第 1 章　数值分析与科学计算引论

1.1　数值分析的对象、作用与特点

1.1.1　数学科学与数值分析

数学是科学之母，科学技术离不开数学，通过建立数学模型可使科学技术与数学产生紧密联系，数学又以各种形式应用于科学技术的各个领域.**数值分析**也称**计算数学**，是数学科学的一个分支，它研究用计算机求解各种数学问题的数值计算方法及其理论与软件实现.

用计算机求解科学技术问题通常经历以下步骤：

① 根据实际问题建立数学模型、分析数学模型的性质（如模型解的存在性及唯一性）.

② 由数学模型给出数值计算方法.

③ 根据计算方法编制算法程序（数学软件）在计算机上算出结果.

第①步建立数学模型、分析数学模型的性质通常是应用数学的任务，而第②③步就是计算数学的任务，也就是数值分析研究的对象，它涉及数学的各个分支，内容十分广泛.作为“数值分析”课程，只介绍其中最基本、最常用的数值计算方法及其理论，它包括插值与数据逼近，数值微分与积分，线性方程组的数值求解，非线性方程与方程组求解，特征值与特征向量计算，常微分方程数值解等.它们都是以数学问题为研究对象，只是不像纯数学那样只研究数学本身的理论，而是把理论与计算紧密结合，着重研究数学问题的数值算法及其理论.与其他数学课程一样，数值分析也是一门内容丰富、研究方法深刻、有自身理论体系的课程，既有纯数学高度抽象性与严密科学性的特点，又有应用广泛性与实际试验高度技术性的特点，是一门与计算机使用密切结合，实用性很强的数学课程.

1.1.2　计算数学与科学计算

几十年来由于计算机及科学技术的快速发展，求解各种数学问题的数值方法也越来越多地应用于科学技术的各个领域，新的计算性交叉学科分支不断涌现，如计算力学、计算物理、计算化学、计算生物学、计算经济学等，统称**科学计算**，它涉及数学的各个分支，研究它们适合于计算机编程的算法就是计算数学的研究范畴. 计算数学是各种计算性学科的共性基础，兼有基础性、应用性和边缘性的数学学科. 科学计算是一门工具性、方法性、边缘性的学科，且发展迅速.它与理论研究和科学实验成为现代科学发展的三种主要手段，它们相辅相成又互相独立.在实际应用中导出的数学模型其完备形式往往不能方便地求出精确解，于是只能转化为简化模型求其数值解，如将复杂的非线性模型忽略一些因素而简化为可以求出精确解的线性模型，但这样做往往不能满足近似程度的要求.因此使用数值方法直接求解做较少简化的模型，可以得到满足近似程度要求的结果，使科学计算发挥更大的作用，这正是得益于计算机与计算数学的快速发展.

1.1.3 计算方法与计算机

数值分析也称**计算方法**,它与计算工具发展密切相关. 在电子计算机出现以前,计算工具只有算盘、算图、算表、算尺和手摇及电动计算机,计算方法只能计算规模较小的问题. 计算方法是数学的一个组成部分,很多方法都与当时的数学家名字相联系,如牛顿(Newton)插值公式、方程求根的牛顿法、解线性方程组的高斯(Gauss)消去法、多项式求值的秦九韶算法、计算积分的辛普森(Simpson)公式等. 这表明计算方法就是数学的一部分,起始它没有形成单独的学科分支. 只是在计算机出现以后,才使计算方法迅速发展并形成数学科学的一个独立分支——计算数学.

当代计算能力的大幅提高既来自计算机的进步,也来自计算方法的进步,两者发展相辅相成又互相促进. 例如,1955—1975 年的 20 年间计算机的运算速度提高数千倍,而同一时期解决一定规模的椭圆形偏微分方程计算方法的效率提高约一百万倍,说明计算方法的进步对提高计算能力的贡献更为重要,由于计算规模的不断扩大和计算方法的发展启发了新的计算机体系结构,诞生并发展了并行计算机.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. 自计算机诞生以来,经典的计算方法业已经历了一个重新评价、筛选、改造和创新的过程,与此同时涌现了许多新概念、新课题和能发挥计算机解题潜力的新方法,这就构成了现代意义的计算数学.

