数值分析

SHUZHI FENXI

主　编　何光辉　胡小兵　董海云
主　审　王开荣

重庆大学出版社
http://www.cqup.com.cn

工程数学系列教材

数值分析

主　编　何光辉　胡小兵　董海云

主　审　王开荣

重庆大学出版社

内 容 提 要

本书是为高校专业硕士和本科生《数值分析》课程编写的教材。书中系统地介绍了数值分析的基本概念,常用算法及有关的理论分析和应用。全书共分9章,第1章是绪论,介绍数值分析中的基本概念;第2章至第9章包含了数值分析中的基本问题,如线性方程组的数值解法,矩阵特征值和特征向量的数值解法,非线性方程及方程组的数值解法,插值方法,数据拟合和函数逼近,数值积分,数值微分以及常微分方程初值问题的数值解法等;在每个章节我们通过一个案例引入即将讲解的内容,加强了本课程在实际工程中的联系。在每个章节后面对部分例题介绍了用Matlab软件求解过程和源程序。教材还通过表格的形式对每章中的内容进行了系统的总结,方便读者总结和概括本章内容。最后各章都给出典型例题并配有一定数量的习题,书后给出了习题答案和提示.。

本书基本概念叙述清晰,语言通俗易懂,注重算法的实际应用和上机实践,可作为专业硕士和本科生的数值分析课程的教科书,还可作为大学本科及硕士生的教学参考书,也可供工程技术人员参考使用。

图书在版编目(CIP)数据

数值分析/何光辉主编. —重庆:重庆大学出版社,2015.8

ISBN 978-7-5624-9342-6

Ⅰ.①数… Ⅱ.①何… Ⅲ.①数值分析 Ⅳ.①0241

中国版本图书馆CIP数据核字(2015)第169829号

数值分析

主　编　何光辉　胡小兵　董海云

主　审　王开荣

责任编辑:文　鹏　　版式设计:周　立

责任校对:关德强　　责任印制:赵　晟

*

重庆大学出版社出版发行

出版人:邓晓益

社址:重庆市沙坪坝区大学城西路21号

邮编:401331

电话:(023)88617190　88617185(中小学)

传真:(023)88617186　88617166

网址:http://www.cqup.com.cn

邮箱:fxk@cqup.com.cn(营销中心)

全国新华书店经销

重庆鹏程印务有限公司印刷

*

开本:787×1092　1/16　印张:14.75　字数:368千

2015年8月第1版　2015年8月第1次印刷

印数:1—3 000

ISBN 978-7-5624-9342-6　定价:29.50元

前　言

科学计算已经与理论证明、科学实验并列成为三种科学研究方法之一。近年来随着计算机的迅速普及以及工程中需要处理的信息快速增长，掌握和应用科学计算方法或者数值分析方法已经是理工科研究生和本科生的基本要求。目前在工科研究生和专业硕士以及本科阶段选修《数值分析》课程的专业和学生占很大比例，这充分说明了科学计算已经成为各种工程技术中重要的研究手段和工具。

本书力求适应工科类专业硕士学习的特点，减少理论证明的过程，加强了实际案例的应用讲解。为了弥补研究生课程没有上机实践的缺点，本书中增加了上机实践的源代码，希望学生理解数学知识在工程应用中的过程，明确学习的目的，增强学习主动性。

因此本书编写时注重了以下原则：一是保持课程体系的完整性，二是增强本课程的实用性，三是加深内容的可读性。为此，我们在系统介绍基础理论的同时，省略了一些繁琐艰深的证明过程，而主要侧重于算法叙述和算例分析。行文时注重通俗易懂，对专业术语尽量作通俗的解释，特别是避免完全用术语解释术语，以增强本书的可读性。同时为常用的算法给出了 Matlab 出程序的源代码，方便读者编程运行进行验证。

学习本书必需的数学基础是微积分、线性代数和常微分方程，这是一般工科专业硕士和大学生都具备的。为便于自学，各章后均附有习题，书后有习题参考答案和提示，师生可结合安排习题练习。在计算习题计算量较大时，可考虑使用程序计算。

