高等学校试用教材

计算数学简明教程

何旭初 苏煜城 包雪松 编

高等教育出版社

高等学校试用教材

计算数学简明教程

何旭初 苏煜城 包雪松 编

高等教育出版社

本书原由人民教育出版社出版。1983年3月9日，上级同意恢复"高等教育出版社"；本书今后改用高等教育出版社名义继续印行。

高等学校试用教材

计算数学简明教程

何旭初 苏煜城 包雪松 编

*

高等教育出版社出版

新华书店上海发行所发行

青浦任屯印刷厂印装

*

开本 850×1168 1/32 印张 12 2/16 字数 292,000

1980年4月第1版 1984年2月第4次印刷

印数 35,501—41,000

书号 13010·0457 定价 1.10 元

说　明

这本教材包含了数值逼近方法(插值法、数值积分法和Chebyshev多项式及其应用),数值代数(线性和非线性方程组的数值解法,特征值的计算方法等),以及常微分方程和偏微分方程数值解法等计算数学中常用的数值方法和建立数值方法的基本原理.目的在为读者对各有关分支进行深入学习和研究提供一个初步的基础.在线性代数方程组的解法中我们着重介绍直接方法,因为迭代法主要用于解大型稀疏矩阵方程组,它是一个专门的分支,所以只对迭代法的基本原理作了简单的介绍.考虑到共轭斜量法的重要性,对此法也作了初步介绍.关于特征值问题,由于教学时间和教材篇幅的限制,我们只重点介绍了计算实对称矩阵的QR方法.解非线性方程组的数值方法是近年来非常活跃的一个分支,教材中也只对Newton法的基本原理作了初步介绍.偏微分方程是一个相当广阔的领域,内容十分丰富,在最后一章中作了扼要的介绍.

本书可供理、工、师范院校有关专业作为计算方法课的教材.教完全书约需一个学年,只开设半年计算方法课的专业,可以只讲授其中的一,二,三,四,五,六及十一诸章,根据具体情况并可酌量予以增减.

本书由厦门大学林坚冰同志主审,参加审稿的还有厦门大学、武汉大学、南京师范学院、曲阜师范学院、南京大学的同志.他们提出了很宝贵的意见,在此谨致谢意.

近年来计算数学这个学科发展很快,各专业的要求也未尽相

同，且编者水平有限，选材不当及错漏之处在所难免．希望广大读者提出宝贵意见，以便修订时作为参考．

编　者

一九七九年十二月于南京大学数学系

目　录

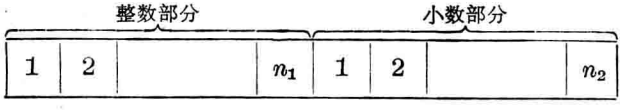
第一章 预 篇

§1 引 言

微积分学的基础是实数系，而严格的极限理论正是在实数理论的基础上建立起来的. 与此相仿，在研究适合于现代的数字电子计算机使用的数值方法的时候，我们也必须考虑计算机中的实数系，即机器数的情况. 微积分学中的实数系是一个连续统，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. 在一个数字电子计算机中所存放的机器数的全体，则是一个有限的离散集合，其分布也是不均匀的，并且，在机器数的运算中，一般的运算律也并非总是成立的. 我们所研究的数值方法在计算机上实现的时候，就是用这个残缺不全的数系在不完全受一般计算律的控制下进行运算的. 为了使我们的数值方法利用计算机进行计算时所得到的结果符合我们的需要，就应该对机器数及其运算的情形加以了解.

§2 在数字电子计算机中数的表示

2.1 数的定点表示 在数字电子计算机中，一个数是在有限个位置上用物理量产生有限个不同的信息来表示的. 一种方法是：用依序排列的 n 个单元来表示一个数，并固定小数点在 n_1 位之后，后边 n_2 个单元为小数部分，$n=n_1+n_2$，每个单元可以产生十种不同的物理信息，于是，一个 n 位十进制数就可以表示成



例如，$n=8$, $n_1=4$, $n_2=4$, 则数 205.7019 和 4325.0128 就可以分别表示为

0	2	0	5	7	0	1	9

4	3	2	5	0	1	2	8

这种固定小数点位置的表示方法称为数的*定点表示*，每一种表示，叫做一个*定点数*，称 n 为*字长*. 一般常取 $n_1=n$ 或 $n_1=0$.

