内 容 简 介

本书内容丰富且颇具特色.

本书综述了数值分析领域的诸多内容,包括配置多项式、有限差分、阶乘多项式、求和法、Newton 公式、算子与配置多项式、样条、密切多项式、Taylor 多项式、插值、数值微分、数值积分、和与级数、差分方程、微分方程、最小二乘多项式逼近、极小化极大多项式逼近、有理函数逼近、三角逼近、非线性代数、线性方程组、线性规划、边值问题、Monte Carlo 方法等内容.

本书的特色主要表现在利用例题及大量详细的题解来透彻地阐明所述内容的内涵,同时附有大量的补充题以使读者进一步巩固和深化从书中获得的数值分析知识.

本书可作为理工科大学生、电大、函授生学习数值分析的教科书,更适合用作理论性较强的数值分析教程的参考书,也可作为自学数值分析课程者的读本.

Francis Scheid: Schaum's Outline of Theory and Problems of Numerical Analysis, Second Edition

ISBN:0-07-05522L-5

图字:01-2001-2115 号

图书在版编目(CIP)数据

数值分析:第 2 版/(美)施依德(Scheid, F.)著;罗亮生,包雪松,王国英译,
- 北京:科学出版社,2002.1

(全美经典学习指导系列)

书名原文:Numerical Analysis

ISBN 7-03-009976-1

Ⅰ.数… Ⅱ.①施… ②罗… ③包… ④王… Ⅲ.计算方法-高等学校-教材 Ⅳ.0241

中国版本图书馆 CIP 数据核字(2001)第 093162 号

科学出版社 出版

北京东黄城根北街16号

邮政编码:100717

http://www.sciencep.com

新蕾印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2002 年 1 月第 一 版 开本:A4(890×1240)

2002 年 1 月第一次印刷 印张:25

印数:1—5 000 字数:723 000

定价:35.00 元

(如有印装质量问题,我社负责调换〈北燕〉)

目　录

第一章　数值分析是什么

算法

数值分析的目的是仅用简单的算术运算解复杂的数值问题，并开发和评估由给出的数据计算数值结果的方法.这些计算方法被称为算法.

我们将致力于对算法的研究.迄今为止，仍未找到某些问题的满意算法；另一方面，当存在着几种不同算法时，我们必须从中作出选择.选择这个而非那个算法时存在种种理由，但有两个明显的标准：速度与精度.速度快显然是一种优势，虽然就适当规模的问题而言，这一优势因计算机的能力而被削弱殆尽.但对于大规模的问题，速度仍是重要的因素，速度慢的算法由于不实用会被淘汰.然而，在其他条件均相同时，速度较快的方法定会被首肯.

例 1.1　求出 2 的平方根，直至具有 4 位十进制小数.

仅用 4 种基本的算术运算，存在着不只一个算法.无疑，算法

$$x_1 = 1, \quad x_{n+1} = \frac{1}{2}\left(x_n + \frac{2}{x_n}\right)$$

是令人满意的.据此，由少数几步心算就能很快得到

$$x_2 = \frac{3}{2}, \quad x_3 = \frac{17}{12}, \quad x_4 = \frac{1}{2}\left(\frac{17}{12} + \frac{24}{17}\right),$$

或四舍五入到 4 位十进制小数，

$$x_2 = 1.500, \quad x_3 = 1.4167, \quad x_4 = 1.4142.$$

最后的 x_4 所有的 4 位小数均是正确的.这个数值算法具有悠久的历史，并且，它将在第 25 章中作为方程求根的特殊情况而再次遇到.

误差

数值计算的乐观主义者会问计算结果有多精确；而数值计算的悲观主义者会问结果中已引入了多少误差.显然，这两个问题是相同的.给出的数据，很少是精确的，因为它通常源于测量过程.从而，输入的信息中或许存在着误差.并且，算法本身通常也会带来误差，或许是不可避免的舍入误差.这样，输出的信息中将包含出自这两种来源的误差.

例 1.2　假设数 0.1492 准确到给出的 4 位十进制小数，换言之，它是处在 0.14915 与 0.14925 之间的一个真值的近似，那么，其误差至多为第 5 位小数的 5 个单位或第 4 位小数的半个单位.在此情况下，这个近似值被称为具有 4 位有效数字.类似地，倘若 14.92 的误差不超过 0.005，则它具有 2 位准确的小数和 4 位有效数字.

例 1.3　当数 0.10664 缩写成 0.1066 时，被称为四舍五入成 4 位小数.而 0.10666 将被四舍五入成 0.1067.若给出的数字是准确的，则在这两种情况下，由舍入造成的误差不大于 0.00005.前一个例子是向下“舍”，后一个例子是向上“入”.像 0.10665 这种边界状况，通常舍入到最接近的偶数数字，这里舍到 0.1066.这避开了长期以来不知向下舍还是向上入的尴尬.

例 1.4　1.492 乘以 1.066，积是 1.590472.计算机按固定的“字长”工作，所有的数均剪裁成这个长度.假设有一台虚构的 4 位数字的机器，那么，上述的积将被四舍五入到 1.590.这种舍入误差是算法误差，由现代计算中不可避免的大量计算所造成.

支撑理论

虽说我们数值分析的着眼点将面向应用，但我们会自然而然地涉及支撑理论(supporting

theory). 这种理论用于发现算法及建立算法的有效性. 通常指导我们的这个理论具有本质的趣味; 它是有魅力的数学. 于是, 我们就有了两个最好的领域. 但切勿忘记, 我们的兴趣更多的应是实用性的而不是纯美学的.

例 1.5 计算三角函数、指数函数以及其他非初等函数的值, 显然要依赖支撑理论. 对于小的 x, 为了获得它的余弦值, 经典的级数

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots$$

仍是一个好选择. 当 $x = 0.5$, 它变成

$$\cos 0.5 = 1 - 0.125 + 0.0026041 - 0.0000217 + \cdots = 0.877582.$$

这是足够精确的. 在此情况下, 更深层的支撑理论可确定误差界: 该理论指出, 对于像这样的级数, 误差不大于第一个略去的项(见题 1.9). 这里, 第一个略去的项是 $\frac{x^8}{8!}$, 对于 $x = 0.5$, 它恰好达到小于 0.0000001.

数的表示法

由于我们的最终目标是数值的, 故关于数的表示法, 仅用一两句话将是不适当的, 由于我们最熟悉十进制的数, 故数值输入通常将使用这种形式. 然而, 几乎众所周知, 通常计算机发现二进制表示法更方便. 它的 0 和 1 对应于电路的关与开或电压的高与低状态. 对于正整数, 二进制的形式是

$$d_n 2^n + d_{n-1} 2^{n-1} + \cdots + d_1 2^1 + d_0 2^0,$$

而对小于 1 的正数, 它是

$$d_{-1} 2^{-1} + d_{-2} 2^{-2} + d_{-3} 2^{-3} + \cdots,$$

其中, 所有的二进制数字 d_i 取 0 或 1. 这种表示是惟一的.