1.1.4 数值问题与算法

能用计算机计算的"**数值问题**"是指输入数据(即问题中的自变量与原始数据)与输出数据(结果)之间函数关系的一个确定而无歧义的描述,输入、输出数据可用有限维向量表示. 根据这种定义,"数学问题"有的是"数值问题",如线性方程组求解. 也有不是"数值问题"的"数学问题",如常微分方程$\dfrac{\mathrm{d}y}{\mathrm{d}x}=x^2+y^2$,$y(0)=0$,它不是数值问题,因为输出不是数据而是连续函数 $y=y(x)$. 但只要将连续问题离散化,使输出数据是 $y(x)$在求解区间$[a,b]$上的离散点 $x_i=a+ih(i=1,2,\cdots,n)$上的近似值,就是"数值问题". 数值问题可用各种数值方法求解,这些数值方法就是算法. 计算方法就是研究各种"数值问题"的算法.

计算机具有运算速度快,适合做重复性操作的特点.在计算机中,计算的基本单位称为算法元,它由输入元、算子和输出元组成. 算子可以是简单操作,如算术运算(+,−,×,/)、逻辑运算,也可以是宏操作,如向量运算、数组传输、基本初等函数求值等;输入元和输出元可分别视为若干变量或向量. 由一个或多个算法元组成一个进程,它是算法元的有限序列,一个数值问题的**算法**是指按规定顺序执行一个或多个完整的进程,通过它们将输入元变换成一个输出元. 从理论上来看,一个数值问题可能是一个无限的过程,如问题的解是一个序列的极限,这是一个无限的过程.但实际用算法实现时,只能取有限步. 这就要求在可接受的精确程度下终止极限过程. 面向计算机的算法可分为串行算法和并行算法两类,只有一个进程的算法适合于串行计算机,称为串行算法. 有两个以上进程的算法适合于并行计算机,称为并行算法.

对于一个给定的数值问题可能有许多不同的算法,它们都能给出近似答案,但所需的计算量和得到答案的精确程度可能相差很大. 一个面向计算机,有可靠理论分析且计算复杂

性好的算法就是一个好算法. 理论分析主要是连续系统的离散化及离散型方程的数值问题求解,它包括误差分析、稳定性、收敛性等基本概念,它刻画了算法的可行性、稳定性、准确性. 计算复杂性包含计算时间复杂性与存储空间复杂性两个方面. 在同一规模、同一精度条件下,计算时间少的算法为计算时间复杂性好,而占用内存空间少的算法为存储空间复杂性好,它实际上就是算法中计算量与存储量的分析. 对解同一问题的不同算法其计算复杂性可能差别很大,例如解 n 阶的线性方程组,依照克莱姆(Cramer)法则要算 $n+1$ 个 n 阶行列式,如果按行列式的原始定义来计算行列式,那么对 $n=20$ 的线性方程组就需要 9.7×10^{20} 次乘除法运算,若用每秒亿次的计算机也要算 30 万年,这是无法实现的,若用高斯列主元消去法(见第 5 章)则只需做 3060 次乘除运算. 且 n 越大相差就越大,这表明算法研究的重要性,也说明只提高计算机速度,而不改进和选用好的算法是不行的.

综上所述,数值分析是研究数值问题的算法,概括起来有四点:

第一,面向计算机,要根据计算机的特点提供切实可行的有效算法. 即算法只能包括加、减、乘、除运算和逻辑运算,这些运算是计算机能直接处理的运算.

第二,有可靠的理论分析,能任意逼近并达到精度要求,对近似算法要保证收敛性和数值稳定性,还要对误差进行分析. 这些都建立在相应数学理论的基础上.

第三,要有好的计算复杂性,时间复杂性好是指节省计算时间,空间复杂性好是指节省存储空间,这也是建立算法要研究的问题,它关系到算法能否在计算机上实现.

第四,要有数值实验,即任何一个算法除了从理论上要满足上述三点外,还要通过数值试验证明是行之有效的.

根据"数值分析"课程的特点,学习时我们首先要注意掌握方法的基本原理和思想,要注意方法处理的技巧及其与计算机的结合,要重视误差分析、收敛性及稳定性的基本理论;其次,要通过例子,学习使用各种数值方法解决实际计算问题;最后,为了掌握本课的内容,还应做一定数量的理论分析与计算练习. 由于本课内容包括了微积分、线性代数、常微分方程的数值方法,读者必须掌握这几门课中与数值分析相关的基本内容,才能学好这门课程.

