第 1 章　绪　　论

1.1　引　　言

计算方法又名数值分析, 是一门随着计算机的发展而形成的新兴学科, 是数学、计算机科学与其他学科交叉的产物. 它是专门研究如何利用计算机有效地求解各类计算问题的有关方法和理论的一门学科. 由于其所涉及的计算问题主要来源于科学研究和工程设计, 因此近年来人们常常称这门科学为科学计算.

对于一些复杂的科学与工程问题, 理论分析往往无能为力, 而实验又无法进行, 社会的发展和科学的进步呼唤着新的科学研究方法的出现. 20 世纪 40 年代, 电子计算机的发明为计算成为第三种科学研究的手段提供了可能. 半个世纪来, 计算机的飞速发展已把计算推向人类科学活动的前沿, 它作为科学研究方法的地位不断地上升. 今天, 实验、理论分析和计算 “三足鼎立”, 已成为当今科学活动的主要方式. 在自然科学和工程技术的发展过程中, 先后产生了计算数学、计算力学、计算物理、计算化学、计算材料学、计算生物学等一系列计算性的分支学科, 统称为计算科学. 今天, 计算在科学研究和工程设计中几乎无处不在, 对科技的发展起到了举足轻重的作用.

计算科学是通过计算的手段来解决实际问题的一门科学, 其处理问题的过程主要有 3 个环节：

- 建立数学模型;
- 设计计算方案 (简称算法), 编制程序, 上机运行, 展示数值结果;
- 将数值结果与理论分析和实验的结果相结合给出实际问题的

答案, 或提出对模型的修正方案.

上述第二个环节正是计算方法这门学科所研究的主要内容, 其核心是算法的设计. 不同的学科, 不同的工程应用会提出不同的问题, 其中多数都可归结为若干典型的数学模型. 给出这些典型问题的数值求解方法, 也就为大多数科学与工程计算问题的解决提供了可能性. 因此, 本课程将着重介绍几类典型问题的数值解法.

大家知道, 电子计算机的运算速度越来越快, 可以承担大运算量的工作. 这是否意味着计算机上的算法可以随意选择呢? 事实上, 对于一个具体的计算问题, 所使用算法的优劣, 不仅影响计算结果的精确程度, 而且有的甚至关系到计算的成败. 请看下例: 假设用著名的 Cramer 法则去求解一个 25 个未知数的线性方程组 (这需要计算 26 个 25 阶的行列式), 再假定是按照行列式的定义来计算行列式的值, 则完成这一计算任务需要的乘法次数约为 26!. 如忽略存取数和加减运算等计算机操作所需要的时间, 用每秒可做万亿次乘法的计算机来完成这一计算任务, 所需的时间大约是一千三百多万年. 然而, 改用消去法, 则可在不到 1 秒的时间内即可完成求解上百阶线性方程组的计算任务.

此外, 许多科学与工程计算问题都有如下特点: 高维数、多尺度、非线性、不适定、长时间、奇异性、复杂区域、高度病态, 不仅计算规模大, 而且要求精度高. 其计算困难也有各种不同的表现, 如计算规模大, 大得难以承受或失去时效; 计算不稳定, 数值结果不可信; 包含奇异性, 计算可能非正常终止等. 这样的问题, 如果不进行深入细致的算法研究, 即使是现在最强大的计算机也无能为力. 人类的计算能力既依赖于计算机的性能, 也取决于计算方法的效能. 计算方法的发展对于提高人类计算能力的贡献与计算机的进步是同等重要的. 计算数学与计算机的发展史也证明了这一论断. 以典型的 Laplace 方程求解问题为例, 过去几十年中, 计算机的发展与计算方法的进步, 基本上平分秋色, 参见图 1.1.1.