关于数的符号，在计算机中用另外的信息来表示.

2.2 数的浮点表示 在科学计算中常常把数，例如

$$0.0031207,\quad 0.091650,\quad 293.7048$$

等，分别表示成

$$0.31207\times10^{-2},\quad 0.91650\times10^{-1}$$

和

$$0.2937048\times10^{3}.$$

这样一来，一个数的数量级就一目了然. 在这种表示方法中，小数点的位置决定于后边那个 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*. 数的浮点表示由两部分组成. 如前述各数中的 0.31207, 0.91650 和 0.2937048, 称为浮点表示的*尾数部*，后边的 10^{-2}, 10^{-1} 和 10^{3}, 称为*定位部*，是用来确定小数点的位置的. 而定位部则由阶码 -2, -1 和 3 来确定. 因此，一个浮点数包含尾数和阶码两部分. 前述诸数在程序中常写成

$$0.31207_{10}-2,\quad 0.91650_{10}-1,\quad 0.2937048_{10}+3.$$

在数字电子计算机中，浮点数尾数的位数是固定的，设为 t，称为计算机的字长. 阶码记作 e，它有固定的上、下限，例如

$$-999<e<999.$$

有一些计算机，为了提高计算结果的精度，可以用双倍、三倍

或多倍字长来表示一个数.

浮点数的阶码，其范围可以用表示阶码的位数来确定，这个位数仍记为 e. 例如，$e=3$，表示阶码用三位数来表示，$e=2$ 表示阶码用二位数来表示. 以后如不特别申明，e 总是表示阶码的位数.

浮点表示的规格化 在前述意义下，一个数可以有不同的浮点表示. 例如，在 $t=4$，$e=2$ 时，8370 可以表示成

$$0.8370_{10}04,$$

也可以表示成

$$0.0837_{10}05.$$

为了避免这种情况发生，我们规定非零的机器数在浮点表示中尾数的第一位数字非零. 在这种规定下的浮点表示，称为规格化的浮点数.

二进制浮点数 在实数的二进制表示中，每一位数字不是0就是1，因而用两种不同的物理信息就可以表示出来. 所以，现在的数字计算机中常使用二进制表示实数. 例如，

$$x=1\cdot 2^{-1}+0\cdot 2^{-2}+1\cdot 2^{-3}+0\cdot 2^{-4}+1\cdot 2^{-5},$$

则 x 的二进制表示为

$$(x)_2=0.10101.$$

又如 $$18.5=1\cdot 2^4+0\cdot 2^3+0\cdot 2^2+1\cdot 2^1+0\cdot 2^0+1\cdot 2^{-1},$$

所以，18.5 的二进制表示为

$$(18.5)_2=10010.1.$$

实数的二进制浮点表示和十进制的情形一样，也是由尾数和阶码两个部分所组成. 为了规格化，也规定非零机器数尾数的第一位数字非零. 例如，

$$0.1010101_2101,\quad 0.10001001_2100,$$

$$0.11101_2-010,\quad 0.110011_2-001$$

等，都是规格化的二进制浮点数(字长不同).

综上所述，实数的浮点表示形式如下：

十进制　$x=a\cdot 10^b$，$|a|<1$，b 为整数，

二进制　$x=a\cdot 2^b$，$|a|<1$，b 为整数.

为了规格化，我们规定 a 的第一位数字非零.