1.2 数值计算的误差

1.2.1 误差来源与分类

用计算机解决科学计算问题首先要建立数学模型,它是对被描述的实际问题进行抽象、简化而得到的,因而是近似的. 我们把数学模型与实际问题之间出现的这种误差称为**模型误差**. 只有实际问题提法正确,建立数学模型时又抽象、简化得合理,才能得到好的结果. 由于这种误差难于用数量表示,通常都假定数学模型是合理的,这种误差可忽略不计,在"数值分析"中不予讨论. 在数学模型中往往还有一些根据观测得到的物理量,如温度、长度、电压等,这些参量显然也包含误差. 这种由观测产生的误差称为**观测误差**,在"数值分析"中也不讨论这种误差. 数值分析只研究用数值方法求解数学模型产生的误差.

当数学模型不能得到精确解时,通常要用数值方法求它的近似解,其近似解与精确解之间的误差称为**截断误差**或**方法误差**. 例如,可微函数 $f(x)$ 用 n 次泰勒(Taylor)多项式

$$p_n(x)=f(0)+\frac{f'(0)}{1!}x+\frac{f''(0)}{2!}x^2+\cdots+\frac{f^{(n)}(0)}{n!}x^n$$

近似代替,则数值方法的截断误差是

$$R_n(x)=f(x)-p_n(x)=\frac{f^{(n+1)}(\xi)}{(n+1)!}x^{n+1},\quad \xi \text{ 在 } 0 \text{ 与 } x \text{ 之间.}$$

有了求解数学问题的计算公式以后,用计算机做数值计算时,由于计算机的字长有限,原始数据在计算机中表示时会产生误差,计算过程又可能产生新的误差,这种误差称为**舍入误差**. 例如,用 3.141 59 近似代替 π,产生的误差

$$R=\pi-3.141\,59=0.000\,002\,6\cdots$$

就是舍入误差.

此外由原始数据或机器中的十进制数转化为二进制数产生的初始误差对数值计算也将造成影响,分析初始数据的误差通常也归结为舍入误差.

研究计算结果的误差是否满足精度要求就是误差估计问题,本书主要讨论算法的截断误差与舍入误差,而截断误差将结合具体算法讨论. 为分析数值运算的舍入误差,先要对误差这个基本概念做简单介绍,尽管在高中物理学习测量知识时可能接触过此概念.

1.2.2 误差限与有效数字

设 x 为准确值,x^* 为 x 的一个近似值,称 $e^*=x^*-x$ 为近似值的**绝对误差**,简称**误差**.

通常我们不能算出准确值 x,因此也不能算出误差 e^* 的准确值,只能根据测量工具或计算情况估计出误差的绝对值不超过某正数 ε^*,也就是误差绝对值的一个上界. ε^* 称为近似值的**误差限**,它总是正数. 例如,用毫米刻度的米尺测量一长度 x,读出和该长度接近的刻度 x^*,x^* 是 x 的近似值,它的误差限是 0.5 mm,于是 $|x^*-x|\leqslant 0.5$ mm;如读出的长度为 765 mm,则有 $|765-x|\leqslant 0.5$. 从这个不等式我们仍不知道准确的 x 是多少,但知道 $764.5\leqslant x\leqslant 765.5$,即 x 在区间[764.5,765.5]上.

对于一般情形,$|x^*-x|\leqslant\varepsilon^*$,即

$$x^*-\varepsilon^*\leqslant x\leqslant x^*+\varepsilon^*,$$

这个不等式有时也表示为 $x=x^*\pm\varepsilon^*$.

误差限的大小还不能完全表示近似值的好坏. 例如,有两个量 $x=10\pm1$,$y=1000\pm5$,则

$$x^*=10,\quad \varepsilon_x^*=1;\quad y^*=1000,\quad \varepsilon_y^*=5.$$

虽然 ε_y^* 是 ε_x^* 的 5 倍,但 $\varepsilon_y^*/y^*=\dfrac{5}{1000}=0.5\%$ 比 $\varepsilon_x^*/x^*=\dfrac{1}{10}=10\%$ 要小得多,这说明 y^* 近似 y 的程度比 x^* 近似 x 的程度要好得多. 所以,除考虑误差的大小外,还应考虑准确值 x 本身的大小. 我们把近似值的误差 e^* 与准确值 x 的比值

$$\frac{e^*}{x}=\frac{x^*-x}{x}$$

称为近似值 x^* 的**相对误差**,记作 e_r^*.

在实际计算中,由于真值 x 总是不知道的,通常取

$$e_r^*=\frac{e^*}{x^*}=\frac{x^*-x}{x^*}$$

作为 x^* 的相对误差,条件是 e_r^* 较小,此时

$$\frac{e^*}{x}-\frac{e^*}{x^*}=\frac{e^*(x^*-x)}{x^*x}=\frac{(e^*)^2}{x^*(x^*-e^*)}=\frac{(e^*/x^*)^2}{1-(e^*/x^*)}$$

是 e_r^* 的平方项级，故可忽略不计.

