计算方法

黄明游　梁振珊　编

吉林大学出版社

计 算 方 法

黄明游　梁振珊　编

吉林大学出版社

计 算 方 法

黄明游　梁振珊　编

责任编辑、责任校对：赵洪波　　封面设计：孙　泓

吉林大学出版社出版　　吉林省新华书店发行
（长春市东中华路 29 号）　　长春大学印刷厂印刷

开本：850×1168 毫米　1/32　　1994 年 6 月第 1 版
印张：7.75　　1994 年 6 月第 1 次印刷
字数：189 千字　　印数：1—1100 册

ISBN 7-5601-1581-0/O・182　　定价：4.50 元

前　言

本书是适应为非计算数学的有关专业开设〈计算方法〉课程的需要而编写的。计算方法的内容极为广泛，目前计算数学专业一般是分三门课程(数值逼近、数值代数、微分方程数值解法)讲授其内容，需要花费近200个学时。然而，对于其他有关专业，上述作法是既不可能，同时也并非是可取的。编写这本教材的目的，是便于在一个学期(约用68个学时)内讲授科学与工程计算中遇到的一些基本问题的计算方法，介绍计算方法的原理和选讲一些典型的、便于在计算机上应用的数值计算方法。

这本书的内容大纲曾在理科力学专业教材会议(1985，兰州)上与北京大学、复旦大学、兰州大学和中山大学的同事们进行过讨论，他们提供了宝贵的修改意见。根据这个大纲，经本书两名编者的合作，于1986年编写出一本'实用计算方法'讲义。该讲义曾作为吉林大学数学系力学专业、应用数学专业和计算机科学系讲授《计算方法》课的教材使用过多遍。本书就是在这个讲义和历年来该课教学实践的基础上，经过重新修改、补充之后写成的。其中第一至第四章由梁振珊同志编写，第五、六两章由黄明游同志编写，大纲的修订和全书内容的校核是黄明游同志完成的。

吉林大学计算数学教研室常玉堂、于加艾、马富明等同志曾先后利用上述教材讲授过计算方法课，他们提供过许多宝贵的修改意见，在此向这些同事表示亲切的谢意。特别地，这里要向参加本教材大纲讨论并给予指导的外校和本校的老师们表示衷心的感谢。

由于水平和经验的原因，本书在取材、衔接、表述等方面一定还存在许多不足乃至某些错误，恳请批评、指正。

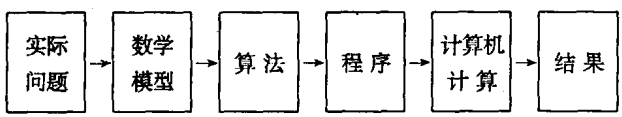
编　者

1994年4月

目　录

绪　言

用电子计算机进行科学计算，解决生产实际问题，其过程大致如下：



我们所说的计算方法，是指在数字电子计算机上使用的计算方法．计算机所执行的算法由算术运算和逻辑运算组成．把数学模型归结为计算机能执行的有效算法，便是计算方法课程讨论的课题．为此，初步了解一下数的计算机表示和运算是必要的．

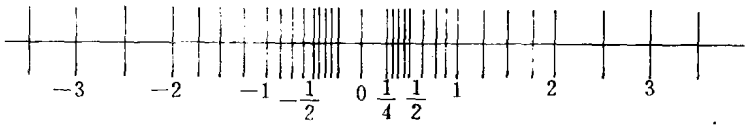
数的浮点表示和浮点四则运算

目前，数字电子计算机大多使用二进制浮点计算系统．在这种二进制浮点计算机上，每个实数 x 具有如下形式：

$$x = \pm 0.d_1d_2\cdots d_k \times 2^p. \tag{1}$$

这里，$d_1=1$，d_i 为 0 或 1，$i=2$，…，k．整数 p 满足 $-m \leqslant p \leqslant M$．零的浮点表示可能是不同的，通常用 $0=\pm 0.00\cdots 0\times 2^{-m}$ 表示．正整数 k，m，M 随计算机而定．因此，一部给定的二进制浮点计算机，只能表示所有形如(1)的有限数集 $S=S(k,m,M)$．这是实数轴上的不等距的有限点集．为了说明问题，我们考虑一个特殊的计算机，它的 $k=3$，$m=1$，$M=2$，那么这个计算机所能表示的浮点数的集合 $S(3,1,2)$ 只有如下图的 33 个