全书共分 9 章，其中第 1 章至第 5 章由何光辉老师执笔；第 6 章、7 章由胡小兵老师编写，第 8、9 章由董海云老师完成，全书由王开荣老师审核。全书设计讲授时数为 40 学时。

由于我们的经验和水平有限，对教材中疏忽和谬误之处，敬请读者指正赐教，以期修订时改进完善。

作者

2015.1

目 录

第1章 绪 论

数值分析是数学学科的一个分支，是一门与计算机科学密切结合的实用性很强的数学课程，也是科学计算的基础。数值分析是以各类数学问题的数值解法作为研究对象，并结合现代计算机科学与技术，为解决科学与工程中遇到的各类数学问题提供基本的算法。

数值分析课程研究内容包括常见的基本数学问题的数值解法，包含了数值代数（线性方程组的解法、矩阵特征值计算等）、非线性方程的解法、数值逼近、数值微分与数值积分、常微分方程的数值解法等。它的基本理论和研究方法建立在数学理论基础之上，研究对象是数学问题，因此它是数学的分支之一。

近年来，个人计算机的飞速发展使得人们可以获得强大的计算能力，使得数值分析的方法发展和使用异常迅速。数值分析是强大的问题求解工具，它能处理大规模方程组、非线性系统和复杂的几何问题。在工程中，这些问题很常见，用解析方法对其求解几乎是不可能的。因此，数值分析增强了人们求解问题的能力。

1.1 算 法

算法是对问题求解过程的一种描述，是为解决一个或一类问题给出的一个确定的、有限长的操作序列。严格说来，一个算法必须满足以下五个重要特性：

（1）有穷性

对于任意一组合法的输入值，在执行有穷步骤之后一定能结束。这里有两重意思，即算法中的操作步骤为有限个，且每个步骤都能在有限时间内完成。

（2）确定性

对于每种情况下所应执行的操作，在算法中都有确切的规定，使算法的执行者或阅读者都能明确其含义及如何执行。并且在任何条件下，算法都只有一条执行路径。确定性表现在对算法中每一步的描述都没有二义性，只要输入相同，初始状态相同，则无论执行多少遍，所得结果都应该相同。

（3）可行性

算法中的所有操作都必须足够基本，都可以通过已经实现的基本操作运算有限次实现之。可行性指的是，序列中的每个操作都是可以简单完成的，其本身不存在算法问题，例如，“求 x 和 y 的公因子”就不够基本。

（4）有输入

作为算法加工对象的量值，通常体现为算法中的一组变量。但有些算法的字面上可以没有输入，实际上已被嵌入算法之中。

(5)有输出

它是一组与“输入”有确定关系的量值,是算法进行信息加工后得到的结果,这种确定关系即为算法的功能。

1.1.1 算法常用描述形式

(1)用数学公式和文字说明描述

这种方式符合人们的理解习惯,和算法的推证相衔接,易于学习接受,但不方便转换成程序语言。

(2)用框图描述

这种方式描述计算过程流向清楚,易于编制程序,但初学者有一个习惯过程。此外,框图描述格式不很统一,详略难以掌握。

(3)伪代码

它是表述算法的一种通用语言,有特定的表述程序和语句。它独立于计算机的硬件和软件系统,但它可以很容易地转换成某种实用的计算机高级语言,同时也具有一定的可读性。

(4)算法程序

算法程序即用计算机语言描述的算法。它是面向计算机的算法,计算机可直接运行。它可以是印刷文本,也可以是存储器上存储的文件,它们常常组装成算法软件包。我们以后讨论的算法,通常都有现成的程序文本和软件可以利用。一个合格的计算工作者,应当能熟练地应用这些已有的软件工具。但从学习算法的角度看,这种描述方式并不十分有利。

由于数值分析课程的特点是公式比较多,因此本教材主要采用数学公式和文字说明的描述方式。读者如果需要编写相关的程序,可以在理解算法的基础上自己编写程序。

1.1.2 数值型算法的基本特点

数值型算法的执行总是与特定的工具有关,而每一种计算工具的有效数位总是一定的。因此相对于普通算法而言,数值分析问题在实际过程中参与运算的数值基本都是近似的。为准确而全面地描述整个问题的解决过程,数值型算法常具有如下特点:

1)无穷过程的截断

例 1.1 计算 $e^x, x \in [0,0.5]$。

解 据 Taylor 公式:

$$e^x = 1 + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{(n)!} + \cdots \tag{1.1}$$

这是一个无穷级数,我们只能在适当的地方“截断”,使计算量不太大,而精度又能满足要求。

如取 $n=5$,计算 $e^{0.25} \approx 1 + \frac{0.25^2}{2!} + \cdots + \frac{0.25^5}{5!} = 0.284\ 025\ 065\ 104\ 167$,据 Taylor 余项公式,它的误差应为

$$R = \frac{\xi^6}{6!}, \quad \xi \in [0,0.5] \tag{1.2}$$

$$|R| \leqslant \frac{(0.5)^6}{720} = 2.170\ 13 \times 10^{-5}$$

可见结果相当精确。

2)连续过程的离散化

例1.2 计算积分值 $I = \int_0^1 (x^2 + 1)\mathrm{d}x$。

解 如图1.1所示,将[0,1]分为4等份,分别计算4个小曲边梯形面积的近似值,然后加起来作为积分的近似值。记被积函数为 $f(x)$,即

$$f(x) = x^2 + 1$$

取 $h = \frac{1}{4}$,有 $x_i = ih, i = 0,1,2,3,4, T_i = \frac{f(x_i) + f(x_{i+1})}{2} h$。

所以有 $I \approx \sum_{i=0}^{3} T_i = 1.343\ 75$,与精确1.333 33比较,可知结果不够精确,如进一步细分区间,精度可以提高。

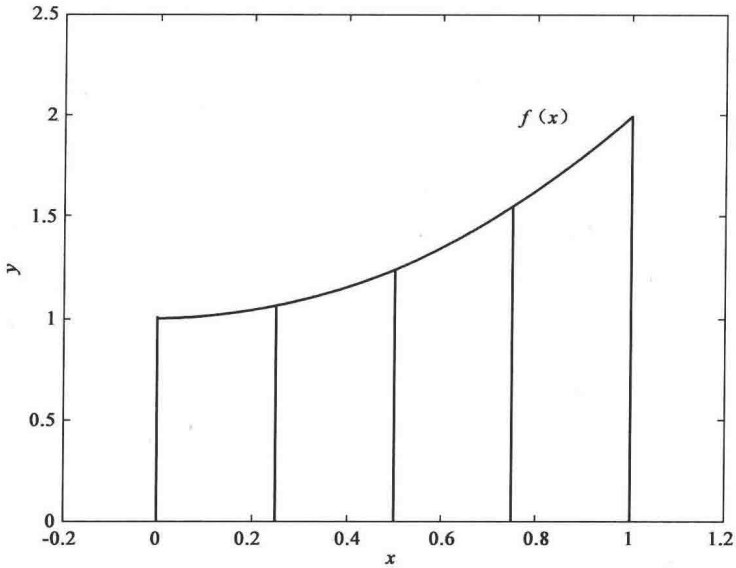

图1.1

3)迭代计算

迭代是指某一简单算法的多次重复,后一次使用前一次的结果。这种形式易于在计算程序中实现,在程序中表现为"循环"过程。例如求最大公约数:求两个数 $a, b(b \neq 0)$ 的最大公约数,可以利用辗转相除法来完成。其具体步骤为:

①求 $\frac{a}{b}$ 的余数 p。

②如果 $p \neq 0$,令 $a = b, b = p$。

③重复步骤①②直到 $p = 0$,最终得到的 a 即为最大公约数。

例1.3 求(319,377)的最大公约数。

解 377 ÷ 319 = 1(余58),(377,319) = (319,58);

$319 \div 58 = 5$(余 29),(319,58) = (58,29);

$58 \div 29 = 2$(余 0),(58,29) = 29;