相对误差也可正可负，它的绝对值上界称为**相对误差限**，记作 ε_r^*，即 $\varepsilon_r^*=\dfrac{\varepsilon^*}{|x^*|}$.

在前面所举的例子中 $\dfrac{\varepsilon_x^*}{|x^*|}=10\%$ 与 $\dfrac{\varepsilon_y^*}{|y^*|}=0.5\%$ 分别为 x 与 y 的相对误差限，可见 y^* 近似 y 的程度比 x^* 近似 x 的程度好.

当准确值 x 有多位数时，基于数字在计算机中的浮点数表示形式，常常按四舍五入的原则得到 x 的前几位近似值 x^*. 一个近似数四舍五入到哪一位，就称它精确到哪一位，这时，该位到其左边第一位非零数字止的所有数字，称为这个近似数的**有效数字**，有效数字的个数称为**有效数字的位数**.例如

$$x=\pi=3.141\ 592\ 65\cdots,$$

四舍五入到第 3 位　$x_3^*=3.14$，$\varepsilon_3^*\leqslant 0.002$，有 3 位有效数字，

四舍五入到第 5 位　$x_5^*=3.1416$，$\varepsilon_5^*\leqslant 0.000\ 008$，有 5 位有效数字，

它们的误差都不超过末位数字的半个单位，即

$$|\pi-3.14|\leqslant\frac{1}{2}\times10^{-2},\quad |\pi-3.1416|\leqslant\frac{1}{2}\times10^{-4}.$$

若近似值 x^* 的误差限是某一位的半个单位，该位到 x^* 的第一位非零数字共有 n 位，则 x^* 有 n 位有效数字. 它可用科学计数法(浮点形式)表示为

$$x^*=\pm10^m\times(a_1+a_2\times10^{-1}+\cdots+a_n\times10^{-(n-1)}),\tag{1.1}$$

其中 $a_i(i=1,2,\cdots,n)$ 是 0 到 9 中的一个数字，$a_1\neq0$，m 为整数，且

$$|x-x^*|\leqslant\frac{1}{2}\times10^{m-n+1}.\tag{1.2}$$

例 1.1　按四舍五入原则写出下列各数的具有 5 位有效数字的近似数：187.9325，0.037 855 51，8.000 033，2.718 281 8.

解　上述各数的具有 5 位有效数字的近似数分别是

$$187.93,\ 0.037\ 856,\ 8.0000,\ 2.7183.$$

注意 $x=8.000\ 033$ 的 5 位有效数字近似数是 8.0000 而不是 8，因为 8 只有 1 位有效数字.

例 1.2　如果以 m/s^2 为单位，重力常数 $g\approx9.80\ m/s^2$；若以 km/s^2 为单位，$g\approx0.009\ 80\ km/s^2$，它们都具有 3 位有效数字，因为按第一种写法

$$|g-9.80|\leqslant\frac{1}{2}\times10^{-2},$$

根据(1.1)式，这里 $m=0$，$n=3$；按第二种写法

$$|g-0.009\ 80|\leqslant\frac{1}{2}\times10^{-5},$$

这里 $m=-3$，$n=3$. 它们虽然写法不同，但都具有 3 位有效数字. 至于绝对误差限，由于单位不同结果也不同，$\varepsilon_1^*=\dfrac{1}{2}\times10^{-2}\ m/s^2$，$\varepsilon_2^*=\dfrac{1}{2}\times10^{-5}\ km/s^2$. 而相对误差相同，因为

$$\varepsilon_r^* = 0.005/9.80 = 0.000\,005/0.009\,80.$$

注意相对误差与相对误差限是无量纲的，而绝对误差与误差限是有量纲的.

例 1.2 说明有效位数与小数点后有多少位数无关. 然而，从(1.2)式可得到具有 n 位有效数字的近似数 x^*，其绝对误差限为

$$\varepsilon^* = \frac{1}{2} \times 10^{m-n+1},$$

在 m 相同的情况下，n 越大则 10^{m-n+1} 越小，故有效位数越多，绝对误差限越小.

至于有效数字与相对误差限的关系，有下面的定理.