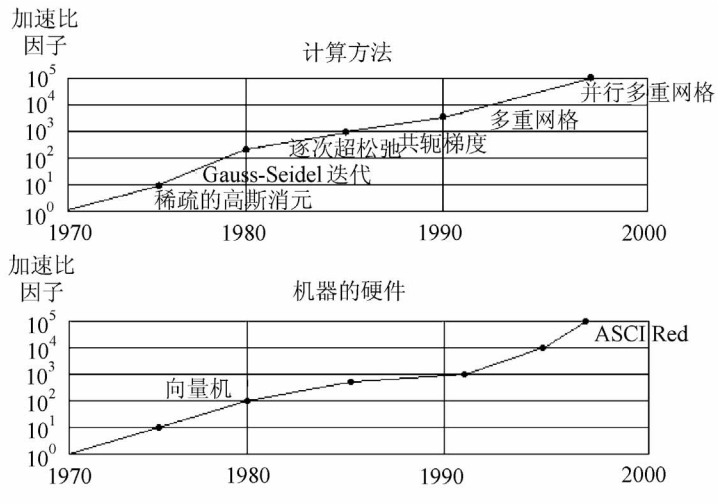


图 1.1.1 Laplace 方程问题计算机的发展与计算方法进步的比较

由此可见, 设计好的算法是尤为重要的. 一般认为, 一个好的算法的评价标准是:

- 运算次数少;
- 运算过程具有规律性, 便于编制程序;
- 要记录的中间结果少;
- 能控制误差的传播和积累, 以保证精度.

简言之, 上述标准就是要求一个好的算法应该既“快”又“准”. 但在实际应用中, 二者一般并不能兼顾, 这就需要根据需要, 权衡利弊, 有所取舍. 例如, 短期天气预报, 预报时间要求紧迫, 应着重于“快”; 而要计算某些精密仪器的设计参数, 就要侧重于“准”.

要判断一个算法的快慢程度和误差对计算结果的影响需要进行适当的理论分析, 在本章的后继部分, 将对这些理论分析所涉及的基本概念和所用的基本方法进行概括性的介绍.

1.2　误差的基本概念

1. 误差来源

通过科学计算求出的解通常都是近似解, 原问题的真解与这个近似解之间的偏差就是误差. 误差的来源有如下 4 个方面：

(1) **模型误差**　从实际问题提炼出数学模型时往往忽略了许多次要因素, 因而即使数学模型能求出准确解, 也与实际问题的真解不同, 它们之差称为模型误差. 一个模型的好坏, 模型误差是关键的因素, 现代计算科学越来越重视模型的建立, 这是因为往往模型误差是 4 种误差中最大的, 而且不可能用数学手段减少.

(2) **观测误差**　原始数据是由观测、实验, 并加以记录而获得的. 由于仪器的精密性、实验手段的局限性、周围环境的变化以及人的工作态度和能力等因素, 而使数据必然带有误差, 这种误差称做观测误差.

(3) **截断误差**　理论上的精确值往往要求用无限次的运算才能获得, 而实际计算时只能用有限次运算的结果来近似, 这样引起的误差称做截断误差.

(4) **舍入误差**　计算机只能对有限位数字进行运算, 每个超出计算机字长的数据都要经过取舍或截断方法处理, 这样引起的误差称做舍入误差.

为了对这些误差有一个更清晰的了解, 下面看一个简单的例子.

假如希望知道地球的表面积是多少, 则首先将地球近似地看作球体, 可得到如下的计算公式：

$$A = 4\pi r^2.$$

显然, 即使由这一公式可以精确地计算, 所得到的值也与地球的真实表面积有一定的误差, 这一步的误差就是模型误差. 假如在计算

中取地球的半径 $r = 6370\text{km}$, 这一值是由前人观测和计算得到的, 具有一定的误差, 这就是观测误差. 公式中的 π 是圆周率, 它是一个无理数, 必须取它的有限位, 例如取 $\pi = 3.141$, 这样就产生了截断误差. 最后将 $4 \times 3.141 \times 6370^2$ 的计算结果进行四舍五入得

$$A = 5.0981 \times 10^8,$$

这一步就产生了舍入误差.