浮点表示法(floating-point representation)格外方便. 在这种形式中, 数字用三个部分来描述: 一是符号, 一是尾数(mantissa), 一是阶(exponent)(自身也带有一个符号). 作为最初的例子, 回到十进制, 数 0.1492 能表示为

$$+ 0.1492\ 10^0$$

它的符号是 +, 尾数是 0.1492, 而阶是 0. 在其他的可能性中, 以 $+ 1.492\ 10^{-1}$ 代替它也是可以的, 但一般的习惯要求第一位(非零)数字恰好出现在小数点后, 然后, 由阶来处理数量级. 这种表示法被称为规格化(normalized). 于是, 1492 将被表示为 $0.1492\ 10^4$.

例 1.6 将十进制的 13.75 转换为二进制的浮点形式.

有更正式的转换方法. 但即使没有它们, 由于小数点的左边是 8 + 4 + 1, 而右边是 $\frac{1}{2} + \frac{1}{4}$, 易见 13.75 等价的二进制数是 1101.11. 现将它写成

$$+ 0.110111(+ 100),$$

其中, 圆括号中的 100 起着阶 4 的作用, 最终转换成

$$01101110100.$$

倘若懂得某种约定, 对于电的效果来说, 这数中仅有 0 和 1 是令人感兴趣的. 首位 0 被解释为正号(1 将意味着负号). 然后, 6 个二进制数字或比特(bit)作为尾数, 并假定在该尾数前有一个二进制的小数点. 紧接着的 0 是另一个正号, 它是阶的符号, 然后, 这个阶结束该表示法. 最终的形式看上去很不像 13.75, 但它能被理解. 实际中, 尾数与阶将包含更多的位数, 并且符号与阶的形式也将产生变化, 而浮点表示法是现代计算中的一个基本工具.

向量与矩阵的范数

一个向量的 Euclid 长度, 对于分量为 v_i 的向量 V 来说, 即

$$(v_1^2 + v_2^2 + \cdots + v_n^2)^{1/2},$$

它也被称为 V 的一个范数(norm),以符号 $\| V \|$ 记之.这个范数 的 3 个基本性质是

1. $\| V \| \geqslant 0$, $\| V \| = 0$ 当且仅当 $V = 0$;
2. $\| cV \| = |c| \| V \|$,对任意常数 c;
3. $\| V + W \| \leqslant \| V \| + \| W \|$.

最后一个式子即通常所说的**三角不等式**.

若干其他的实函数也具有这些性质,并且也被称为范数.令人特别感兴趣的是 L_p 范数

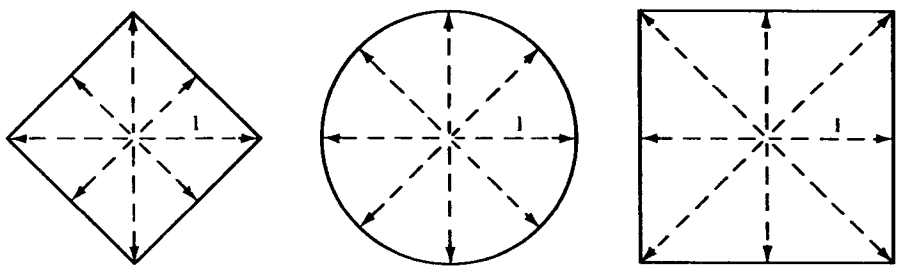
$$\| V \|_p = \left(\sum_{i=1}^{n} | v_i |^p \right)^{\frac{1}{p}}, \quad p \geqslant 1.$$

当 $p = 1$ 时,它是 L_1 范数,是各分量的绝对值之和.当 $p = 2$ 时,它就是熟知的向量长度或 Euclid范数.当 p 趋向于无穷大时,取出占优势的 v_i,于是我们得到最大范数(maximum norm)

$$\| V \|_\infty = \max_i | v_i |.$$

在不止一个场合,尤其是在研究算法的误差特性时,我们将能找到这些范数的用途.

例 1.7　利用 L_1 范数,向量$(1,0)$,$\left(\frac{1}{2}, \frac{1}{2}\right)$,$(0,1)$中的每一个均有范数 1.这种单位向量的平面图给在图 1.1(a)中.原点是它们的起点,而它们的终点构成一个正方形.图 1.1(b)表示更为常见的 Euclid 范数下的单位向量.利用 L_∞范数,向量$(1,0)$,$(1,1)$,$(0,1)$中的每一个均有范数 1,它们的平面图如图 1.1(c)所示,它们的终点也构成一个正方形.



(a)　(b)　(c)

图 1.1

转到矩阵,我们定义

$$\| A \| = \max \| AV \|,$$

这个最大值是取遍所有的单位向量 V 而得到的.这里,"单位"的含义取决于所用的向量范数的类型.这种矩阵范数有平行于上文对向量列出的那些性质:

1. $\| A \| \geqslant 0$, $\| A \| = 0$ 当且仅当 $A = 0$;
2. $\| cA \| = |c| \| A \|$,对于任意常数 c;
3. $\| A + B \| \leqslant \| A \| + \| B \|$.

此外,对于矩阵 A、B 与向量 V,还有如下有用的性质:

4. $\| AV \| \leqslant \| A \| \| V \|$;
5. $\| AB \| \leqslant \| A \| \| B \|$.

L_1 范数和 L_∞范数具有容易计算的优点,前者是具有最大绝对值的一列之和,

$$\| A \|_1 = \max_j \sum_{i=1}^{n} | a_{ij} |,$$

而后者是具有最大绝对值的一行之和,

$$\| A \|_\infty = \max_i \sum_{j=1}^{n} | a_{ij} |.$$

它们的许多特性将在题解中被证明.

例 1.8 找出矩阵

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

的 L_1, L_2 与 L_∞ 范数.

能立即找出最大的列和及行和,于是我们从

$$L_1 = L_\infty = 2$$

迅速着手.不幸的是,没有相应的支撑理论对 L_2 提供帮助以及无困难地使这一外表十分简单的矩阵获得 L_2 范数的值.根据定义,A 的 L_2 范数是向量

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x + y \\ x \end{pmatrix}$$

的最大 L_2 范数,其中 $x^2 + y^2 = 1$,即(x, y)在图 1.1(b)的单位圆周上.这个范数的平方是

$$(x + y)^2 + x^2 = 1 + 2xy + x^2 = 1 + 2x\sqrt{1 - x^2} + x^2.$$

根据基本的微积分知识能取到它的最大值.在这里,由于对(x, y)与$(-x, -y)$,范数取值相同,故不妨设 y 是正的.最后得到:当 $x^2 = \frac{1}{2} + \frac{\sqrt{5}}{10}$时,有最大值

$$\| A \|_2^2 = \frac{3 + \sqrt{5}}{2}.$$

题　解

1.1 计算多项式

$$p(x) = 2x^3 - 3x^2 + 5x - 4$$

在自变量 $x = 3$ 时的值.