2.3　机器数系　上面介绍的数的浮点表示方法为现代的数字电子计算机所通用，是我们研究数值方法的基础. 把计算机中浮点数的全体组成的集合记作 F，则 F 中的浮点数具有以下的形式：

$$x=\pm\left(\frac{d_1}{\beta}+\frac{d_2}{\beta^2}+\cdots+\frac{d_t}{\beta^t}\right)\cdot\beta^e, \tag{1}$$

其中的 $d_1, d_2, \cdots, d_t$ 为整数，它们满足关系

$$0\leqslant d_i\leqslant\beta-1,\quad i=1, \cdots, t,$$

β 为浮点数的基底，一般

$$\beta=10,\quad 2\quad 或\quad 16.$$

自然数 t 为计算机的字长，e 为浮点表示的阶码，它有固定的下限 L 和上限 U，

$$L\leqslant e\leqslant U. \tag{2}$$

t，L 和 U 随计算机而异.

若对 $x\neq 0$ 规定(1)中的 $d_1\neq 0$，则 F 为规格化的浮点数系. 不难证明，F 中共有

$$2(\beta-1)\beta^{t-1}(U-L+1)+1 \tag{3}$$

个浮点数. 例如，若 $\beta=2$，$t=3$，$L=-1$，$U=2$，则相应的浮点数系中共有 33 个浮点数.

§3　浮点运算和舍入误差

3.1　数据的机器数近似　浮点数系 F 既然是一个离散的有限集合，在利用计算机进行计算时，初始数据和中间结果都可能不

在 F 中，于是便发生用 F 中的数来近似地表示相应数据的问题. 设实数 x 不属于 F，我们要用 F 中的一个浮点数作为 x 的近似，记这个浮点数为 $fl(x)$，它应有最好的逼近性质，即

$$|x-fl(x)|=\min_{g\in F}|x-g|. \tag{4}$$

如果 x 的绝对值大于 F 中的最大正数，或小于 F 中的最小正数，就会发生上溢或下溢现象. 在以后讨论运算时，对给定的数据以及中间结果，它们的浮点数近似都在 F 中而不发生溢出.

对给定的数 x，为了说明确定 $fl(x)$ 的方法，今举例如下.

设 $t=4$，$e=2$，$\beta=10$，则

$$fl(0.20456_{10}12)=0.2046_{10}12,$$

$$fl(0.15732_{10}02)=0.1573_{10}02.$$

一般说来，若 x 的尾数为 a，且

$$|a|=0.\alpha_1\alpha_2\cdots\alpha_t\alpha_{t+1}\cdots, \qquad 0\leqslant\alpha_i\leqslant 9,\ \alpha_1>0, \tag{5}$$

$$a'=\begin{cases}0.\alpha_1\alpha_2\cdots\alpha_t, & \text{若 } 0\leqslant\alpha_{t+1}\leqslant 4,\\ 0.\alpha_1\alpha_2\cdots\alpha_t+10^{-t}, & \text{若 } \alpha_{t+1}\geqslant 5.\end{cases} \tag{6}$$

若 $x=a\cdot 10^b$，则令

$$fl(x)=\operatorname{sign}(x)a'\cdot 10^b, \tag{7}$$

其中

$$\operatorname{sign}(x)=\begin{cases}+1, & \text{若 } x>0,\\ -1, & \text{若 } x<0.\end{cases}$$

显然可知，用上述方法确定的 $fl(x)$ 作为 x 的浮点数近似，相对误差满足不等式

$$\left|\frac{fl(x)-x}{x}\right|\leqslant\frac{5\cdot 10^{-(t+1)}}{|a|}\leqslant 5\cdot 10^{-t}. \tag{8}$$

因为 $|a|\geqslant 10^{-1}$，于是，令 $\text{eps}=5\cdot 10^{-t}$，则得关系式

$$fl(x)=x(1+\varepsilon),\quad |\varepsilon|\leqslant\text{eps}. \tag{9}$$

我们称 $\text{eps}=5\cdot 10^{-t}$ 为计算机的精度，并称上述确定浮点数 $fl(x)$ 的方法为“舍入”.