定理 1.1 设近似数 x^* 表示为

$$x^* = \pm 10^m \times (a_1 + a_2 \times 10^{-1} + \cdots + a_l \times 10^{-(l-1)}), \tag{1.1}'$$

其中 $a_i (i = 1, 2, \cdots, l)$ 是 0 到 9 中的一个数字，$a_1 \neq 0$，m 为整数. 若 x^* 具有 n 位有效数字，则其相对误差限

$$\varepsilon_r^* \leqslant \frac{1}{2a_1} \times 10^{-(n-1)};$$

反之，若 x^* 的相对误差限 $\varepsilon_r^* \leqslant \frac{1}{2(a_1+1)} \times 10^{-(n-1)}$，则 x^* 至少具有 n 位有效数字.

证明 由(1.1)′式可得

$$a_1 \times 10^m \leqslant |x^*| < (a_1 + 1) \times 10^m,$$

当 x^* 具有 n 位有效数字时

$$\varepsilon_r^* = \frac{\varepsilon^*}{|x^*|} \leqslant \frac{0.5 \times 10^{m-n+1}}{a_1 \times 10^m} = \frac{1}{2a_1} \times 10^{-n+1};$$

反之，由

$$|x - x^*| \leqslant \varepsilon^* = |x^*| \varepsilon_r^* < (a_1 + 1) \times 10^m \times \frac{1}{2(a_1+1)} \times 10^{-n+1} = 0.5 \times 10^{m-n+1},$$

故 x^* 至少具有 n 位有效数字. □

定理 1.1 说明，有效位数越多，相对误差限越小.

例 1.3 要使 $\sqrt{20}$ 的近似值的相对误差限小于 0.1%，要取几位有效数字？

解 设取 n 位有效数字，由定理 1.1，$\varepsilon_r^* \leqslant \frac{1}{2a_1} \times 10^{-n+1}$. 由 $4 < \sqrt{20} < 5$，知 $a_1 = 4$，故只要取 $n = 4$，就有

$$\varepsilon_r^* \leqslant 0.125 \times 10^{-3} < 10^{-3} = 0.1\%,$$

即只要对 $\sqrt{20}$ 的近似值取 4 位有效数字，其相对误差限就小于 0.1%. 此时 $\sqrt{20} \approx 4.472$，$\sqrt{20} - 4.472 = 4.47\,213\,595 - 4.472 = 1.3595 \times 10^{-4}$.

1.2.3 数值运算的误差估计

设两个近似数 x_1^* 与 x_2^* 的误差限分别为 $\varepsilon(x_1^*)$ 及 $\varepsilon(x_2^*)$，则它们进行加、减、乘、除运算得到的误差限分别满足不等式

$$\varepsilon(x_1^* \pm x_2^*) \leqslant \varepsilon(x_1^*) + \varepsilon(x_2^*);$$

$$\varepsilon(x_1^* x_2^*) \leqslant |x_1^*|\varepsilon(x_2^*)+|x_2^*|\varepsilon(x_1^*);$$

$$\varepsilon(x_1^*/x_2^*) \leqslant \frac{|x_1^*|\varepsilon(x_2^*)+|x_2^*|\varepsilon(x_1^*)}{|x_2^*|^2}, \quad x_2^* \neq 0.$$

更一般的情况是,当自变量有误差时计算函数值也产生误差,其误差限可利用函数的泰勒展开式进行估计. 设 $f(x)$是一元可微函数,x 的近似值为 x^*,以 $f(x^*)$近似 $f(x)$,其误差界记作 $\varepsilon(f(x^*))$,由泰勒展开式

$$f(x)-f(x^*)=f'(x^*)(x-x^*)+\frac{f''(\xi)}{2}(x-x^*)^2, \quad \xi \text{ 介于 } x, x^* \text{ 之间},$$

取绝对值得

$$|f(x)-f(x^*)| \leqslant |f'(x^*)|\varepsilon(x^*)+\frac{|f''(\xi)|}{2}\varepsilon^2(x^*).$$

假定 $f''(x^*)$与 $f'(x^*)$的比值不太大,可忽略 $\varepsilon(x^*)$的高阶项,于是可得计算函数的误差限

$$\varepsilon(f(x^*)) \approx |f'(x^*)|\varepsilon(x^*).$$

当 f 为多元函数时,例如计算 $A=f(x_1,x_2,\cdots,x_n)$. 如果 $x_1,x_2,\cdots,x_n$ 的近似值为 $x_1^*,x_2^*,\cdots,x_n^*$,则 A 的近似值为 $A^*=f(x_1^*,x_2^*,\cdots,x_n^*)$,于是由泰勒展开式得函数值 A^* 的误差 $e(A^*)$为

$$e(A^*)=A^*-A=f(x_1^*,x_2^*,\cdots,x_n^*)-f(x_1,x_2,\cdots,x_n)$$

$$\approx \sum_{k=1}^{n}\left(\frac{\partial f(x_1^*,x_2^*,\cdots,x_n^*)}{\partial x_k}\right)(x_k^*-x_k)=\sum_{k=1}^{n}\left(\frac{\partial f}{\partial x_k}\right)^* e_k^*,$$