点：



一个实数 x，在计算机上只能按一定规则用 S 中最接近 x 的数来近似表示. 这就带来了数的表示误差. 在我们这个小计算机上，$|x|>3.5$ 就溢出了（上溢出），$|x|<0.25$ 就是机器零（下溢出）.

计算机只能对浮点数集 S 中的数做加、减、乘、除四则运算. 由于用二进制数作四则运算我们不习惯，所以这里不妨用10进制的4位浮点数的四则运算来说明，因为它们的情况是一样的. 设 S_0 是所有形如

$$\pm\ 0.d_1d_2d_3d_4 \times 10^p$$

的4位10进制浮点数的集合，其中 $1\leqslant d_1\leqslant 9$，$0\leqslant d_i\leqslant 9$，$i=2,3,4$，整数 p 满足 $-9\leqslant p\leqslant 10$. 下面举例说明 S_0 上的算术运算，“→”表示在运算器上的运算过程和结果.

1) $0.1984\times10^4+0.2008\times10^2$

$\rightarrow 0.1984\times10^4+0.0020\times10^4$

$\rightarrow 0.2004\times10^4$

2) $0.1984\times10^4+0.9876\times10^{-1}$

$\rightarrow 0.01984\times10^4+0.0000\times10^4$

$\rightarrow 0.1984\times10^4$

3) $0.1984\times10^{-6}-0.1976\times10^{-6}$

$\rightarrow 0.0008\times10^{-6}$

$\rightarrow 0.8000\times10^{-9}$

4) $(0.5678\times10^3)\times(0.6789\times10^{-5})$

$\rightarrow (0.5678\times0.6789)\times10^{-2}$

$\rightarrow 0.38547942\times10^{-2}$

$\rightarrow 0.3855\times10^{-2}$

5) $(0.5678\times10^{4})\div(0.4567\times10^{-5})$

$\rightarrow 0.5678\times10^{9}\div0.4567$

$\rightarrow 0.05678000\times10^{10}\div0.4567$

$\rightarrow 0.12432669\times10^{10}$

$\rightarrow 0.1243\times10^{10}$

由这些例子可以看出，计算机上的算术运算不可避免地带来舍入误差. 尤其应当注意，尽量避免象 2)、3)、5) 那样的情况发生：绝对值大小悬殊的两个数相加、减；相近的数相减；相对被除数来说绝对值很小的数作除数. 这些是数值计算中的基本原则. 在 5) 中当除数为 0.4567×10^{-6}时，商 0.1243×10^{11}已不属于浮点数集 S_0，这就是所谓“溢出”.

研究计算方法的必要性

了解了数的计算机表示和计算机的算术运算，就不难理解为什么要讨论计算方法. 这里，我们通过几个简单的例题，来进一步说明这个问题.

例 1 在使用上述浮点数集 S_0 的 4 位 10 进制浮点计算机上，解一元二次方程

$$x^2-18x+1=0.$$

按求根公式，两个根为

$$x_1=9+\sqrt{80},\quad x_2=9-\sqrt{80}.$$

在我们的计算机上，$\sqrt{80}=0.8944\times10^{1}$. 于是得

$$x_1=0.1794\times10^{2},\quad x_2=0.5600\times10^{-1}.$$

但是换一种办法计算 x_2，由 $x_1x_2=1$ 得

$$x_2=\frac{1}{9+\sqrt{80}}=0.5574\times10^{-1}.$$

它的误差不超过 $0.0001\times10^{-1}\times0.5=0.000005$（末位有效数字的一半）；而前面按求根公式算得的 x_2，其误差不超过 $0.01\times10^{-1}\times0.5=0.0005$. 显然，在数学上公式 $x_2=9-\sqrt{80}$ 和 $x_2=1/(9+\sqrt{80})$ 是等价的，但在计算机上它们是不同的. 从计算方法的观点看来，后者远优于前者，前者的缺欠在于两个相近数相减.

