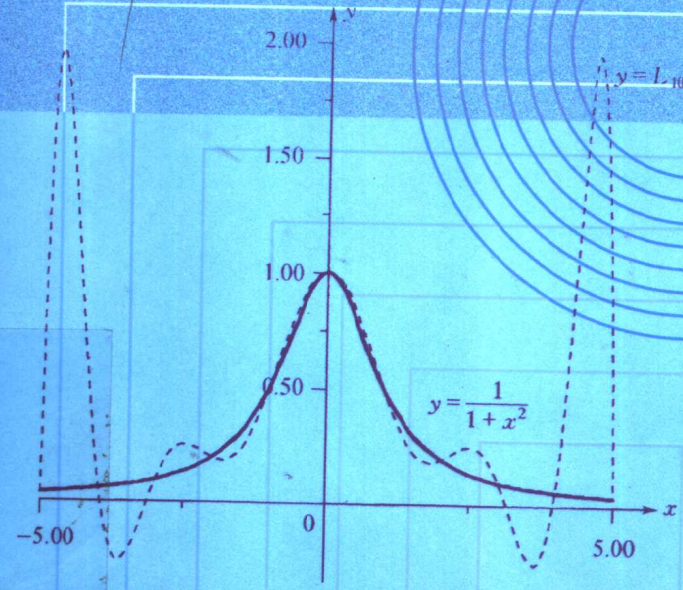
数值分析

翟瑞彩　谢伟松



天津大学出版社

数值分析

翟瑞彩　谢伟松

天津大学出版社

内容提要

本书介绍科学与工程计算中常用的数值计算方法及其有关理论，其中包括线性代数方程组的直接解法与迭代法、矩阵特征值问题的数值解法、插值法与数值逼近、数值积分与数值微分、常微分方程的数值解法、非线性方程(组)的数值解法，并简单介绍了偏微分方程的差分法与有限元方法.各章都有应用例题和一定量的习题.可作为大学本科生及硕士研究生的教科书或教学参考书，也可供科技工作者参考.

图书在版编目（CIP）数据

数值分析/翟瑞彩，谢伟松主编. —天津：天津大学出版社，2001.9 重印

ISBN 7-5618-1366-X

Ⅰ.数… Ⅱ.①翟… ②谢… Ⅲ.计算方法 Ⅳ.0241.4

中国版本图书馆 CIP 数据核字（2000）第 76060 号

出　版　天津大学出版社
出版人　杨风和
地　址　天津市卫津路 92 号天津大学内（邮编：300072）
电　话　发行部：022—27403647 邮购部：022—27402742
印　刷　河北省枣强新华胶印厂
发　行　新华书店天津发行所
开　本　880mm×1230mm　1/32
印　张　11.25
字　数　328 千
版　次　2000 年 11 月第 1 版
印　次　2001 年 9 月第 2 次
印　数　2001－5000
定　价　17.50 元

前　言

随着电子计算机的广泛应用和科学技术的迅速发展，使用计算机进行科学和工程技术领域的科学计算已经成为不可缺少的重要环节.事实上，科学计算已经与理论分析、科学实验成为平行的研究和解决科技问题的科学手段，经常被科技工作者所采用.作为科学计算的核心内容——数值分析(或数值计算方法)，已逐渐成为广大科技工作者必备的基本知识并越来越被人们所重视.因此，目前在高等理工院校，数值分析已经成为普遍开设的基础课.编写本书的目的是介绍科学计算中常用的数值计算方法及其相关理论，旨在使读者了解科学计算的重要性，掌握基本数值计算方法及其理论，懂得如何构造算法、评价算法的优劣，培养读者应用计算机独立从事科学计算的能力.

本书是编者在多年来为天津大学理工科大学生及硕士研究生讲授数值分析课程所用教材的基础上，经重新修改、补充、整理编写而成.书中系统地介绍了数值计算的基本方法及其相关理论，包括解线性方程组的直接法、迭代法，矩阵特征值问题的数值解法，插值与逼近，数值积分与数值微分，常微分方程数值解法，非线性方程(组)的数值解法，并简单介绍了偏微分方程的数值解法——差分法与有限元法.对误差估计、数值算法的收敛性与稳定性等做了适当的分析，各章节都列举了应用例题并配有一定数量的习题.若教学时数较少，则可对上述教学内容做适当的删减.故本书兼顾了多学时与少学时之要求.

本书可作为理工科大学本科高年级学生和硕士研究生数值分析(或计算方法)课程的教科书或教学参考书，也可供从事科学计算的科技工作者参考.