2. 绝对误差、相对误差和有效数字

设 x 为某量的精确值, $\widetilde{x}$ 为其近似值, 称

$$e(x) = x - \widetilde{x}$$

为近似值 $\widetilde{x}$ 的**绝对误差**. 由于精确值 x 一般是未知的, 通常只能给出绝对误差大小 $|x - \widetilde{x}|$ 的某个上界 ε, 即

$$|x - \widetilde{x}| \leqslant \varepsilon.$$

通常称这个数 ε 为近似值 $\widetilde{x}$ 的**绝对误差限**.

有了绝对误差限后, 工程上常用

$$x = \widetilde{x} \pm \varepsilon$$

表示精确值所在的范围. 请注意, 此式的确切含义为

$$\widetilde{x} - \varepsilon \leqslant x \leqslant \widetilde{x} + \varepsilon.$$

而且在很多时候, 将绝对误差限 ε 称为绝对误差, 本书也沿用这一习惯.

但是, 绝对误差还不足以刻画出近似数的精确程度, 需要引入一个更能刻画出误差本质的概念, 这就是相对误差的概念. 称绝对误差与精确值之比

$$e_r = \frac{e(x)}{x} = \frac{x - \widetilde{x}}{x}$$

为近似值 $\widetilde{x}$ 的**相对误差**, 而称

$$\varepsilon_r = \frac{\varepsilon}{|x|}$$

为近似值 $\widetilde{x}$ 的**相对误差限**, 其中 $\varepsilon > 0$ 为近似值 $\widetilde{x}$ 的绝对误差限. 由于 x 一般是未知的, 常常用

$$e_r = \frac{x - \widetilde{x}}{\widetilde{x}}$$

来表示相对误差, 并用

$$x = \widetilde{x}(1 \pm \varepsilon_r)$$

表示精确值所在的范围.

在实际应用中, 还经常用有效数字的概念来反映一个近似数的准确程度. 若

$$|x - \widetilde{x}| \leqslant \frac{1}{2} \times 10^{-k}$$

其中 k 为非负整数, 则称 $\widetilde{x}$ 是 x 的一个近似到小数点后第 k 位的近似值, 并称从小数点后的第 k 位数字起直到最左边的非零数字之间的所有数字为**有效数字**.

把 $\widetilde{x}$ 写成规范形式

$$\widetilde{x} = \pm 0.a_1 a_2 \cdots a_i \cdots \times 10^m,$$

其中 m 为整数, $a_i \in \{0, 1, 2, \cdots, 9\}, a_1 \neq 0$. 如果有

$$|x - \widetilde{x}| \leqslant \frac{1}{2} \times 10^{m-n}$$

则 $\widetilde{x}$ 有 n 位有效数字：$a_1, a_2, \cdots, a_n$.

近似值的有效数字位数与相对误差限密切相关, 具有 n 位有效数字与相对误差限的量级 10^{-n} 是等价的. 例如, $\widetilde{\pi} = 3.1416 = 0.31416 \times 10$ 为 π 的近似值, 由于

$$|\pi - \widetilde{\pi}| < \frac{1}{2} \times 10^{1-5}$$

所以 $\tilde{\pi}$ 是 π 的具有 5 位有效数字的近似值.

从准确值按四舍五入原则截取得到的近似数, 其每一位数字都是有效数字. 值得注意的是, 从计算机输出的近似值, 一般并非每一位数字都是有效数字.