例 2 考虑对于给定 x 计算多项式

$$P_n(x)=a_0x^n+a_1x^{n-1}+\cdots+a_{n-1}x+a_n \tag{2}$$

的值. 显然，它等价于

$$P_n(x)=(\cdots((a_0x+a_1)x+a_2)x+\cdots+a_{n-1})x+a_n \tag{3}$$

但从计算方法的观点看来，两者却大不相同：公式（2）需先算出 $x^2, x^3, \cdots, x^n$，共需 $n-1$ 次乘法；还要保存下来，需增加 $n-1$ 个存贮单元；然后再相乘（n 个乘法）、相加（n 个加法），共需 $2n-1$ 次乘法，n 次加法. 公式（3）只需 n 个乘法，n 个加法，不用增加存贮单元. 公式（2）和（3）的差别远不止这些. 实际上，在我们上述的计算机上，$x=10$ 时按（2）就不能计算，因为 10^{10} 已经溢出.

例 3 考虑解线代数方程组

$$Ax=b. \tag{4}$$

设系数矩阵 A 是 $n\times n$ 方阵，其行列式 $D=\det(A)\neq0$. 按 Gramer 法则，(4) 的解为

$$x_i=D_i/D,\quad i=1,2,\cdots,n.$$

共需计算 $n+1$ 个 n 阶行列式. 用 Laplace 展开计算 n 阶行列式，需要

$$n!+\frac{n!}{2!}+\frac{n!}{3!}+\cdots+\frac{n!}{(n-1)!}$$

$$=n!\left(1+\frac{1}{2!}+\frac{1}{3!}+\cdots+\frac{1}{(n-1)!}\right)$$

个乘法，姑且算作 $n!$ 个乘法. 这样，不计加法，用 Gramer 法

则解 n 阶方程组，用 Laplace 展开计算行列式，共需 $(n+1)n!=(n+1)!$ 以上的乘法. 对于一个 20 阶的方程组，就需要 $21!\doteq 5.11\times10^{19}$ 以上的乘法. 设用每秒可作百万次乘法的计算机，一年可作 $365\times24\times3600\times10^6\doteq3.15\times10^{13}$ 次乘法. 所以，在每秒作百万次乘法的计算机上，用 Gramer 法则解 20 阶的线代数方程组，需要的时间在 $(5.11\times10^{19})\div(3.15\times10^{13})=1.62\times10^6\doteq162$ 万年以上!

例 4 考虑计算积分

$$I_n=\int_0^1 x^n e^{x-1}dx. \tag{5}$$

它满足递推关系

$$I_n=1-nI_{n-1}. \tag{6}$$

我们首先算出 I_0 的近似值 $\bar{I}_0=0.6321$. 利用递推关系 (6) 依次算得：

$\bar{I}_0$	$\bar{I}_1$	$\bar{I}_2$	$\bar{I}_3$	$\bar{I}_4$	$\bar{I}_5$	$\bar{I}_6$	$\bar{I}_7$
0.6321	0.3680	0.2640	0.2080	0.1680	0.1600	0.0400	0.7200

显然 I_7 的递推计算结果 $\bar{I}_7$ 是错误的. 因为

$$I_n<e^{-1}\left(\max_{0\leqslant x\leqslant1}e^x\right)\int_0^1 x^n dx=\frac{1}{n+1},$$