于是误差限

$$\varepsilon(A^*) \approx \sum_{k=1}^{n}\left|\left(\frac{\partial f}{\partial x_k}\right)^*\right|\varepsilon(x_k^*); \tag{1.3}$$

而 A^* 的相对误差限为

$$\varepsilon_r^* = \varepsilon_r(A^*)=\frac{\varepsilon(A^*)}{|A^*|} \approx \sum_{k=1}^{n}\left|\left(\frac{\partial f}{\partial x_k}\right)^*\right|\frac{\varepsilon(x_k^*)}{|A^*|}.$$

例 1.4 已测得某场地的长 l 的值为 $l^*=110$ m,宽 d 的值为 $d^*=80$ m,已知 $|l-l^*| \leqslant 0.2$ m,$|d-d^*| \leqslant 0.1$ m. 试求面积 $s=ld$ 的绝对误差限与相对误差限.

解 因 $s=ld$, $\dfrac{\partial s}{\partial l}=d$, $\dfrac{\partial s}{\partial d}=l$,由(1.3)式知

$$\varepsilon(s^*) \approx \left|\left(\frac{\partial s}{\partial l}\right)^*\right|\varepsilon(l^*)+\left|\left(\frac{\partial s}{\partial d}\right)^*\right|\varepsilon(d^*),$$

其中

$$\left(\frac{\partial s}{\partial l}\right)^*=d^*=80 \text{ m}, \quad \left(\frac{\partial s}{\partial d}\right)^*=l^*=110 \text{ m},$$

而 $\varepsilon(l^*)=0.2$ m,$\varepsilon(d^*)=0.1$ m,于是绝对误差限

$$\varepsilon(s^*) \approx 80\times(0.2)+110\times(0.1)=27 \text{ m}^2;$$

相对误差限

$$\varepsilon_r(s^*)=\frac{\varepsilon(s^*)}{|s^*|}=\frac{\varepsilon(s^*)}{l^*d^*}\approx\frac{27}{8800}=0.31\%.$$

1.3 误差定性分析与避免误差危害

数值运算中的误差分析是个重要且复杂的问题. 1.2 节讨论了不精确数据运算结果的误差限,它只适用于简单情形,然而一个工程或科学计算问题往往要运算千万次. 由于每步运算都有误差,如果每步都做误差分析是不可能的,也不科学,因为误差积累有正有负,绝对值有大有小,都按最坏的情况估计误差限得到的结果比实际误差大得多,这种保守的误差估计不能反映实际误差积累. 考虑到误差分布的随机性,有人用概率统计方法,将数据和运算中的舍入误差视为适合某种分布的随机变量,然后确定计算结果的误差分布,这样得到的误差估计更接近实际,这种方法称为**概率分析法**.

20 世纪 60 年代以后人们对舍入误差估计提出了一些新方法,较重要的是威尔金森(Wilkinson)的向后误差分析法和穆尔(Moore)的区间分析法.但都不是十分有效,到目前为止舍入误差的定量估计尚无有效的分析方法,为确保数值计算的正确性通常只进行定性分析.

1.3.1 算法的数值稳定性

用一个算法进行计算,由于初始数据误差在计算中传播使计算结果误差增长很快,算法就是数值不稳定的,先看下例.

例 1.5 计算 $I_n=\mathrm{e}^{-1}\int_0^1 x^n\mathrm{e}^x\,\mathrm{d}x\,(n=0,1,2,\cdots)$ 并估计误差.