解 ☞ 依照自然过程,我们求出 $x^2 = 9, x^3 = 27$,然后将它们合在一起,

$$p(3) = 54 - 27 + 15 - 4 = 38,$$

共计执行了 5 次乘法,1 次加法,2 次减法.

现将该多项式重新整理为

$$p(x) = [(2x - 3)x + 5]x - 4,$$

并且再试算一次.从 $x = 3$ 开始,我们相继得到 6, 3, 9, 14, 42 与 38,这回只用了 3 次而不是 5 次乘法.减少量虽不惹人注意,但它却具有启发意义.对于一个一般的 n 次多项式,第一个算法需要 $2n - 1$ 次乘法,而第二个算法仅需要 n 次算法.在一个较大的运算中,包含着许多多项式的求值运算,减少时间与算法(舍入)误差会意义重大.

1.2 定义一个近似值的误差.

解 ☞ 传统的定义是

$$\text{真值} = \text{近似值} + \text{误差}.$$

因此,例如,

$$\sqrt{2} = 1.414214 + \text{误差},$$
$$\pi = 3.1415926536 + \text{误差}.$$

1.3 相对误差是什么?

解 ☞ 相对误差(relative error)是相对于真值所度量的误差:

$$\text{相对误差} = \frac{\text{误差}}{\text{真值}}.$$

通常的情况下，真值是未知的或者是不适用的，稍加放宽，用近似值代替它并将该结果仍称为相对误差. 于是，对于$\sqrt{2}$来说，熟知的近似值 1.414 有大约为

$$\frac{0.0002}{1.414} = 0.00014$$

的相对误差，而更粗糙的近似值 1.41 有一个接近 0.003 的相对误差.

1.4 设数 $x_1, x_2, \cdots, x_n$ 分别是 $X_1, X_2, \cdots, X_n$ 的近似值，而各自最大的可能误差是 E，证明 x_i 之和的最大的可能误差是 nE.

证 ☞ 因为

$$x_i - E \leqslant X_i \leqslant x_i + E,$$

由加法得到

$$\sum x_i - nE \leqslant \sum X_i \leqslant \sum x_i + nE,$$

因此

$$-nE \leqslant \sum X_i - \sum x_i \leqslant nE.$$

这就是所要证的.

1.5 计算$\sqrt{1} + \sqrt{2} + \cdots + \sqrt{100}$的和，其中，所有的平方根计算到小数点后两位. 按照上题，最大的可能误差是多少？

解 ☞ 由适当选择的很少几行程序，或更老式地，求助于表格，能得到这个问题中的各个根，然后求和，其结果是 671.38. 由于每个根有一个最大的可能误差 $E = 0.005$，所以，和的最大的可能误差 $nE = 100(0.005) = 0.5$. 这意味着所得的和也许连一位准确的小数也没有.

1.6 一个计算结果的概率误差(probable error)的意思是什么？

解 ☞ 这是一种使实际误差超过估计值有$\frac{1}{2}$概率的误差估计. 换言之，实际误差可能大于或小于估计. 由于它取决于误差的分布，所以不是一个容易研究的对象，而通常它被$\sqrt{n}E$ 粗略代替. 其中，E 是最大的可能误差.

1.7 题 1.5 的结果中实际误差是多少？将它与最大误差及概率误差相比较，结果会怎样？

解 ☞ 当平方根求到有 5 位小数时，新的计算方法以和 671.46288 为结果. 此时，最大误差是 100(0.000005)，即 0.0005. 从而我们有准确到 3 位小数的和 671.463. 于是，题 1.5 的结果中实际误差大约是 0.08，与最大(的可能)误差 0.5 及概率误差 0.05 相比，我们的估计一个太悲观而另一个又有些乐观.

1.8 1000 个而不仅仅是 100 个平方根相加，若想得到 3 位小数精确度，那么，参与计算的各个平方根应精确到何种程度？

解 ☞ 为了确保精度，最好是假设最坏的结果即最大的可能误差能达到. 题 1.4 中的公式 nE 变成 $1000E$，这指出了在这种长度的一个求和中，可能会失去 3 位十进制小数. 由于希望在输出中有 3 位准确小数. 所以在输入中具有 6 位准确小数也许是明智的. 其要点是，在一个长长的计算中，存在着非常小的误差汇聚成大值的机会.

1.9 计算级数

$$1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \cdots,$$

准确到 3 位数字.

解 ☞ 这个级数描述了一个常用的分析定理：因为它的项在符号上交替出现正、负，并且不断变小，其部分和来回穿越其极限即该级数的值，这意味着在任一点处的误差将小于第一个舍去的项. 为了获得指定的精度，我们因此需要$\frac{1}{n} \leqslant 0.0005$ 或 $n \geqslant 2000$. 必须将这 2000 项相加. 对 8 位十进制小数运算，2000 个舍入误差可能会积累到 $nE = 2000(0.000000005) = 0.00001$. 这看上去是微不足道的，从而我们允许计算进行下去，将结果舍到 3 位，有 0.693.

注意，在这个问题中没有输入误差，只有算法误差. 首先，我们仅用一个部分和代替这个级数，然

后，在试图求这个和的值时，我们制造了大量的舍入误差. 前者被称为截断误差(truncation error)，并且，在这个问题中，它看上去是两个误差来源中较大的一个. 概言之，

$$\begin{aligned}\text{实际误差} &= \text{截断误差} + \text{舍入误差}\\ &= 0.0005 + 0.00001.\end{aligned}$$

实际上，这个级数的值是 2 的自然对数，而取 3 位小数，它就是我们的 0.693.

1.10 证明：若级数

$$a_1 - a_2 + a_3 - a_4 + \cdots$$

是收敛的，而且，所有的 a_i 均为正数，则

$$\frac{1}{2}a_1 + \frac{1}{2}(a_1 - a_2) - \frac{1}{2}(a_2 - a_3) + \frac{1}{2}(a_3 - a_4) + \cdots$$

也是收敛的，并且表示相同的数.

证 ☞ 用 A_n 和 B_n 表示上述两个级数的 n 项部分和. 易见，$A_n - B_n = \pm\frac{1}{2}a_n$. 由于前一个级数是收敛的，$\lim a_n = 0$，从而得到结果.

1.11 将前题中的定理用于求题 1.9 中的级数值，仍准确到 3 位小数.

解 ☞ 稍用一点代数知识就可得出 $B_1 = \frac{1}{2}$，且对于 $n > 1$，

$$B_n = \frac{1}{2} + \sum_{k=2}^{n}(-1)^k \frac{1}{2k(k-1)}.$$

这仍是一个具有单调下降项的交错级数，从而题 1.9 中的定理仍有效. 为了 3 位数字精确，我们需要

$$\frac{1}{2n(n+1)} \leqslant 0.0005,$$

或 $n \geqslant 32$. 这远远少于题 1.9 中所需的项数，因而在一个 8 位十进制的机器中，舍入误差简直就不成为一个问题了. 新算法较之另一个快多了，并且，它用较少的工作量就获得了相同的 0.693.