最终得到(319,377) = 29。

例 1.4 不用开平方计算 $\sqrt{a}(a > 0)$ 的值。

解 假定 x_0 是 $\sqrt{a}$ 的一个近似值,$x_0 > 0$,则 $\frac{a}{x_0}$ 也是 $\sqrt{a}$ 的一个近似值,且 x_0 和 $\frac{a}{x_0}$ 两个近似值必有一个大于 $\sqrt{a}$,另一个小于 $\sqrt{a}$。可以设想它们的平均值应为 $\sqrt{a}$ 的更好的近似值,于是设计一种算法:

$$x_{k+1} = \frac{1}{2}\left(x_k + \frac{a}{x_k}\right), \quad k = 0,1,2,\cdots \tag{1.3}$$

如计算 $\sqrt{5}$,取 $x_0 = 3$,有 $x_{k+1} = \frac{1}{2}\left(x_k + \frac{5}{x_k}\right), \quad k = 0,1,2,\cdots$

计算得到以下结果:

$$\begin{aligned} x_0 &= 3 \\ x_1 &= 2.333\ 333\ 333\ 333\ 334 \\ x_2 &= 2.238\ 095\ 238\ 095\ 238 \\ x_3 &= 2.236\ 068\ 895\ 643\ 363 \\ &\vdots \end{aligned}$$

可见此法收敛速度很快,只迭代 3 次就得到 6 位精确数字。

迭代法应用时要需要考虑迭代格式是否收敛、收敛条件及收敛速度等问题,这将在第 5 章进一步讨论。

1.1.3 算法的复杂度

通常有两种衡量算法效率的方法:事后统计法和事前分析估算法。相比之下,前者的缺点是必须在计算机上实地运行程序,容易有其他因素掩盖算法本质;而后者的优点是可以预先比较各种算法,以便均衡利弊而从中选优。

1)算法的时间复杂度

(1)估算算法的时间效率

和算法执行时间相关的因素有:①算法所用“策略”;②算法所解问题的“规模”;③编程所用“语言”;④“编译”的质量;⑤执行算法的计算机的“速度”。显然,后三条受到计算机硬件和软件的制约。既然是“估算”,我们仅需考虑前两条。

一个算法的“运行工作量”通常是随问题规模的增长而增长,因此比较不同算法的优劣主要应该以其“增长的趋势”为准则。假如,随着问题规模 n 的增长,算法执行时间的增长率 $T(n)$ 和 $f(n)$ 的增长率相同,则可记作 $T(n) = O(f(n))$。其中,“O”的数学含义是:若存在两个常量 C 和 n_0,当 $n > n_0$ 时,$|T(n)| \leqslant C|f(n)|$。

(2)估算算法的时间复杂度

任何一个算法都是由一个“控制结构”和若干“原操作”组成的,因此,一个算法的执行时间可以看成是所有原操作的执行时间之和。

$$\text{算法运行时间} = \sum \text{原操作}(i)\text{的执行次数} \times \sum \text{原操作的执行时间}$$

“原操作”指的是固有数据类型的操作，显然每个原操作的执行时间和算法无关，相对于问题的规模是常量。同时，由于算法的时间复杂度只是算法执行时间增长率的量度，因此只需要考虑在算法中“起主要作用”的原操作即可。这种原操作称为“基本操作”，它的重复执行次数和算法的执行时间成正比，从而算法的执行时间与所有原操作的执行次数之和成正比。

从算法中选取一种对于所研究的问题来说是基本操作的原操作，以该基本操作在算法中重复执行的次数作为算法时间复杂度的依据。这种衡量效率的办法所得出的不是时间量，而是一种增长趋势的量度。它与软硬件环境无关，只暴露算法本身执行效率的优劣。

例1.5 两个 $n \times n$ 的矩阵相乘。其中矩阵的“阶” n 为问题的规模。

算法：

```
void Mult_matrix (int c[][], int a[][], int b[][], int n)
{
    // a、b 和 c 均为 n 阶方阵，且 c 是 a 和 b 的乘积
    for (i=1; i<=n; ++i)
    for (j=1; j<=n; ++j) {
      c[i,j] = 0;
      for (k=1; k<=n; ++k)
        c[i,j] += a[i,k] * b[k,j];
    }
}// Mult_matrix
```