本书第1,3,9章由谢伟松编写，其余各章均由翟瑞彩编写.

承蒙寇述舜教授对全书作了仔细的审阅，提出了不少宝贵的修改意见.在此，谨致以诚挚的谢意。

对本书编写过程中所用参考书和文献资料的作者一并致谢，向天

津大学出版社以及关心、支持并帮助本书出版的同志致谢.

由于编者水平有限,如有不当,恳请读者批评指正.

编　者

2000 年 8 月

目　录

第 1 章　引论 …… (1)
1.1　数值分析的研究对象 …… (1)
1.2　数值计算误差的基本知识 …… (1)
1.3　数值算法的稳定性和收敛性 …… (7)
习题 1 …… (12)
第 2 章　线性方程组的数值解法 …… (13)
2.1　Gauss 消去法 …… (13)
2.2　矩阵的三角分解及其应用 …… (25)
2.3　向量和矩阵的范数 …… (38)
2.4　方程组的性态与误差分析 …… (52)
2.5　解线性方程组的迭代法 …… (59)
2.6　迭代法的收敛性分析 …… (66)
习题 2 …… (74)
第 3 章　矩阵特征值与特征向量的计算 …… (79)
3.1　乘幂法与反幂法 …… (79)
3.2　Jacobi 方法 …… (87)
3.3　*QR* 方法 …… (96)
习题 3 …… (103)
第 4 章　函数的插值 …… (104)
4.1　插值问题的基本概念 …… (104)
4.2　Lagrange 插值公式及其余项 …… (107)
4.3　Newton 插值公式及其余项 …… (114)
4.4　Hermite 插值 …… (126)
4.5　分段插值 …… (132)
4.6　三次样条插值 …… (138)
习题 4 …… (149)
第 5 章　函数的数值逼近 …… (151)

5.1 正交多项式 …… (151)
5.2 最佳平方逼近 …… (161)
5.3 用正交多项式作函数的最佳平方逼近 …… (164)
5.4 曲线拟合的最小二乘法 …… (167)
习题 5 …… (178)
第 6 章 数值积分与数值微分 …… (180)
6.1 数值积分公式及其代数精度 …… (180)
6.2 插值型数值积分公式与 Newton-Cotes 公式 …… (182)
6.3 复化求积法 …… (190)
6.4 变步长的梯形公式与 Romberg 算法 …… (194)
6.5 Gauss 求积公式 …… (201)
6.6 数值微分 …… (214)
习题 6 …… (220)
第 7 章 常微分方程的数值解法 …… (223)
7.1 初值问题计算格式的建立 …… (224)
7.2 Runge-Kutta 方法 …… (230)
7.3 收敛性与稳定性 …… (236)
7.4 线性多步法 …… (242)
7.5 一阶常微分方程组与高阶方程的数值解法 …… (251)
7.6 常微分方程边值问题的差分解法 …… (254)
习题 7 …… (263)
第 8 章 非线性方程与方程组的数值解法 …… (265)
8.1 二分法 …… (265)
8.2 迭代法 …… (268)
8.3 Newton 法 …… (279)
8.4 弦截法 …… (286)
8.5 非线性方程组的解法 …… (288)
习题 8 …… (303)
第 9 章 偏微分方程的数值方法 …… (306)
9.1 椭园型方程的差分方法 …… (306)

9.2 发展型方程的差分方法 …… (317)
9.3 发展型方程差分格式的收敛性和稳定性 …… (329)
9.4 有限元方法简介 …… (340)
习题 9 …… (348)
参考文献 …… (350)

第 1 章　引　　论

1.1　数值分析的研究对象

随着计算机科学的迅速发展，目前几乎在所有的科技及工程领域中，均采用了数值计算作为其研究、设计手段. 数值计算已经在科学研究和工程实际中，得到了越来越广泛的应用. 并且，随着计算机被广泛地应用于大型的科学和工程计算，数值计算方法作为数学科学中的一个重要分支，已迅速地发展起来.

应用数值计算方法解决科学研究和工程实际中的科学计算问题，首先要建立描述具体问题的适当数学模型；其次要选择一定的计算方法并制定相应的计算方案；并编制、设计合理的计算机程序；最后由计算机计算出数值结果. 其中，计算方案的设计和计算方法的选择是上述求解过程中极其重要的一个环节，是程序设计和分析数值计算结果准确性的基础.

本书将介绍科学和工程计算中的一些算法以及有关的数学理论，即针对某些具体的数学模型，给出其具体的数值计算方法，并对这些方法的适用性做出理论分析和论证.