解 由分部积分可得计算 I_n 的递推公式

$$\begin{cases}I_0=\mathrm{e}^{-1}\int_0^1\mathrm{e}^x\,\mathrm{d}x=1-\mathrm{e}^{-1},\\ I_n=1-nI_{n-1},\quad n=1,2,\cdots.\end{cases}\tag{1.4}$$

若计算出 I_0,代入(1.4)式,可逐次求出 $I_1,I_2,\cdots$的值. 要算出 I_0 就要先计算 e^{-1},若用泰勒多项式展开部分和

$$\mathrm{e}^{-1}\approx 1+(-1)+\frac{(-1)^2}{2!}+\cdots+\frac{(-1)^k}{k!},$$

并取 $k=7$,用 4 位小数计算,则得 $\mathrm{e}^{-1}\approx0.3679$,截断误差

$$R_7=|\,\mathrm{e}^{-1}-0.3679\,|\leqslant\frac{1}{8!}.$$

计算过程中小数点后第 5 位的数字按四舍五入原则舍入,由此产生的舍入误差这里先不讨论. 当初值取为 $I_0\approx0.6321=\tilde{I}_0$ 时,用(1.4)式递推的计算方法为

$$\text{算法(A)}\begin{cases}\tilde{I}_0=0.6321,\\ \tilde{I}_n=1-n\tilde{I}_{n-1},\quad n=1,2,\cdots.\end{cases}$$

计算结果见表 1-1 的 $\tilde{I}_n$ 列. 用 $\tilde{I}_0$ 近似 I_0 产生的误差 $E_0=I_0-\tilde{I}_0$ 就是初值误差,它对后面计算结果是有影响的.

表 1-1 计算结果

n	$\tilde{I}_n$(用算法(A))	I_n^*(用算法(B))	n	$\tilde{I}_n$(用算法(A))	I_n^*(用算法(B))
0	0.6321 ↓	0.6321	5	0.1480 ↓	0.1455
1	0.3679	0.3679	6	0.1120	0.1268
2	0.2642	0.2642	7	0.2160	0.1121
3	0.2074	0.2073 ↑	8	−0.7280	0.1035 ↑
4	0.1704	0.1709	9	7.5520	0.0684

从表 1-1 中看到 $\tilde{I}_8$ 出现负值,这与一切 $I_n>0$ 相矛盾. 实际上,由积分估值得

$$\frac{\mathrm{e}^{-1}}{n+1}=\mathrm{e}^{-1}\left(\min_{0\leqslant x\leqslant 1}\mathrm{e}^{x}\right)\int_0^1 x^n\mathrm{d}x<I_n<\mathrm{e}^{-1}\left(\max_{0\leqslant x\leqslant 1}\mathrm{e}^{x}\right)\int_0^1 x^n\mathrm{d}x=\frac{1}{n+1}. \tag{1.5}$$

因此,当 n 较大时,用 $\tilde{I}_n$ 近似 I_n 显然是不正确的. 这里计算公式与每步计算都是正确的,那么是什么原因使计算结果出现错误呢? 主要就是初值 $\tilde{I}_0$ 有误差 $E_0=I_0-\tilde{I}_0$,由此引起以后各步计算的误差 $E_n=I_n-\tilde{I}_n$ 满足关系

$$E_n=-nE_{n-1},\quad n=1,2,\cdots.$$

由此容易推得

$$E_n=(-1)^n n!\ E_0,$$

这说明 $\tilde{I}_0$ 有误差 E_0,则 $\tilde{I}_n$ 就是 E_0 的 $n!$ 倍误差.

例如,$n=8$,若 $|E_0|=\frac{1}{2}\times 10^{-4}$,则 $|E_8|=8!\times|E_0|>\frac{1}{2}$. 这就说明 $\tilde{I}_8$ 完全不能近似 I_8 了. 它表明算法(A)是数值不稳定的.

我们现在换一种计算方法. 由(1.5)式取 $n=9$,得

$$\frac{\mathrm{e}^{-1}}{10}<I_9<\frac{1}{10},$$

我们粗略取 $I_9\approx\frac{1}{2}\left(\frac{1}{10}+\frac{\mathrm{e}^{-1}}{10}\right)=0.0684=I_9^*$,然后将(1.4)式倒过来算,即由 I_9^* 算出 I_8^*,I_7^*,…,I_0^*,计算方法为

$$\text{算法(B)}\begin{cases}I_9^*=0.0684,\\ I_{n-1}^*=\dfrac{1}{n}(1-I_n^*),\quad n=9,8,\cdots,1;\end{cases}$$

计算结果见表 1-1 的 I_n^* 列. 我们发现 I_0^* 与 I_0 的误差不超过 10^{-4}. 记 $E_n^*=I_n-I_n^*$,则 $|E_0^*|=\frac{1}{n!}|E_n^*|$,$E_0^*$ 比 E_n^* 缩小了 $n!$ 倍,因此,尽管 E_9^* 较大,但由于误差逐步缩小,故可用 I_n^* 近似 I_n. 反之,当用算法(A)计算时,尽管初值 $\tilde{I}_0$ 相当准确,由于误差传播是逐步扩大的,因而计算结果不可靠. 此例说明,数值不稳定的算法是不能使用的.