1.12 给出足够准确的数 0.1492 和 0.1498，即，其误差不大于第 5 位小数的 5 个单位. 根据所考虑的商 $\frac{1}{0.1492 - 0.1498}$ 来阐明相对误差的形成.

解 ☞ 对于这些给出的数来说，相对误差约为 $\frac{5}{15000}$，接近 0.03 个百分点. 对于它们的和与差而言，产生第 4 位小数中一个单位的最大误差是可能的. 在和的情况下，我们仍导出一个大约为 0.03 个百分点的相对误差，但对于差 0.0006 而言，我们得出一个达到 $\frac{1}{6}$ 的误差，这是 17 个百分点. 回到所求的商，它也许恰好达到最糟的情况. 如所给的，取最接近的整数，算出的商将是 1667. 可设想它本该是 $\frac{1}{0.14985 - 0.14915}$ 但却被取代了，而这使我们得到 1429. 另一个极端是 $\frac{1}{0.14975 - 0.14925} = 2000$. 这个非常简单的例子清楚地说明，在某些持续计算的内部过程中，一个大的相对误差完全可能引出大的绝对误差.

1.13 数值问题的条件指的是什么？

解 ☞ 如果输入信息中的小变化只引起输出的小变化，则称该问题是良态的(well-conditioned)；反之，它是病态的(ill-conditioned). 例如，系统

$$\begin{aligned}x + y &= 1,\\ 1.1x + y &= 2\end{aligned}$$

呈现出一个明显的难点：它表示几乎平行的两直线的交叉，而解为 $x = 10, y = -9$.

今将 1.1 改为 1.05 然后再求解. 此时 $x = 20$ 而 $y = -19$. 一个系数上的 5 个百分点的改变导致了解的 100 个百分点的改变.

1.14 什么是**稳定的**算法？

解 ☞ 在持续的计算中，将可能产生许多舍入误差. 这些舍入误差中的每一个都扮演着剩下的计算中输入误差的角色，并且都对最终的输出结果产生影响. 若所有这些误差对算法的累积影响是有限的，算法能获得有用的结果，则称之为**稳定算法**(stable algorithms). 不幸的是，有这种时候：误差的累积是破坏性的，而解被误差淹没. 不言而喻，这种算法被称为是**不稳定的**(unstable).

1.15　解释浮点十进制数 $0.1066 * 10^4$.

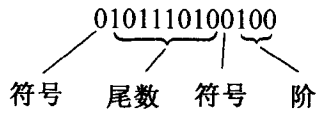
解　☞ 显然,阶 4 使小数点右移 4 位,成为 1066.类似,$0.1066 * 10^{-2}$是 0.001066.

1.16　解释浮点二进制符号 $+0.10111010 * 2^4$

解　☞ 这里的阶将二进制的小数点向右移动 4 位,使之成为 1011.1010.它等价于十进制数 $11+\frac{5}{8}$或 11.625.类似地,$+0.10111010 * 2^{-1}$是 0.01011101,显然,它是原先所给定数的 1/32 倍.

1.17　解释浮点二进制记号 0101110100100.除了它们的符号,尾数使用 8 位而阶使用 3 位.

解　☞ 第一和第十位上的零表示正号,



二进制的小数点取在尾数的前面,按照这些来理解,我们又一次得到了 $+0.10111010 * 2^4$.类似地,并以相同的约定,$+0.10111010 * 2^{-1}$变为 0101110101001,最后 4 位数字的含义是阶为 -1.

1.18　利用前题中的约定,将以下浮点数相加:

$$0101101110010$$
$$0100011001100$$

解　☞ 不管怎么样,二进制小数点必须"对齐".对于记号的解释引出如下的和:

$$\begin{array}{r} 10.110111 \\ +\ 0.000010001100 \\ \hline =10.111001001100 \end{array}$$

使用输入时的形式,它就变成

$$0101110010010$$

这里,除了符号,尾数仍取 8 位,阶取 3 位.由于机器的容量,最后 6 位二进制数字被删除,这就产生了舍入误差.

1.19　什么是溢出?

解　☞ 仍利用我们虚构的机器中的约定,能表示的最大数是 0111111110111,这个数中,尾数和阶均是最大的.二进制小数点向右移动 7 位,等价于 1111111.1,这是十进制的数 $127+\frac{1}{2}$或 2^7-2^{-1}.在已知的约定下,任何大于这个数的数都不能被表示,而这就称为一个溢出(overflow).

1.20　什么是下溢?

解　☞ 除了零和负数,在我们虚拟的机器中,形式上能被表示的最小数是 0000000011111.然而,鉴于各种理由,这样做是适当的:要求尾数的第一位数是 1,然后来确定阶.这就是有名的**规格化形式**(normalized form).零必须作为一个例外.如果规格化是必须的,则最小的正数变成 0100000001111.在十进制数中,它是 $2^{-1} * 2^{-7}$或 2^{-8}.小于这个数的任何正数都不能用该机器来表示,而这就称为**下溢**(underflow).表示数字的任何浮点系统都会受到这种限制,并且都将使用溢出和下溢的概念.

1.21　假想有一个甚至更简单的浮点系统,在这个系统中尾数仅有 3 个二进制数字,且阶只是 -1,0 或 1,那么,这些数是如何分布在一条实线上的?

解　☞ 假定规格化了,则这些数除了阶之外有如下形式:$1\times\times$.于是,整个集合包含三个子集,每一个子集包括 4 个数如下:

0.0100	0.0101	0.0110	0.0111	(对于阶为 -1)
0.100	0.101	0.110	0.111	(对于阶为 0)
1.00	1.01	1.10	1.11	(对于阶为 1)

它们被标在图 1.2 中

注意:较小的数较稠密的存贮.从前一组到后一组,其间隔从$\frac{1}{16}$增加到$\frac{1}{4}$.显然,这是由于我们仅

有 3 位有效数字(第一位固定为 1),它随着阶的增加提供逐级放大的量的缘故.例如,1.005 在这里是不能表示的,它需要 4 位有效数字.而在这部分范围内,集合没有那么稠密.现实中的浮点系统处在更复杂的情况下,但具有相同的特征,并且,有效数字的思想与相对误差是有关的.

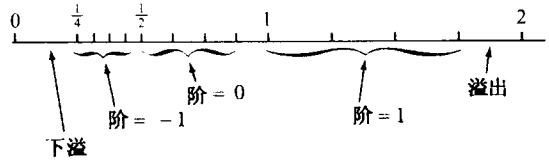


图 1.2

1.22 假定数 x 由浮点二进制符号表示,四舍五入为 n 位尾数,又假定它是规格化了的.那么,由舍入引起的绝对误差(absolute error)和相对误差的界是什么?