1.2　数值计算误差的基本知识

1.2.1　误差的产生及其基本概念

在本书中，用于科学计算的各种方法多数是通过数值计算来实现的. 注意到计算机中的数一般都表示为 16 位、32 位或 64 位的二进制

数，因此任何一个实数只能表示为有限位数，二者之间存在着一个误差——舍入误差.

另外，对于计算中经常遇到的超越运算和极限运算，其精确解是很难在有限步内得到的.但是，在实际的数值计算过程中，通常采用的方法是利用有限的算术运算，用有限步骤内得到的近似结果来代替其精确解.由于这里截去了若干步之后的计算内容，故所引起的误差称为“截断误差”.

另外，对于大型的科学计算问题，计算过程中所采用的某些已知数据（如初始条件等）往往是通过实验结果或测量结果给出的，与实际数据有着一定的误差，这种误差被称为“观测误差”.

由于“观测误差”是一种由具体的实验或测量手段所引入的误差，不是科学计算过程所能够避免的，因此本书考虑的通常为前两种误差：舍入误差和截断误差.

在分析数值计算结果的误差时，需要有一个比较客观的评价尺度.针对评价方法的不同，在对数值计算结果的精度进行表示时，可分别用绝对误差、相对误差和有效数字三种方式进行描述.下面对以上概念做适当的说明.

定义 1.2.1 设 x^* 为准确值 x 的近似值，则称

$$E(x)=x-x^* \tag{1.2.1}$$

为近似值 x^* 的绝对误差.若存在一个正数 η，使得

$$|x-x^*|\leqslant\eta, \tag{1.2.2}$$

则称 η 为近似值 x^* 的绝对误差限.

由式(1.2.2)知，x 一定落在区间 $[x^*-\eta, x^*+\eta]$ 上.在工程中常用 $x=x^*\pm\eta$ 表示准确值 x，用以标明近似值 x^* 的精确度或准确值 x 所在区间的范围.

但是，绝对误差的大小尚不能完全刻画近似值的准确程度.例如，如果测量 10m 的长度时有 1cm 的绝对误差，测量 1000m 的长度时有 10cm 的绝对误差，那么后者的绝对误差是前者的 10 倍.但是如果考虑被测量的长度本身的数值，则后者每 10m 才有 0.1cm 的绝对误差，绝对误差所占的比例为 1/10 000；而前者这一比例为 1/1 000.因此，后者

的测量应该是较精确的.这就启发我们,在考虑绝对误差的同时,还要考虑近似值本身的大小.为此,我们引入相对误差的概念.

定义 1.2.2 设 x^* 为准确值 x 的近似值,则称

$$E_r(x)=\frac{x-x^*}{x} \tag{1.2.3}$$

为近似值 x^* 的相对误差.在实际应用中,常将

$$E_r^*(x)=\frac{x-x^*}{x^*} \tag{1.2.4}$$

称为近似值 x^* 的相对误差.如果存在一个正数 δ,使得

$$|E_r^*(x)|\leqslant\delta, \tag{1.2.5}$$

则称 δ 为近似值 x^* 的相对误差限.

对于十进制数而言,为了能从近似值本身得到其相对误差的大小,我们将引入有效数字的概念.

注意到,近似数 x^* 可以写成

$$x^*=\pm 10^m\sum_{j=1}^{n}a_j10^{-j},\ m\text{ 为整数},a_1\neq 0. \tag{1.2.6}$$

若 x^* 的最后一位是由 x 经四舍五入得到的,则

$$|x-x^*|\leqslant\frac{1}{2}\times 10^{m-n}. \tag{1.2.7}$$

定义 1.2.3 若 x 的近似值可以写成式(1.2.6)的形式,且其绝对误差限为 $\frac{1}{2}\times 10^{m-n}$,则称近似值 x^* 具有 n 位有效数字.

根据以上定义可知,π 的近似值 3.14 具有三位有效数字,而近似值 3.141 6 具有五位有效数字,将 $x=0.092\ 53$ 舍成 $x^*=0.092\ 5$ 后具有三位有效数字,而不能说它具有五位有效数字.也就是说,有效数字是从第一位不等于零的数算起的,而与小数点后有多少位数字没有直接关系.另外,一个准确数的有效位数,应该说是有无穷多位.例如 $\frac{1}{8}=0.125$ 是准确值,它有无穷多位有效数字.最后,在有效位数的意义下,形如 6.32 和 6.320 00 的精确度是不同的,前者表示具有三位有效数字,其绝对误差不超过 $\frac{1}{2}\times 10^{-2}$,而后者具有六位有效数字,其绝对

误差不超过$\frac{1}{2}\times10^{-5}$.

