-x$ 称为近似值 x 的**误差**，通常也叫**绝对误差**。

由于一般无法确定准确值 x^*，故误差 e 的精确值也难以求出，但按具体情形，往往可以估计出 e 的绝对值不超过某个正数 ε，即

$$|e|=|x^*-x|\leqslant\varepsilon \tag{1.4.1}$$