所以

$$I_7<\frac{1}{8}=0.1250.$$

实际上，序列 $\{I_n\}$ 是一个下降序列. 为什么会产生这样面目全非的错误结果呢？这是由于在计算 $\bar{I}_0$ 时的原始误差（表示误差和舍入误差）以及计算 $\bar{I}_1$ 的误差，在按(6)递推时，逐次乘以因子 $2,3,\cdots,7$，使误差急剧地增长和积累，从而产生错误的没有意义的结果. 然而，如果将(6)改写成

$$I_{n-1} = (1 - I_n)/n, \qquad (7)$$

并先计算出 I_7 的近似值 $\bar{I}_7 = 0.1124$，从 $\bar{I}_7$ 出发按（7）递推算得：

$\bar{I}_7$	$\bar{I}_6$	$\bar{I}_5$	$\bar{I}_4$	$\bar{I}_3$	$\bar{I}_2$	$\bar{I}_1$	$\bar{I}_0$
0.1124	0.1269	0.1455	0.1708	0.2073	0.2643	0.3680	0.6320
0.1124	0.1268	0.1456	0.1709	0.2073	0.2642	0.3679	0.6321

表中末行是 I_n 的精确值的舍入结果. 可见按（7）递推计算出的 $\bar{I}_n$（$n=6, 5, \cdots, 0$）是令人满意的. 有趣的是，即使从 $\bar{I}_7 = 0$ 开始按（7）递推，也可得到 $\bar{I}_0 = 0.6320$. 这是由于原始误差每步依次乘以 $\frac{1}{7}$，$\frac{1}{6}$，…，1，从而被大大地缩小了.

通过上述例题可以看出，即使有了数学模型，即使数学上已经有了完善的结果，仍然存在能不能在计算机上实现和如何实现的问题. 所以我们必须研究计算方法，提供计算机上的有效算法.

计算方法的主要内容

对于给定的数学问题，常常可以提出各式各样的算法. 怎样评价这些算法的优劣呢？这要具体问题具体分析. 一般说来，从计算方法的观点考虑，自然提出以下要求：

第一，计算结果可靠. 即精确度高、误差小. 通常原始数据是通过仪器观测得到的，含有**观测误差**；数据送入计算机，产生计算机的**表示误差**；计算机进行算术运算，产生**舍入误差**；计算机只能作有限的算术四则运算，微商、积分、微分方程等只能通过算术运算给出近似公式. 这就产生所谓**截断误差**或**余项**.

观测误差和表示误差取决于仪器的精度和计算机，这里无须讨论. 浮点运算的舍入误差分析，比较复杂，超出我们课程的范围. 截断误差或余项，在我们讨论的算法中，一般都要给出.

从上面例 4 可以看到，计算结果的可靠性不仅和误差有关，而且和算法有关. 像递推公式 (6) 那样的算法，将原始误差不断扩大 (即使按公式精确计算也是如此)，这样的算法说是**不稳定的**. 像递推公式 (7) 那样的算法，使原始的误差得到控制，不再增长，这样的算法称作**稳定的**. 为了保证计算结果可靠，我们使用的算法必须是稳定的.

第二，算法简单，运算量少.

第三，存贮量小. 计算机的存贮设备有内存和外存，调用外存比使用内存慢得多，使用内存中的数据才能实现快速计算. 而内存总是有限的 (外存几乎可以看成是无限的)，所以我们提出的算法，占用内存单元应当尽可能少.

在电子计算机飞速发展的今天，计算机的应用已经深入到科研、生产和生活的各个领域，计算机上使用的计算方法已浩如烟海. 本书只限于介绍科学计算方面的最基本的计算方法. 主要内容有：线代数方程组的数值解法和矩阵特征值、特征向量的计算，非线性方程和非线性方程组的解法，最优化问题的计算方法，函数插值和数值积分，常微分方程初值问题的数值解法和偏微分方程的数值解法.