解 ☞ 舍入引起的误差至多为二进制数第 $n+1$ 位的一个单位,或第 n 位的半个单位.于是

$$\text{绝对误差} \leqslant 2^{-n-1}.$$

而对于相对误差,我们必须考虑其真值 x.规格化意味着其尾数不小于 $\frac{1}{2}$,这导致如下的界:

$$|\text{相对误差}| \leqslant \frac{2^{-n-1}}{2^{-1}} = 2^{-n}.$$

设 $\mathrm{fl}(x)$ 表示 x 的浮点符号,用它来改写是有益的.于是,

$$\text{相对误差} = \frac{\mathrm{fl}(x) - x}{x} = E$$

或

$$\mathrm{fl}(x) = x(1+E) = x + xE,$$

其中 $|E| \leqslant 2^{-n}$.舍入误差的运算可以看成由一个扰动值(perturbed value) $x + xE$ 去代替 x,而这个扰动相对来说是小的.

1.23 找出由两个浮点数相加造成的相对误差界.

解 ☞ 设这两个数是 $x = m_1 * 2^e$, $y = m_2 * 2^f$,而 y 较小.则 m_2 必须向右移动 $e-f$ 位(对齐二进制的小数点),然后尾数相加,将结果规格化并加以舍入.这样存在两种可能性:或者对二进制小数点左边发生溢出(这里所说的溢出不是题 1.19 意义上的溢出),或者不发生.第一种可能性由

$$1 \leqslant |m_1 + m_2 * 2^{f-e}| < 2$$

来描述,而第二种可能性由

$$\frac{1}{2} \leqslant |m_1 + m_2 * 2^{f-e}| < 1$$

来描述.若确实发生这种溢出,将要求向右移一位,然后我们有

$$\mathrm{fl}(x+y) = [(m_1 + m_2 * 2^{f-e}) * 2^{-1} + \varepsilon] * 2^{e+1},$$

其中,ε 是舍入误差.这可以改写为

$$\begin{aligned}\mathrm{fl}(x+y) &= (x+y)\left(1 + \frac{2\varepsilon}{m_1 + m_2 * 2^{f-e}}\right)\\ &= (x+y)(1+E),\end{aligned}$$

而 $|E| \leqslant 2\varepsilon \leqslant 2^{-n}$.

若不发生这种溢出,则

$$\begin{aligned}\mathrm{fl}(x+y) &= [(m_1 + m_2 * 2^{f-e}) + \varepsilon] * 2^e\\ &= (x+y)\left(1 + \frac{\varepsilon}{m_1 + m_2 * 2^{f-e}}\right)\\ &= (x+y)(1+E),\end{aligned}$$

其中 E 的界同前.

浮点数减法的相应结果,将在题 1.45 中给出.

1.24 找出两个浮点数相乘的相对误差界.

解 ☞ 再一次设这两个数为 $x = m_1 * 2^e$ 和 $y = m_2 * 2^f$，则 $xy = m_1 m_2 * 2^{e+f}$. 因为已规格化，故 $\frac{1}{4} \leqslant |m_1 m_2| < 1$. 这意味着规格化使乘积将至多左移一位. 因此，舍入后的积将或者是 $m_1 + m_2 + \varepsilon$，或者是 $2m_1 m_2 + \varepsilon$，而 $|\varepsilon| \leqslant 2^{-n-1}$. 这能概述如下：

$$\mathrm{fl}(xy) = \begin{cases} (m_1 m_2 + \varepsilon) * 2^{e+f}, & \text{若 } |m_1 m_2| \geqslant \frac{1}{2}, \\ (2m_1 m_2 + \varepsilon) * 2^{e+f-1}, & \text{若 } \frac{1}{2} > |m_1 m_2| \geqslant \frac{1}{4} \end{cases}$$

$$= m_1 m_2 * 2^{e+f} \begin{cases} 1 + \frac{\varepsilon}{m_1 m_2}, & \text{若 } |m_1 m_2| \geqslant \frac{1}{2}, \\ 1 + \frac{\varepsilon}{2m_1 m_2}, & \text{若 } \frac{1}{2} > |m_1 m_2| \geqslant \frac{1}{4} \end{cases}$$

$$= xy(1 + E),$$

而 $|E| \leqslant 2|\varepsilon| \leqslant 2^{-n}$.

题 1.46 中，对于除法运算概述了一个类似结果. 这意味着在所有的 4 种算术运算中，利用浮点数，引来的相对误差不超过尾数的最末位有效数字上的 1.

1.25 估计利用浮点运算计算和

$$x_1 + x_2 + \cdots x_k$$

所产生的误差.

解 ☞ 我们考虑部分和 s_i. 设 $s_1 = x_1$，则

$$s_2 = \mathrm{fl}(s_1 + x_2) = (s_1 + x_2)(1 + E_1).$$

正如题 1.23 中所证明的，E_1 的界为 2^{-n}. 改写

$$s_2 = x_1(1 + E_1) + x_2(1 + E_1),$$

继续下去

$$s_3 = \mathrm{fl}(s_2 + x_3) = (s_2 + x_3)(1 + E_2)$$
$$= x_1(1 + E_1)(1 + E_2) + x_2(1 + E_1)(1 + E_2) + x_3(1 + E_2),$$

而最后有

$$s_k = \mathrm{fl}(s_{k-1} + x_k) = (s_{k-1} + x_k)(1 + E_{k-1})$$
$$= x_1(1 + c_1) + x_2(1 + c_2) + \cdots + x_k(1 + c_k),$$

其中，对于 $i = 2, \cdots, k$，

$$1 + c_i = (1 + E_{i-1})(1 + E_i)\cdots(1 + E_{k-1}),$$

且 $1 + c_1 = 1 + c_2$. 鉴于 E_j 的界相同，对于 $1 + c_i$，现在我们有如下估计：

$$(1 - 2^{-n})^{k-i+1} \leqslant 1 + c_i \leqslant (1 + 2^{-n})^{k-i+1},$$

故可概括出

$$\mathrm{fl}\Big(\sum_{j=1}^{k} x_j\Big) = \Big(\sum_{j=1}^{k} x_j\Big)(1 + E),$$

其中，

$$E = \sum_{j=1}^{k} x_j c_i \Big/ \sum_{j=1}^{k} x_j.$$

注意：若真和 $\sum x_j$ 相对于 x_j 是小的，则相对误差 E 可能是大的，这是由减法引起的相消结果，早在题 1.12 中就被注意到了.

1.26 阐述向前误差分析.

解 ☞ 假定 $A(B + C)$ 的值是由近似值 a, b, c 计算出来的，这些近似值误差量是 e_1, e_2, e_3，则真值

$$A(B + C) = (a + e_1)(b + e_2 + c + e_3) = ab + ac + \text{误差},$$

其中，

$$\text{误差} = a(e_2 + e_3) + be_1 + ce_1 + e_1 e_2 + e_1 e_3.$$

假设有统一的误差界 $|e_i| \leqslant e$，且误差之积可以忽略不计，那么，我们得到

$$|\text{误差}| \leqslant (2|a|+|b|+|c|)e.$$

这个典型的过程被称为向前误差分析(forward error analysis),原则上它能运用于任何算法.然而,通常这种分析不是不知所措的就是冗长乏味的.除此之外,所产生的这个界,通常是非常保守的,只适应于想知道最坏可能发生什么情况.在此例中,很少被留意但确实出现的一点是:看上去 a 的值比 b 和 c 的值敏感两倍.