对于二进制系统，舍入规则和前面相仿．设 x 是一个二进制数，$x=a\cdot 2^b$，其中

$$2^{-1}\leqslant|a|<1, \tag{10}$$

且

$$|a|=0.\alpha_1\alpha_2\cdots\alpha_t\alpha_{t+1}\cdots,\ \alpha_i=0\ 或\ 1,\quad \alpha_1=1, \tag{11}$$

令

$$a'=\begin{cases}0.\alpha_1\cdots\alpha_t, & 若\ \alpha_{t+1}=0,\\ 0.\alpha_1\cdots\alpha_t+2^{-t}, & 若\ \alpha_{t+1}=1.\end{cases} \tag{12}$$

最后令

$$fl(x)=\mathrm{sign}(x)\ a'\cdot 2^b. \tag{13}$$

对二进制系统，计算机精度

$$\mathrm{eps}=2^{-t},$$

这时仍有

$$fl(x)=x(1+\varepsilon),\quad |\varepsilon|\leqslant\mathrm{eps}. \tag{14}$$

公式(9)(或(14))可以写成

$$fl(x)-x=x\varepsilon, \tag{15}$$

或

$$\frac{fl(x)-x}{x}=\varepsilon,\ 若\ x\neq 0. \tag{16}$$

公式(15)中的 $x\varepsilon$ 为用浮点数 $fl(x)$ 表示 x 时的绝对误差，(16)中的 ε 为相应的相对误差．

绝对误差与 x 所表示的物理量的单位有关，而相对误差则不依赖于表示物理量的单位．

此外，不能根据绝对误差的大小(范围)来确定近似程度的好坏．例如，有两个温度计，其一测量1000℃时的绝对误差为 ±5℃，而另一个测量 100℃ 时的绝对误差为 ±1℃，虽然后者绝对误差的数值较小，但第一种温度计要更为精确些．实际上，第一种温度计的相对误差为 ±0.5%，而第二种的相对误差为 ±1%，它是前者的两倍．

前面所讲的舍入规则，就是通常所说的“四舍五入”．有的计算机却只“舍”而不“入”．这时产生的相对误差的界，即计算机的

精度为

$$\text{eps}=\begin{cases}10^{1-t}, & \text{十进制},\\ 2^{1-t}, & \text{二进制}.\end{cases} \tag{17}$$

3.2 浮点数的算术运算 现在我们来考虑计算机中浮点数的运算. 设 x, y 都是规格化的浮点数, 即 x, $y\in F$. 它们的算术运算

$$x+y,\quad x-y,\quad x\times y,\quad x/y$$

的精确结果则不一定是 F 中的浮点数. 例如, 若 $t=4$, $\beta=10$,

$$x=0.3127_{10}-6,\quad y=0.4153_{10}-4,$$

则

$$\begin{aligned}x+y&=0.003127_{10}-4+0.4153_{10}-4\\&=0.418427_{10}-4.\end{aligned}$$

所以, $x+y$ 不属于 F.

关于乘积 $x\times y$, 其尾数一般为 $2t$ 位, 至少为 $2t-1$ 位, 所以也不是 F 中的数.

在计算机中进行浮点运算时, 计算机自动把运算结果用 F 中的浮点数表示出来. 记算术运算的结果为

$$fl(x\pm y),\quad fl(x\times y),\quad fl(x/y).$$

这些运算结果怎样用机器数来表示, 随计算机的功能而异. 我们假定计算机具有双精度累加寄存器, 即在运算时先保留 $2t$ 位, 最后再把第 $t+1$ 位"四舍五入". 这时前例中的运算结果为

$$fl(x+y)=0.4184_{10}-4.$$

由于这个过程相当于用 F 中的浮点数近似表示一般的数, 所以, 浮点运算和精确算术运算之间的关系为

$$\begin{aligned}fl(x+y)&=(x+y)(1+\varepsilon_1),\\fl(x-y)&=(x-y)(1+\varepsilon_2),\\fl(x\times y)&=(x\times y)(1+\varepsilon_3),\\fl(x/y)&=(x/y)(1+\varepsilon_4),\end{aligned} \tag{18}$$

其中

$$|\varepsilon_i| \leqslant \text{eps}, \quad i=1, 2, 3, 4. \tag{19}$$

公式(18)的右端表示算术运算的精确结果.