3. 运算误差分析

任何数学问题的解 y 总与某些参量 $x_1, x_2, \cdots, x_n$ 有关, 即

$$y = \varphi(x_1, x_2, \cdots, x_n),$$

参量的值若有误差, 则解也有误差, 设 $x_1, x_2, \cdots, x_n$ 的近似值为 $\widetilde{x}_1$, $\widetilde{x}_2, \cdots, \widetilde{x}_n$, 相应解为 $\widetilde{y}$, 则绝对误差为

$$|y - \widetilde{y}| = |\varphi(x_1, x_2, \cdots, x_n) - \varphi(\widetilde{x}_1, \widetilde{x}_2, \cdots, \widetilde{x}_n)|,$$

相对误差为

$$\frac{|y - \widetilde{y}|}{|y|} = \frac{|y - \widetilde{y}|}{|\varphi(x_1, x_2, \cdots, x_n)|}.$$

由此得误差估计

$$|y - \widetilde{y}| \leqslant \sum_{i=1}^{n} \left| \frac{\partial \varphi}{\partial x_i} \right| |x_i - \widetilde{x}_i|,$$

$$\frac{|y - \widetilde{y}|}{|y|} \leqslant \sum_{i=1}^{n} \left| \frac{x_i}{y} \frac{\partial \varphi}{\partial x_i} \right| \frac{|x_i - \widetilde{x}_i|}{|x_i|}.$$

在这两个公式中,

$$\left| \frac{\partial \varphi}{\partial x_i} \right| \text{ 和 } \left| \frac{x_i}{y} \frac{\partial \varphi}{\partial x_i} \right|$$

表示解的误差相对参量 x_i 的误差放大或缩小的倍数.

把两个数 a, b 的 $+$、$-$、$\times$、$\div$ 看成二元函数, 则有:

$$|e(a \pm b)| \leqslant |e(a)| + |e(b)|, \tag{1.2.1}$$

$$|e_r(a \pm b)| \leqslant \frac{|e(a)| + |e(b)|}{|a \pm b|} = \frac{|a|\,|e_r(a)| + |b|\,|e_r(b)|}{|a \pm b|}, \tag{1.2.2}$$

$$|e(ab)| \leqslant |b|\,|e(a)| + |a|\,|e(b)|, \tag{1.2.3}$$

$$|e_r(ab)| \leqslant \frac{|e(a)|}{|a|} + \frac{|e(b)|}{|b|} = |e_r(a)| + |e_r(b)|, \tag{1.2.4}$$

$$\left|e\left(\frac{a}{b}\right)\right| \leqslant \frac{|e(a)|}{|b|} + \left|\frac{a}{b^2}\right| |e(b)|, \tag{1.2.5}$$

$$\left|e_r\left(\frac{a}{b}\right)\right| \leqslant |e_r(a)| + |e_r(b)|. \tag{1.2.6}$$

其中式 (1.2.2) 表明：在绝对误差 $e(a), e(b)$ 不变时, a 与 b 越接近, 则 $a - b$ 的相对误差就变得越大. 也就是说, 相近数相减会严重丢失有效数字. 式 (1.2.3) 表明, 在作乘法运算时, 两数中如果有一个绝对值很大的数, 则积的绝对误差就会很大. 式 (1.2.5) 表明, 在作除法运算时, 如果除数的绝对值很小, 则商的绝对误差就会很大.

1.3 浮 点 数 系

在数字电子计算机上, 实数系 $\mathbb{R}$ 是用所谓的浮点数系统 $\mathfrak{F}$ 来近似的. 二进制浮点数系统 $\mathfrak{F}(2, t, L, U)$ 定义为

$$\mathfrak{F} = \{\pm 0.d_1 d_2 \cdots d_t \times 2^m\} \bigcup \{0\},$$

其中 $d_1 = 1$, $d_j (2 \leqslant j \leqslant t)$ 或为 0 或为 1; t 为一个给定的自然数, 称为字长; m 为整数, 满足 $L \leqslant m \leqslant U$, L, U 为给定的整数.

表 1.3.1 列出了几种浮点数系统, 其中 IEEE(Institute of Electrical and Electronics Engineers) 标准中规定的系统最常见.