容易看出，算法中的控制结构是三重循环，每一重循环的次数是 n 。原操作有赋值运算、加法运算和乘法运算。显然，在三重循环之内的“乘法”是基本操作，它的重复执行次数是 n^3 ，即算法的时间复杂度为 $O(n^3)$ 。

2）算法的空间复杂度

算法的存储量指的是算法执行过程中所需的最大存储空间。算法执行期间所需要的存储量应该包括以下三部分：①输入数据所占空间；②程序本身所占空间；③）辅助变量所占空间。

类似于算法的时间复杂度，通常以算法的空间复杂度作为算法所需存储空间的量度。定义算法空间复杂度为 $S(n) = O(g(n))$ ，表示随着问题规模 n 的增大，算法运行所需辅助存储量的增长率与 $g(n)$ 的增长率相同。

程序代码本身所占空间对不同算法通常不会有数量级之差别，因此在比较算法时可以不加考虑；算法的输入数据量和问题规模有关，若输入数据所占空间只取决于问题本身，和算法无关，则在比较算法时也可以不加考虑。由此只需要分析除输入和程序之外的额外空间。

和算法时间复杂度的考虑类似，若算法所需存储量依赖于特定的输入，则通常按最坏情况考虑。

1.2 误 差

计算机科学的发展,提供了数据处理的高速度和高精度的计算工具,使得许多复杂的数值分析问题得到了较好的解决。但计算机与任何计算工具相同,它所处理的数值型的问题在任何环节中都是近似的,因而数值分析中误差总是不可避免存在的。也就是说,在数值分析方法中,绝大多数情况下不存在绝对的精确。我们关注的是如何估计误差,并将误差控制在可以接受的范围内。因此,在数值分析中误差分析是十分重要的。

1.2.1 误差的来源

在运用数学方法解决实际问题的过程中,每一步都可能带来误差。

(1)模型误差

在建立数学模型时,往往要忽视很多次要因素,把模型"简单化"、"理想化"。这时模型就与真实背景有了差距,即带入了误差。

(2)测量误差

数学模型中的已知参数多数是通过测量得到,而测量过程受工具、方法、观察者的主观因素、不可预料的随机干扰等影响必然带入误差。

(3)截断误差

数学模型常难于直接求解,往往要近似替代,简化为易于求解的问题,这种简化带入的误差称为方法误差或截断误差。

(4)舍入误差

计算机只能处理有限数位的小数运算,初始参数或中间结果都必须进行四舍五入运算,这必然产生舍入误差。

在数值分析课程中不分析讨论模型误差。截断误差是数值分析课程的主要讨论对象,它往往是计算中误差的主要部分,在讲到各种算法时,通过数学方法可推导出截断误差限的公式,如式(1.2);舍入误差的产生往往带有很大的随机性,讨论比较困难,在问题本身呈病态或算法稳定性不好时,它可能成为计算中误差的主要部分;至于测量误差,我们把它作为初始的舍入误差看待。

详尽的误差分析是困难的,有时是不可能的。数值分析中主要讨论截断误差及舍入误差。当我们发现计算结果与实际不符时,应当能诊断出误差的来源,并采取相应的措施加以改进,直至建议对模型进行修改。

1.2.2 误差的基本概念

1)误差与误差限

定义 1.1 设 x^* 是准确值,x 是它的一个近似值,称

$$e = x - x^* \tag{1.4}$$

为近似值 x 的绝对误差,简称误差。误差是有量纲的量,量纲同 x,它可正可负。

误差一般无法准确计算,只能根据测量或计算情况估计出其绝对值的一个上限,这个上限称为近似值 x 的误差限,记为 ε。即

$$|x - x^*| \leqslant \varepsilon \tag{1.5}$$

其意义是：$-\varepsilon \leqslant x - x^* \leqslant \varepsilon$，在工程中常记为：$x^* = x \pm \varepsilon$。如 $P = 10.2 \pm 0.05$ kw，$R = 1\,500 \pm 10\ \Omega$ 等。