3.3 舍入误差在算术运算中的传播 在前述浮点运算的情况下,通常的运算律一般就不复成立. 例如,设 $t=8$,

$$x=0.23371258_{10}-4,$$

$$y=0.33678429_{10}2,$$

$$z=-0.33677811_{10}2,$$

则

$$fl(x+(y+z))=fl(0.23371258_{10}-4$$
$$+0.61800000_{10}-3)=0.64137126_{10}-3.$$

但

$$fl((x+y)+z)=fl(0.33678452_{10}2-0.33677811_{10}2)$$
$$=0.64100000_{10}-3.$$

精确的结果为

$$x+y+z=0.641371258_{10}-3.$$

现在我们来分析误差在以不同方式进行运算

$$x+y+z$$

时传播的情形.

1) $fl((x+y)+z)$.

根据公式(18),我们有

$$fl((x+y)+z)=fl((x+y)(1+\varepsilon_1)+z)$$
$$=[(x+y)(1+\varepsilon_1)+z](1+\varepsilon_2)$$
$$=(x+y+z)\left[1+\frac{x+y}{x+y+z}\varepsilon_1(1+\varepsilon_2)+\varepsilon_2\right],$$

其中 $|\varepsilon_i| \leqslant \text{eps}$, $i=1, 2$. 略去高阶项 $\varepsilon_1\varepsilon_2$,便得到近似等式

$$fl((x+y)+z)=(x+y+z)\left[1+\frac{x+y}{x+y+z}\varepsilon_1+\varepsilon_2\right]$$
$$=(x+y+z)(1+\varepsilon), \tag{20}$$

其中的相对误差

$$\varepsilon=\frac{x+y}{x+y+z}\varepsilon_1+\varepsilon_2. \tag{21}$$

2) $fl(x+(y+z))$.

仿前，

$$\begin{aligned} fl(x+(y+z)) &= (x+y+z)(1+\varepsilon'),\\ \varepsilon' &= \frac{y+z}{x+y+z}\varepsilon_1'+\varepsilon_2', \end{aligned} \tag{22}$$

其中$|\varepsilon_i'|\leqslant \text{eps}$, $i=1, 2$.

这两种不同方式运算结果的精度，视$|\varepsilon|$和$|\varepsilon'|$的大小而定. 由于ε_i和ε_i'具有随机性，$|\varepsilon|$和$|\varepsilon'|$的大小一般决定于因子

$$\frac{x+y}{x+y+z} \quad 和 \quad \frac{y+z}{x+y+z}.$$

若$|x+y|<|y+z|$，则前一种较好. 反之，若$|y+z|<|x+y|$，则后者较好. 这就是说，绝对值较小者先加比较有利. 在前例中

$$\frac{x+y}{x+y+z}=\frac{0.33\cdots_{10}2}{0.64\cdots_{10}-3}\approx\frac{1}{2}\cdot 10^5,$$

$$\frac{y+z}{x+y+z}=\frac{0.618\cdots_{10}-3}{0.64\cdots_{10}-3}\approx 0.97.$$

所以，$fl(x+(y+z))$比$fl((x+y)+z)$要精确些.

3.4 数据误差对计算结果的影响(条件问题) 在前一小节中所述参与运算的数x, y等，都是规格化的浮点数，它们本身没有误差，计算结果中的误差是由于计算机在计算过程中的舍入而产生的. 实际上，初始数据可能是由计算得到的，它本身带有舍入误差. 此外，初始数据也可以是由观测而得到的，由于观测手段的限制，得到的数据也必然带有一定程度的误差，这种误差称为观测误差. 在把观测数据送进计算机中，观测误差和由于用浮点数近似初始数据而产生的舍入误差相比，一般说，前者居于主导地位.