1.27 什么是向后误差分析?

解 ☞ 向后误差分析(backward error analysis)的本质思想是:先接受计算结果,然后去确定能产生它的输入数据的范围.在这里,重要的是不要误解这样做的动机即不存在修改输入数据使之适应计算结果的企图.若完成了向后误差分析而显示出所获结果与输入数据的观测或舍入误差的范围相一致,则结果可以信赖.反之,则在另外的地方存在误差的主要来源,按推测,它存在于算法本身.

1.28 说明在题 1.23 中的误差分析就是向后误差分析.

解 ☞ 1.23 中得到的结果是

$$\mathrm{fl}(x+y) = (x+y)(1+E),$$

$|E| \leqslant 2^{-n}$,其中,n 是二进制尾数的位数.将它改写为

$$\mathrm{fl}(x+y) = x(1+E) + y(1+E).$$

回顾题 1.22,我们发现计算所得的和,即 $\mathrm{fl}(x+y)$,仍是与原始数据 x 和 y 相差不大于一个舍入误差界 E 的两个数的真和,也就是说,输出可由恰当地落在认可的误差限制内的输入数据来解释.

1.29 说明在题 1.24 中所做的分析是向后误差分析.

解 ☞ 我们发现,

$$\mathrm{fl}(xy) = xy(1+E)$$

可看成是 x 乘以 $y(1+E)$的积.这意味着计算出来的 $\mathrm{fl}(xy)$也是与原来的 x, y 的差别不大于舍入误差的两个数的真积(true product),这与输入数据在我们认可的误差限制内是一致的.

1.30 在题 1.25 中,向后误差分析说明了什么?

解 ☞ 首先,方程

$$\mathrm{fl}\Big(\sum_{j=1}^{k} x_j\Big) = x_1(1+c_1) + x_2(1+c_2) + \cdots + x_k(1+c_k)$$

表示 k 个数 x_1 到 x_k 的浮点运算之和也是 k 个数的真和,这 k 个数与原来的 k 个 x_j 差在相对误差 c_j 上.不幸的是,题 1.25 中得到的估计也表明这些误差可能远远大于单一的舍入误差.

1.31 取 L_2 范数,由先证 Cauchy-Schwarz 不等式

$$\Big(\sum a_i b_i\Big)^2 \leqslant \Big(\sum a_i^2\Big)\Big(\sum b_i^2\Big),$$

来证明向量长度的三角性质.

解 ☞ 一种有趣的证明从注意到 $\sum (a_i - b_i x)^2$ 非负时开始,于是二次方程

$$\Big(\sum b_i^2\Big)x^2 - 2\Big(\sum a_i b_i\Big)x + \sum a_i^2 = 0$$

不能有不同的实根,这要求

$$4\Big(\sum a_i b_i\Big)^2 - 4\sum a_i^2 \sum b_i^2 \leqslant 0,$$

约去 4,我们便得到 Cauchy-Schwarz 不等式.

现在,仅用一点点代数知识,就能立即得到三角不等式,写成分量形式,有

$$[(v_1+w_1)^2 + \cdots + (v_n+w_n)^2]^{\frac{1}{2}} \leqslant (v_1^2 + \cdots + v_n^2)^{\frac{1}{2}} + (w_1^2 + \cdots + w_n^2)^{\frac{1}{2}}.$$

对它平方,合并同类项,再平方,利用 Cauchy-Schwarz 不等式就将得到所需的结果(见题 1.50).

1.32 试证,当 p 趋向于无穷,向量的 L_p 范数逼近于 $\max|v_i|$.

证 ☞ 假设 v_m 是绝对值最大的分量,并将和改写为

$$|v_m| \left(1 + \sum_{i \neq m} \left|\frac{v_i}{v_m}\right|^p\right)^{1/p},$$

括号内除了第一项外,所有的项趋于零,于是有所需的结果.

1.33　试证，对于单位向量 V，定义的 $\|A\| = \max \|AV\|$ 能满足引言中所给的性质 1～3.

证　☞ 这些性质很容易从相伴向量范数对应的性质得到. 由于 AV 是向量，$\|AV\| \geqslant 0$，故 $\|A\| \geqslant 0$. 若 $\|A\| = 0$，而 A 哪怕仅有一个不为零的元素，则能选择某 V 使得 AV 的一个分量是正的，这与 $\|AV\| = 0$ 矛盾. 这就证出了性质 1.

其次，我们有

$$\|cA\| = \max \|cAV\| = \max |c| \cdot \|AV\| = |c| \cdot \|A\|.$$

这证明了性质 2. 性质 3 可类似地处理.

1.34　单位矩阵(identity matrix)的 L_1, L_2 和 L_∞ 范数分别是什么?

解　☞ 它们均为 1. 因为 V 是单位向量，我们有

$$\|I\| = \max \|IV\| = \max \|V\| = 1.$$

1.35　矩阵 $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ 的 L_1, L_2 和 L_∞ 范数分别是什么?

解　☞ 我们有

$$AV = \begin{bmatrix} v_1 + v_2 \\ v_1 + v_2 \end{bmatrix}.$$

为了简单起见，设 v_1, v_2 非负. 由于 V 是 L_1 范数中的单位向量，因而对于 L_1 范数，我们相加有 $\|AV\| = 2(v_1 + v_2) = 2$，于是 $\|A\|_1 = 2$. 对于 L_2 范数，我们需对两分量平方然后相加，得到 $2(v_1^2 + 2v_1v_2 + v_2^2)$. 在这种范数中，$v_1^2 + v_2^2 = 1$，于是我们对 v_1v_2 取最大值，由基本的微积分运算可得 $v_1 = v_2 = \frac{1}{\sqrt{2}}$，立得 $\|A\|_2 = 2$. 最后，因为使用 L_∞ 范数我们要找的是最大分量，所以 $\|AV\|_\infty = v_1 + v_2$. 由于使用这种范数，$v_1, v_2$ 均不超过 1，故这里最大值仍是 2. 利用下面的题或它的相伴题的结果，L_1 和 L_∞ 范数能很快被解出.

1.36　试证

$$\|A\|_\infty = \max_i \sum_{j=1}^{n} |a_{ij}|.$$

证　☞ 选择一个所有分量为 1 且符号与 a_{ij} 相匹配的向量 V，使得 $\sum |a_{ij}|$ 最大. 于是，$\sum a_{ij}v_j$ 是 AV 的一个等于该最大值的元素，并且，它显然不可能被超过. 由于 V 的范数是 1，故 A 的范数仍取此值. L_1 范数的类似结果留作题 1.52.

1.37　证明 $\|AV\| \leqslant \|A\| \cdot \|V\|$.