2）相对误差与相对误差限

误差不能完全刻画近似值的精度。如测量百米跑道产生 10 cm 的误差与测量一个地球月球距离产生 100 km 的误差，我们不能简单地认为前者更精确，还应考虑被测值的大小。下面给出定义：

定义 1.2 误差与精确值的比值

$$e_r = \frac{e}{x^*} = \frac{x - x^*}{x^*} \tag{1.6}$$

称为 x 的相对误差。相对误差是无量纲的量，常用百分比表示，它也可正可负。

相对误差也常不能准确计算，而是用相对误差限来估计。相对误差限：

$$\varepsilon_r = \frac{\varepsilon}{|x^*|} \geqslant \frac{|x - x^*|}{|x^*|} = |e_r| \tag{1.7}$$

实际上，由于 x^* 不知道，用式(1.7)无法确定 ε_r，常用 x 代 x^* 作分母，以后就用 $\varepsilon_r = \frac{\varepsilon}{|x|}$ 表示相对误差限。

例 1.6 在刚才测量的例子中，若测得跑道长为 100 ± 0.1 m，月地距离为 $384\,400 \pm 100$ km，则

$$\varepsilon_r^{(1)} = \frac{0.1}{100} = 0.1\%$$

$$\varepsilon_r^{(2)} = \frac{100}{384\,400} = 0.026\%$$

显然前者比后者相对误差大。

1.2.3 有效数字

定义 1.3 设 x 的误差限 ε 是它某一数位的半个单位，我们就说 x 准确到该位，从这一位起直到前面第一个非零数字为止的所有数字称为 x 的有效数字。

如：$x = \pm 0.a_1a_2\cdots a_n \times 10^m$，其中 $a_1, a_2, \cdots, a_n$ 是 0 ~ 9 的整数，且 $a_1 \neq 0$。如 $|e| = |x - x^*| \leqslant \varepsilon = 0.5 \times 10^{m-l}$，$1 \leqslant l \leqslant n$，则称 x 有 l 位有效数字。有效数字的位数称为有效数位。

以上定义可以用于在知道精确值 x^* 时去确定 x 的有效数位。如：$\pi = 3.141\,592\,65\cdots$ 则 3.14 和 3.141 6 分别有 3 位和 5 位有效数字，而 3.143 相对于 π 也只能有 3 位有效数字。

但在更多的情况下，我们不知道准确值 x^*。如果认为计算结果的各数位可靠，将它四舍五入到某一位，这时从这一位起到前面第一个非零数字共 l 位，由于四舍五入的原因，它与计算结果之差必不超过该位的半个单位。因此，从这一位起到前面第一个非零数字都是有效数字，我们习惯上说将计算结果保留 l 位有效数字。

如计算机上得到方程 $x^3 - x + 1 = 0$ 的一个正根为 1.324 72，保留 4 位有效数字的结果为 1.325，保留 5 位有效数字的结果为 1.324 7。

相对误差与有效数位的关系十分密切。定性地讲，相对误差越小，有效数位越多，反之亦正确。定量地讲，有如下两个定理：

定理 1.1　设近似值 $x = \pm 0.a_1a_2\cdots a_n \times 10^m$，有 n 位有效数字，则其相对误差限

$$\varepsilon_r \leqslant \frac{1}{2a_1} \times 10^{-n+1} \tag{1.8}$$

此定理的证明不难，可作为习题完成。

定理 1.2　设近似值 $x = \pm 0.a_1a_2\cdots a_n\cdots \times 10^m$ 的相对误差限不大于

$$\varepsilon_r \leqslant \frac{1}{2(a_1+1)} \times 10^{-n+1} \tag{1.9}$$

则它至少有 n 位有效数字。

证　$|x| \leqslant (a_1+1) \times 10^{m-1}$

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

由定义 1.3 知 x 有 n 位有效数字。

对有效数字的观察比估计相对误差容易得多，如监视到有效数字在算法中某一步突然变少，便意味着相对误差在这一步的突然扩大，这就是计算出问题的地方。

例 1.7　计算 $\frac{1}{1\ 359} - \frac{1}{1\ 360}$，视已知数为精确值，用 5 位浮点数计算。

解　原式 $= 0.735\ 84 \times 10^{-3} - 0.735\ 29 \times 10^{-3} = 0.55 \times 10^{-6}$。