最后给出绝对误差、相对误差和有效数字三者之间关系的定理.

定理 1.2.1 设准确值 x 的近似值 x^* 可以写成式(1.2.6),其相对误差为 $E_r^*(x)$,则 x^* 的有效位数与 $E_r^*(x)$ 具有以下关系:

(1) 若 x^* 有 n 位有效数字,则

$$|E_r^*(x)|\leqslant\frac{5}{a_1}\times10^{-n}. \tag{1.2.8}$$

(2) 若 $|E_r^*(x)|\leqslant\frac{5}{a_1+1}\times10^{-n}$,则 x^* 至少具有 n 位有效数字.

证明:由式(1.2.6)知

$$a_1\times10^{m-1}\leqslant|x^*|<(a_1+1)\times10^{m-1},$$

于是,若 x^* 有 n 位有效数字,则

$$|x^*-x|\leqslant\frac{1}{2}\times10^{m-n},$$

所以

$$|E_r^*(x)|=\left|\frac{x-x^*}{x^*}\right|\leqslant\frac{\frac{1}{2}\times10^{m-n}}{a_1\times10^{m-1}}=\frac{5}{a_1}\times10^{-n}.$$

反之,若 $|E_r^*(x)|\leqslant\frac{5}{a_1+1}\times10^{-n}$,则

$$\begin{aligned}|x-x^*|&\leqslant|E_r^*(x)|\cdot|x^*|\\&<\frac{5}{a_1+1}\times10^{-n}\times(a_1+1)\times10^{m-1}\\&=\frac{1}{2}\times10^{m-n}.\end{aligned}$$

故 x^* 至少具有 n 位有效数字.

另外,当 m 一定时,对于具有 n 位有效数字的近似值 x^* 而言,n 越大则绝对误差越小,反之亦然.

1.2.2 避免误差增大的若干原则

解决一个实际问题往往有多种不同的算法,而其中每一步计算又

都可能产生误差.因此,用不同算法计算的结果其精度是不同的:有些算法将使其各计算步所产生的误差累积成更大的误差,也有些算法其各步所产生的误差可以互相抵消乃至减少.因而,人们自然希望设计出计算量小而且精度较高的算法.为此,给出设计数值计算方法应该注意的若干原则.

(1) 尽量避免两个相近数进行减法运算.

如果对两个相近数进行减法运算,将造成有效数字的严重损失,亦即使相对误差急剧增加.为防止上述误差的迅速增加,应考虑将原有算式进行适当改变,而采用另一与之等价的算法进行计算.

例如,$\cos 2^\circ \approx 0.999\ 4$,具有四位有效数字,而 $1-\cos 2^\circ \approx 1-0.999\ 4=0.000\ 6$,却至多具有一位有效数字,其相对误差限为 $\frac{1}{12}$.

如在此基础上计算 $\frac{1-\cos x}{\sin x}$,其中 $x=2^\circ$,则有

$$\frac{1-\cos x}{\sin x} \approx \frac{0.000\ 6}{0.034\ 9} \approx 0.017\ 2. \tag{1.2.9}$$

如果用其等价形式 $\frac{\sin x}{1+\cos x}$ 进行计算,则有

$$\frac{1-\cos x}{\sin x}=\frac{\sin x}{1+\cos x} \approx \frac{0.034\ 9}{1.999\ 4} \approx 0.017\ 5. \tag{1.2.10}$$

注意到 $\frac{1-\cos x}{\sin x}=\tan \frac{x}{2} \approx 0.017\ 5$,显然采用式(1.2.10)进行计算的精度要高于式(1.2.9)的计算结果.

(2) 简化计算步骤,以减少算术运算的次数.

减少算术运算的次数,不但可以提高计算速度,而且有可能减小计算过程中的累积误差.

例如,计算多项式 $p_n(x)=\sum_{k=0}^{n} a_k x^k$ 的值.如果直接计算 $a_k x^k$ 后,再对各项求和,则一共需做 $\frac{1}{2}n(n+1)$ 次乘法和 n 次加法运算.但若按下列递推方法

$$\begin{cases} u_n = a_n, \\ u_k = xu_{k+1} + a_k \quad (k = n-1, n-2, \cdots, 1, 0) \end{cases} \tag{1.2.11}$$

进行计算，则只需 n 次乘法和 n 次加法就可以计算出 u_0，亦即 $p_n(x)$ 的值. 这就是著名的秦九韶算法.

(3) 防止出现机器零和数据溢出现象，并保证某些重要的物理量不被吃掉.