证　☞ 根据 $\|A\|$ 的定义，对于单位向量 U，我们有

$$\|AU\| = \max_U \|AU\| = \|A\|,$$

选取 $U = V/\|V\|$ 并应用性质 2，

$$\left\| A\left(\frac{V}{\|V\|}\right) \right\| \leqslant \|A\|, \quad \|AV\| \leqslant \|A\| \, \|V\|.$$

1.38　证明 $\|AB\| \leqslant \|A\| \cdot \|B\|$.

证　☞ 我们重复利用题 1.37 的结果，

$$\|AB\| = \max \|ABU\| \leqslant \max \|A\| \cdot \|BU\| \leqslant \max \|A\| \cdot \|B\| \cdot \|U\|$$
$$= \|A\| \cdot \|B\|.$$

补　充　题

1.39　利用支撑理论

$$\frac{1}{1-x} = 1 + x + x^2 + \cdots$$

计算 $\frac{1}{0.982}$，其中 $x = 0.018$.

1.40 当误差不超过 0.005 时，数字准确到 2 位小数，下面的平方根取自一个表.

n	11	12	13	14	15	16	17	18	19	20
$\sqrt{n}$四舍五入到 3 位小数	3.317	3.464	3.606	3.742	3.873	4.000	4.123	4.243	4.359	4.472
$\sqrt{n}$四舍五入到 2 位小数	3.32	3.46								
舍入误差的近似值	-0.003	-0.004								

试将每个数舍入到两位小数并注出舍入误差的数量大小. 这些舍入误差与误差最大值 0.005 相比情况如何？理论上这 10 个数总舍入误差在 10(-0.005)到 10(0.005)之间，而实际上是多少？它与“概率误差”$\sqrt{10}(0.005)$相比，情况如何？

1.41 假设 N 个数均准确到给定的位数，求其和. 用概率误差公式来估计，当 N 大约多大时，计算出来的和的最后一位数字将可能无意义？当 N 大约多大时，和的最后两位数可能无意义？

1.42 序列 $J_0, J_1, J_2, \cdots$ 由

$$J_{n+1} = 2nJ_n - J_{n-1}$$

定义，$J_0 = 0.765198$，$J_1 = 0.440051$ 均准确到 6 位小数. 试计算 $J_2, \cdots, J_7$ 且将它们与下表中的准确值进行比较(这些准确值是由另一完全不同的方法得到的. 对误差的解释见下题)

n	2	3	4	5	6	7
精确的 J_n	0.114903	0.019563	0.002477	0.000250	0.000021	0.000002

1.43 试证：对于上题的序列，精确地有

$$J_7 = 36767J_1 - 21144J_0.$$

由给定的 J_0 和 J_1 值来计算，将得到同样不准确的值. 大系数乘给定的 J_0 和 J_1 值中的舍入误差，则合并后的结果含有一个大误差.

1.44 数 J_8 直到 6 位都将为零，按题 1.42 中的公式，实际得出什么？

1.45 试证浮点数减法所产生的误差以 2^{-n} 为界. 如在题 1.23 中那样，设 $x = m_1 * 2^e$，$y = m_2 * m^f$，则 $x - y = (m_1 - m_2 * 2^{f-e})2^e$，除非它为零，否则就有

$$2^{-n} \leqslant | m_1 - m_2 * 2^{f-e} | < 2.$$

对这个新的尾数进行规格化处理，也许小数点需要左移 $n-1$ 位，而实际的数 s 由

$$2^{-s-1} \leqslant | m_1 - m_2 * 2^{-ef} | < 2^{-s}$$

所决定. 今试证

$$\mathrm{fl}(x - y) = [(m_1 - m_2 * 2^{f-e}) * 2^s + \varepsilon] * 2^{e-s},$$

而最终

$$\mathrm{fl}(x - y) = (x - y)(1 + E),$$

其中 $|E| \leqslant 2^{-n}$.

1.46 试证浮点数除法所产生的误差以 2^{-n} 为界. 按照题 1.24 中的约定，让分子尾数的一半除以分母的尾数(这是为了避免商大于 1)而阶相减，它给出了

$$\frac{x}{y} = \left(\frac{m_1}{2m_2}\right) * 2^{e-f+1}$$

其中 $\frac{1}{4} \leqslant | m_1/2m_2 | < 1$. 现仿效对乘法运算所作的分析步骤，再一次证明相对误差 E 如所述，是有界的.

1.47 分析内积计算

$$s_k = \mathrm{fl}(x_1y_1 + x_2y_2 + \cdots + x_ky_k).$$

它酷似题 1.25. 设

$$t_i = \mathrm{fl}(x_iy_i), \quad i = 1, \cdots, k,$$

接着令

$$s_1 = t_1 \quad s_i = \mathrm{fl}(s_{i-1} + t_i), \quad i = 1, \cdots, k$$

它造出了所求的内积 s_k. 今求出类似于那些在前面的题中得到的关系式和估计.

1.48　使用题 1.17 的约定，解释浮点符号 0100110011010(这是仅以 8 位尾数对0.1492最可能的接近).

1.49　仿效题 1.21，想象一个浮点系统，在该系统中规格化的尾数有 4 位，而阶为 $-1,0,1$. 证明这些数形成了三组，各有 8 个数；对应于它们的阶，一组落在 1/4 到 1/2 区间中，另一组在 1/2 到 1 区间中，而第三组在 1 与 2 之间. 哪个正数会造成溢出？哪个下溢？

1.50　完成在题 1.31 中开始的证明.

1.51　通过证明两个矩阵和的范数不超过它们范数的和来完成题 1.33.

1.52　通过对单位向量的适当选择(一个分量为 1，其余的为零)，证明矩阵 A 的 L_1 范数可以从绝对值元素的最大列和算得，并与题 1.36 中相关的证明相比较.

1.53　证明对矩阵 $A=\begin{bmatrix} a & b \\ b & a \end{bmatrix}$，$L_1, L_2$ 及 L_3 范数均相等.

1.54　证明对矩阵 $A=\begin{bmatrix} a & b \\ b & -a \end{bmatrix}$，其 L_2 范数为 $(a^2+b^2)^{1/2}$.

1.55　证明对矩阵 $A=\begin{bmatrix} a & a \\ a & b \end{bmatrix}$，可以得到一个使 $\|AV\|_2$ 为极大的向量 V，其形式为 $(\cos t, \sin t)^T$. 其中，在 $b^2=a^2$ 的情况下，$\cos 2t=0$，而在另外的情况下 $\tan 2t=2a/(a-b)$.