结果只剩 2 位有效数字，有效数字大量损失，造成相对误差的扩大。

若通分后再计算：

$$\text{原式} = \frac{1}{1\ 359 \times 1\ 360} = \frac{1}{1.848\ 2 \times 10^7} = 0.541\ 06 \times 10^{-6}$$

就得到 5 位有效数字的结果。

1.3　数值运算时误差的传播

当参与运算的数值带有误差时，结果也必然带有误差，但我们在计算过程当中通过注意观察结果的误差与原始误差相比是否会扩大，了解并分析扩大的原因，可以有效地避免误差的迅速扩大，从而提高计算的精度。

1.3.1　一元函数计算的误差传播

设 x 是 x^* 的近似值，则结果误差

$$e(f(x)) = f(x) - f(x^*)$$

用 Taylor 展式分析

$$f(x^*) = f(x) + f'(x)(x^* - x) + \frac{f''(\xi)}{2}(x^* - x)^2$$

$$e(f(x)) = f'(x)(x - x^*) - \frac{f''(\xi)}{2}(x^* - x)^2$$

$$|e(f(x))| \leqslant \varepsilon(f(x)) \leqslant |f'(x)|\varepsilon(x) + \left|\frac{f''(\xi)}{2}\right|\varepsilon^2(x)$$

忽略第二项高阶无穷小之后，可得函数 $f(x)$ 的误差限估计式

$$\varepsilon(f(x)) \approx |f'(x)|\varepsilon(x) \tag{1.10}$$

1.3.2 多元函数计算时的误差传播

若 $x_1^*, x_2^*, \cdots, x_n^*$ 的近似值分别是 $x_1, x_2, \cdots, x_n$，则多元函数的准确值为 $A^* = f(x_1^*, x_2^*, \cdots, x_n^*)$，近似值为 $A = f(x_1, x_2, \cdots, x_n)$。

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

$$|A - A^*| = |f(x_1, x_2, \cdots, x_n) - f(x_1^*, x_2^*, \cdots, x_n^*)| \leqslant \sum_{k=1}^{n} \left| \frac{\partial f(x_1, x_2, \cdots, x_n)}{\partial x_k} \right| |x - x^*| + O((\Delta x)^2)$$

其中，$\Delta x = \max\limits_{1 \leqslant k \leqslant n} |x_k - x^*|$。

略去高阶项后，有

$$\varepsilon(A) \approx \sum_{k=1}^{n} \left| \frac{\partial f(x_1, x_2, \cdots, x_n)}{\partial x_k} \right| \varepsilon(x_k) \tag{1.11}$$

当函数是二元函数时，公式成为

$$\varepsilon(f(x,y)) \approx \left| \frac{\partial f(x,y)}{\partial x} \right| \varepsilon(x) + \left| \frac{\partial f(x,y)}{\partial y} \right| \varepsilon(y) \tag{1.12}$$

1.3.3 四则运算中误差的传播

四则运算可以看作是二元函数运算，按公式(1.12)易得近似数作四则运算后的误差限公式：

$$\varepsilon(x \pm y) = \varepsilon(x) + \varepsilon(y) \tag{1.13}$$

$$\varepsilon(x \cdot y) \approx |y|\varepsilon(x) + |x|\varepsilon(y) \tag{1.14}$$

$$\varepsilon\left(\frac{x}{y}\right) \approx \frac{|y|\varepsilon(x) + |x|\varepsilon(y)}{y^2} \quad (y \neq 0) \tag{1.15}$$