$\forall x \in \mathbb{R}$, 它总可以写成

$$x = \pm 0.1 d_2 \cdots d_t d_{t+1} \cdots \times 2^m.$$

表 1.3.1 几种浮点数系统

系统	进制	t	L	U
IEEE 单精度	2	24	-126	127
IEEE 双精度	2	53	-1022	1023
Cray	2	48	-16383	16384
HP calculator	10	12	-499	499
IBM mainframe	16	6	-64	63

记 x 在一个浮点数系 $\mathfrak{F}$ 中的表示为 $fl(x)$. 一般来说, $fl(x)$ 只是 x 的某种近似. 常用的近似方法有两种, 一种是截断 (chop), 另一种是舍入 (round to nearest). 在 IEEE 标准中采用的是舍入. 于是有

$$\left|\frac{fl(x)-x}{x}\right| \leqslant 2^{-t}.$$

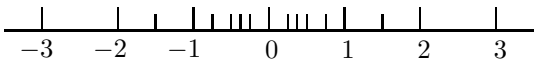
这就是实数的浮点数表示所产生的误差. 将常数 2^{-t} 称为机器精度, 常常记为 $\varepsilon_{\text{mach}}$.

总之, 浮点数系统 $\mathfrak{F}$ 有下列值得注意的特点:

- 是实数系 $\mathbb{R}$ 的一个只包含 $2^t(U-L+1)+1$ 个数的有限集, 这些数对称地离散分布在区间 $[UFL, OFL]$ 和 $[-OFL, -UFL]$ 中, 其中

$$UFL = 2^L \times 0.1, \qquad OFL = 0.11\cdots1 \times 2^U. \tag{1.3.1}$$

  值得注意的是, 这些数的分布是不等距的. 例如, 若 $\beta=2$, $t=2$, $L=-1$ 和 $U=2$, 则 $\mathfrak{F}$ 中 17 个数的分布如下图所示.



- 具有最小正数 $UFL = 2^L \times 0.1$.
- 具有最大正数 $OFL = 0.11\cdots1 \times 2^U$.

- 具有机器精度 $\varepsilon_{\text{mach}} = 2^{-t}$.
- 其中的四则运算并不满足实数系中的运算规律.

在一个浮点数系统中 (或一台计算机上), 绝对值超过 OFL 的实数就被认为是无穷大, 绝对值小于 UFL 的实数就被认为是零. 最后需要指出的是, 由于浮点数系只是一个有限的数集, 因而一些数学上等价的计算公式在计算机上的计算结果可能是不相同的. 例如, 数学上有三角恒等式

$$\sin(x+\varepsilon) - \sin x = 2\cos\Big(x + \frac{\varepsilon}{2}\Big)\sin\frac{\varepsilon}{2},$$

当 ε 比较小时, 在计算机上用后者计算的结果要比用前者计算的结果精确得多, 这是因为两个相近的数相减, 在计算机上会导致严重损失有效数字. 再如假设在字长为 8 的十进制浮点数系下计算 $(x+y)+z$ 和 $x+(y+z)$, 其中

$$\begin{aligned} x &= 0.23371258 \times 10^{-4}, \\ y &= 0.33678429 \times 10^{2}, \\ z &= -0.33677811 \times 10^{2}, \end{aligned}$$

则由浮点数的运算规则可得

$$\begin{aligned} (x+y)+z &= 0.64100000 \times 10^{-3}, \\ x+(y+z) &= 0.64137126 \times 10^{-3}, \end{aligned}$$

其精确值为

$$x+y+z = 0.641371258 \times 10^{-3}.$$

比较可以发现, $(x+y)+z \neq x+(y+z)$, 而且 $x+(y+z)$ 要比 $(x+y)+z$ 精确得多, 这是由于计算机上 "大数吃小数" 的原因而导致的. 因此, 在设计算法时要尽可能避免一个很大的数和一个很小的数相加的