1.56　下面的信息已经被建议作为本行星生活着智慧生命的信号传播到外层空间. 这里的想法是，无论在什么地方的任何形式的智慧生命都会理解这信息的智慧内涵，并由此推知这里存在着我们所拥有的智慧. 该信息

11.00100100001111101110

的意义是什么？

1.57　若以 x, y 为分量的向量 $\boldsymbol{V}$ 表示平面上的一个点 (x, y)，则对应于取 L_2 范数的单位向量的点形成古典的单位圆. 如图 1.1 所示，对 L_1 及 L_∞ 范数，该“圆”取作正方形. 在一个有正方形街区的城市中，对出租车行进来说，哪一种是合适的范数(从一个交叉点出发，在给定距离中，找出所有的交叉点). 在一个棋盘上，为什么对国王的行进而言，合适的范数是 L_∞ 范数？

第二章　配置多项式

多项式逼近

多项式逼近是数值分析中最古老的思想之一，而且是迄今仍最受重用的方法之一.对于一个函数 $y(x)$，用一个多项式 $p(x)$代替它有着众多的理由，其中最重要的也许是因为多项式便于计算，它仅涉及到简单的整数幂.它们的导数和积分也不难得到，并且仍然是多项式.多项式较之其他函数易于求根，故而流行用多项式代替其他函数是不难理解的.

逼近准则

差 $y(x)-p(x)$是逼近误差.显然，其核心思想是保持该误差合理地小.由于多项式简单，允许以不同的方法接近这个目标.在这些方法中我们考虑的是

1. 配置(collocation)
2. 密切(osculation)
3. 最小二乘(least squares)
4. 极小-极大(min-max)

配置多项式

配置多项式(collocation polynomial)是这一章及下面少数几章中的研究对象，在某些指定的点上，它与 $y(x)$重合，这种多项式与一般的多项式的许多性质在展开过程中起作用.

1. **存在和惟一性定理**指出，对于自变量 $x_0,\cdots,x_n$，恰好存在一个 n 次配置多项式，即，使得对这些变量，$y(x)=p(x)$.存在性将由实际展示在后继的章节中的多项式证实.惟一性将在本章中被证明.它是多项式某些基本性质的一个结果.
2. **辗转相除法**.任何多项式可以表示为
$$p(x)=(x-r)q(x)+R,$$
其中，r 是任意数，$q(x)$是一个 $n-1$ 次多项式，而 R 是一个常数.它有两个直接的推论.
3. **剩余定理**.(remainder theorem)指出，$p(r)=R$.
4. **因式定理**.(factor-theorem)指出，若 $p(r)=0$，则 $x-r$ 是 $p(x)$的因式.
5. **零点限制**.一个 n 次多项式至多有 n 个零点，这意味着方程 $p(x)=0$ 至多有 n 个根.作为需要证明的惟一性定理，是一个直接的推论.正如将证明的那样.
6. **综合除法(synthetic division)**.对于获取 $q(x)$和 R 的辗转相除法来说，是一个经济的程序(或算法).通常它被用于求 R，由剩余定理，$R=p(r)$.求 $p(r)$的这条路也许比直接计算多项式的值更好.
7. **乘积**.$\pi(x)=(x-x_0)(x-x_1)\cdots(x-x_n)$在配置理论中起着重要作用.注意，在配置自变量 $x_0,x_1,\cdots,x_n$ 处，乘积为零.配置多项式的误差将被证明是
$$y(x)-p(x)=\frac{y^{(n+1)}(\xi)\pi(x)}{(n+1)!},$$
其中，ξ 取决于 x.并且，倘若 x 是配置端点，则 ξ 位于端点间.注意到这个公式在 $x_0,x_1,\cdots,x_n$ 处为零，从而在这些点上，$p(x)$确实与 $y(x)$相配置，而在其他的地方，我们将 $p(x)$看作是对 $y(x)$的逼近.

题　解

2.1 证明：任一多项式 $p(x)$能表示为

$$p(x) = (x - r)q(x) + R,$$

其中，r 是任意数，$q(x)$是一个 $n-1$ 次多项式，而 R 是一个常数.

证 ☞ 这是辗转相除法的一个例子.设 $p(x)$为 n 次多项式，

$$p(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_0,$$

则

$$p(x) - (x - r)a_n x^{n-1} = q_1(x) = b_{n-1}x^{n-1} + \cdots$$

将不大于 $n-1$ 次.类似地有，

$$q_1(x) - (x - r)b_{n-1}x^{n-2} = q_2(x) = c_{n-2}x^{n-2} + \cdots$$

将不大于 $n-2$ 次.依次类推，最终我们可得到一个 0 次多项式 $q_n(x)$，即一个常数，记这个常数为 R，我们有

$$p(x) = (x - r)(a_n x^{n-1} + b_{n-1}x^{n-2} + \cdots) + R = (x - r)q(x) + R.$$

2.2 证明 $p(r) = R$.而这被称为剩余定理.

证 ☞ 设题 2.1 中 $x = r$，立得 $p(r) = 0 \cdot q(r) + R$.

2.3 利用 $r = 2$，$p(x) = x^3 - 3x^2 + 5x + 7$，来阐述“综合除法”，用以完成题 2.1 中所述的除法.

解 ☞ “综合除法”只不过是题 2.1 中相同运算的缩写形式，它仅出现各系数.对于上述的 $p(x)$和 r，开始的格式是

$$\begin{array}{l|llll} r=2 & 1 & -3 & 5 & 7 \leftarrow p(x)\text{的系数} \\ \hline & 1 & & & \end{array}$$

“乘以 r 且相加”三次后，完成了格式.

$$\begin{array}{l|llll} r=2 & 1 & -3 & 5 & 7 \\ & & 2 & -2 & 6 \\ \hline & \underbrace{1 \quad -1 \quad 3}_{q(x)\text{的系数}} & & & 13 \leftarrow \text{此为数 } R \end{array}$$

于是，$q(x) = x^2 - x + 3$，$R = f(2) = 13$.由计算 $(x - r)q(x) + R$ 可验证，这就是 $p(x)$.对于寻找 $q(x)$ 而言，“长除法(long division)”也是有用的.它从常见的格式

$$(x - 2)\sqrt{x^3 - 3x^2 + 5x + 7}$$

开始.将产生此结果的计算与刚才完成的“综合”除法相比较，易见两者是等价的.

2.4 证明：若 $p(r) = 0$，则 $x - r$ 是 $p(x)$的一个因式.这就是**因式定理**，那剩下的因式为 $n-1$ 次.

证 ☞ 若 $p(r) = 0$，则 $0 = 0 \cdot q(x) = R$，从而 $R = 0$.于是，

$$p(x) = (x - r)q(x).$$

2.5 证明 n 次多项式至多有 n 个零点，这意味着 $p(x) = 0$ 至多有 n 个根.

证 ☞ 假设存在 n 个根，记为 $r_1, r_2, \cdots, r_n$，将因式定理应用 n 次，则有

$$p(x) = A(x - r_1)(x - r_2)\cdots(x - r_n)$$

其中，A 有零次幂，是一个常数.这清楚地表明不可能存在其他的根(同时指出 $A = a_n$).

2.6 证明：至多有一个 n 次多项式在给定的自变量 x_k 处能取到指定的值 y_k，其中 $k = 0, 1, \cdots, n$.

证 ☞ 假设存在两个这样的多项式 $p_1(x)$和 $p_2(x)$，那么其差 $p(x) = p_1(x) - p_2(x)$将不大于 n