其中，公式(1.13)取等号，是因为作为多元函数，加减运算是一次函数，Taylor 展开式没有二次余项。

例 1.8 若电压 $V = 220 \pm 5$ V，电阻 $R = 300 \pm 10\ \Omega$，求电流 I 并计算其误差限及相对误差限。

解 $$I \approx \frac{220}{300} = 0.733\ 3\ (\text{A})$$

$$\varepsilon(I) = \frac{|V|\varepsilon(R) + |R|\varepsilon(V)}{R^2} = \frac{220 \times 10 + 300 \times 5}{90\ 000} = 0.041\ 1$$

$$I = 0.733\ 3 \pm 0.041\ 1(\text{A}) \qquad \varepsilon_r(I) = \frac{0.041\ 1}{0.733\ 3} = 0.056 = 5.6\%$$

1.3.4 设计算法时应注意的问题

1)避免两个相近数相减

由公式(1.15)有 $\varepsilon(x - y) = \varepsilon(x) + \varepsilon(y)$，可推出

$$\varepsilon_r(x-y)=\frac{\varepsilon(x-y)}{|x-y|}=\frac{|x|}{|x-y|}\times\frac{\varepsilon(x)}{|x|}+\frac{|y|}{|x-y|}\times\frac{\varepsilon(y)}{|y|}$$
$$=\frac{|x|}{|x-y|}\times\varepsilon_r(x)+\frac{|y|}{|x-y|}\times\varepsilon_r(y)$$

当 x,y 十分相近时，$|x-y|$ 接近零，$\frac{|x|}{|x-y|}$ 和 $\frac{|y|}{|x-y|}$ 将很大，所以 $\varepsilon_r(x-y)$ 将比 $\varepsilon_r(x)$ 或 $\varepsilon_r(y)$ 大很多，即相对误差将显著扩大。

从直观上看，相近数相减会造成有效数位的减少，例 1.7 就是一个例子。有时，通过改变算法可以避免相近数相减。

例 1.9 解方程 $x^2-18x+1=0$（用 4 位浮点计算）。

解 用公式解法：

$$x_1=\frac{18+\sqrt{18^2-4}}{2}=9+\sqrt{80}=17.94,\ x_2=9-\sqrt{80}=9.000-8.944=0.056。$$

因为相近数相减，第二个根只有两位有效数字，精度较差。若第二个根改为用韦达定理计算

$$x_2=\frac{1}{x_1}=\frac{1}{17.94}=0.055\ 74,$$

可以得到较好的结果。

又如 $\sqrt{x+1}-\sqrt{x}(x\gg1)$ 可改为 $\frac{1}{\sqrt{x+1}+\sqrt{x}}$，$1-\cos x(|x|\ll1)$ 可改为 $2\sin^2\left(\frac{x}{2}\right)$ 等，都可以得到比直接计算好的结果。

2）避免除法中除数的数量级远小于被除数

由公式(1.17)

$$\varepsilon\left(\frac{x}{y}\right)=\frac{|y|\varepsilon(x)+|x|\varepsilon(y)}{y^2}\approx\frac{|x|}{y^2}\varepsilon(y)+\frac{1}{|y|}\varepsilon(x)$$

若 $|y|\ll|x|$，则 $\frac{|x|}{y^2}\gg1$，这时 $\varepsilon\left(\frac{x}{y}\right)$ 将比 $\varepsilon(y)$ 扩大很多。

3）合理安排运算顺序

在以前学习的公理系统中，加法满足交换律，即 $a+b+c\equiv a+c+b$。但在数值分析中就不一定成立了，如果计算机只有 4 位字长，而假设 $a=10^{20}$，$b=2$，$c=-10^{20}$。如果按照顺序计算得到的结果是 0，而改写成先计算 $a+c$ 后结果为 2。由此可知，在数值分析中，加法的交换律和结合律可能不成立，这是在大规模数据处理时应注意的问题。

4）注意运算步骤的简化

减少算术运算的次数，除可以减少运算时间、提高运算效率外，还有一个重要作用就是减少误差的累积效应。同时，参与运算的数字的精度应尽量保持一致，否则那些较高精度的量的精度没有太大意义。

1.3.5 病态问题数值算法的稳定性

在某一数学问题的计算过程中，舍入误差是否增长直接影响计算结果的可靠性。这里可能是数学问题本身性态不好，也可能是选择的算法出了问题。