定义 1.1 一个算法如果输入数据有误差,而在计算过程中舍入误差不增长,则称此算法是**稳定**的;否则称此算法为**不稳定**的.

在例 1.5 中算法(B)是稳定的,而算法(A)是不稳定的.

1.3.2 病态问题与条件数

定义 1.2 对一个数值问题本身如果输入数据有微小扰动(即相对误差),引起输出数据

(即问题解)相对误差很大,则称此数值问题为**病态问题**. 输出数据的相对误差与输入数据的相对误差的比值称为此数值问题的**条件数**.

例如,计算函数值 $f(x)$ 时,若 x 有扰动 $\Delta x = x - x^*$,其相对误差为 $\dfrac{\Delta x}{x}$,函数值 $f(x^*)$ 的相对误差为 $\dfrac{f(x)-f(x^*)}{f(x)}$. 比值

$$\left|\frac{f(x)-f(x^*)}{f(x)}\right| \Big/ \left|\frac{\Delta x}{x}\right| \approx \left|\frac{xf'(x)}{f(x)}\right| = C_p \tag{1.6}$$

称为计算函数值问题的条件数. 一个数值问题的条件数可以视为处理此数值问题的过程中,相对误差的放大(缩小)倍数. 自变量相对误差一般不会太大,如果条件数 C_p 很大,将引起函数值相对误差很大,出现这种情况的问题就是病态问题.

例如,取 $f(x)=x^n$,则有 $C_p=n$,它表示相对误差可能放大 n 倍. 如 $n=10$,有 $f(1)=1$,$f(1.02)\approx 1.22$,若取 $x=1$,$x^*=1.02$ 自变量相对误差为 2%,函数值相对误差为 22%,这时问题可以认为是病态的. 一般情况下,条件数 $C_p \geqslant 10$ 就认为是病态,C_p 越大病态越严重.

例 1.6 求解线性方程组

$$\begin{cases} x + \alpha y = 1, \\ \alpha x + y = 0. \end{cases} \tag{1.7}$$

解 当 $\alpha=1$ 时,系数行列式为零,方程无解,但当 $\alpha \neq 1$ 时解为 $x=\dfrac{1}{1-\alpha^2}$,$y=-\dfrac{\alpha}{1-\alpha^2}$. 当 $\alpha \approx 1$ 时,若输入数据 α 有微小扰动(误差),则解的误差很大. 例如,取 $\alpha=0.99$,则解 $x \approx 50.25$;如果 α 有误差 0.001,取 $\alpha^*=0.991$,则解 $x^* \approx 55.81$,误差 $|x^*-x| \approx 5.56$ 很大,表明此时线性方程组(1.7)是病态的. 实际上,由 $x=\dfrac{1}{1-\alpha^2}$ 是 α 的函数,利用(1.6)式可求得

$$C_p = \left|\frac{\alpha x'(\alpha)}{x(\alpha)}\right| = \left|\frac{2\alpha^2}{1-\alpha^2}\right|.$$

当 $\alpha=0.99$ 时 $C_p \approx 100$,表明条件数很大,故问题是病态的.

注意病态问题不是计算方法引起的,是数值问题自身固有的,因此,对数值问题首先要分清问题是否病态,对病态问题就必须采取相应的特殊方法以减少误差危害.

1.3.3 避免误差危害

数值计算中通常不采用数值不稳定算法,在设计算法时还应尽量避免误差危害,防止有效数字损失,通常要避免两个相近数相减和用绝对值很小的数做除数(用绝对值很大的数做乘数),另外,还要注意运算次序和减少运算次数.下面举例说明.

例 1.7 求 $x^2-16x+1=0$ 的小正根.

解 $x_1=8+\sqrt{63}$,$x_2=8-\sqrt{63}$. 如果取 $\sqrt{63}=7.94$,即 $\sqrt{63}$ 的 3 位有效数字近似,那么 $x_2 \approx 8-7.94=0.06=x_2^*$,$x_2^*$ 只有 1 位有效数字. 若改用

$$x_2 = 8-\sqrt{63} = \frac{1}{8+\sqrt{63}} \approx \frac{1}{15.94} \approx 0.0627,$$

具有 3 位有效数字.

例 1.8 计算 $A=10^7(1-\cos 2^\circ)$(用四位数学用表).