在计算机上进行数值运算时，当某些中间步的数值很小以致于计算机显示不出来时，就会将其赋为零值(机器零)；而当中间结果特别大时，如超出了计算机所能表示的数值范围，则会发生溢出现象. 因此，为保证计算结果的精度，常需在计算中采取一定的措施，使计算机能正常运行，并给出合理的计算结果.

例如，若计算的有效数字范围是 $2^{-32} \sim 2^{32}$，则 $e^{-26}/10^{-8}$ 的计算解为 0，这是由于 e^{-26} 的机器解为零值. 而如果改用 $[e^{-13}/10^{-4}]^2$ 来计算，则可得到较准确的解 $5.109\ 08 \times 10^{-4}$.

又如，若计算机所能表示的有效数字为十位，那么 $10 + 10^{14} - 9.999\ 999\ 996 \times 10^{13}$ 与 $10 + (10^{14} - 9.999\ 999\ 996 \times 10^{13})$ 的计算结果分别为 40000 和 40010，这是由于前者在计算中因有效数字位数的原因，而吃掉了第一个量 10，因此给计算结果带来了较大的误差.

(4) 计算函数值的坏条件判别法.

考虑定义域 $[a, b]$ 上的一阶连续可微函数 $f(x)$. 如果在计算过程中 x 的机器值 $x^* \neq x$，则函数值 $f(x)$ 被 $f(x^*)$ 所代替，其绝对误差为

$$Ef(x) = f(x) - f(x^*) = f'(\xi)E(x), \xi \text{ 介于 } x \text{ 与 } x^* \text{ 之间}.$$

如果在 $[a, b]$ 上恒有 $|f'(x)| \leqslant 1$，则有 $|Ef(x)| \leqslant |E(x)|$，即自变量的微小变动引起的函数值的变动更小，这正是我们所希望的. 但实际情况并不能保证 $|f'(x)| \leqslant 1$ 恒成立，如果存在某一点 $\bar{x} \in [a, b]$，使得 $|f'(\bar{x})|$ 很大，我们就说 $f(x)$ 在 $\bar{x}$ 点的计算是坏条件的.

例如，对函数 $f(x) = \frac{1}{n}\sin(n^2 x)$，有 $f'(x) = n\cos(n^2 x)$. 故当 n 很大时，在点 $x = 0$ 的计算是坏条件的.

对于大多数问题来讲,其计算效果是用相对误差衡量的.若 $x\neq 0$, x^* 充分接近 x,则当

$$C_{p(x)}=\left|\frac{E_r f(x)}{E_r(x)}\right|=\left|\frac{x\Delta f}{f\Delta x}\right|\approx\left|\frac{xf'(x)}{f(x)}\right|$$

很大时,称函数 $f(x)$在点 x 的计算在相对误差意义下是坏条件的.这里 $\Delta f=f(x+\Delta x)-f(x)$.

例如,对函数 $f(x)=\ln x$,有

$$C_{p(x)}\approx\left|\frac{1}{\ln x}\right|,$$

故当 x 很接近于 1 时,计算是坏条件的.

1.3 数值算法的稳定性和收敛性

1.3.1 计算机的浮点舍入误差

众所周知,大多数的数学运算都是建立在诸如实数集这样的具有一定稠密性和连续性的不可数集之上的,以这类集合为定义域的多项式函数一般具有较好的连续性.

但是,正如上节中曾经提到的,计算机中的数值是有一定的范围限制的,在计算机的运算过程中,会出现机器零及溢出等现象.不仅如此,由于计算机上数据存贮方式的关系,任何一台计算机所能表示的数都是有限的、残缺不全和离散的.因此,由计算机所表示的数与其实际值之间常有一定的误差.为此,需要对计算机中数据的表示方式做一介绍,以保证数值计算的准确性.

在计算机中,每个数都是由有限位的二进制数字组成,因而实数系 **R** 是由它的一个很小的离散子集来代替的.也就是说,这个子集只不过是有理数集合的一个小子集,它包含的每一个数的浮点形式由正负号、确定小数点位置的阶码及小数形式的尾数三部分组成.因此在计算机上,任何一个二进制、具有 t 位有效数字的实数 x 总可以表示成

$$x=\pm\left(\frac{d_1}{2}+\frac{d_2}{2^2}+\cdots+\frac{d_t}{2^t}\right)\times 2^l. \tag{1.3.1}$$

其中 d_i 为 0 或 1. 这里 2^l 称为指数部分，l 称为阶码，$d_1, d_2, \cdots, d_t$ 称为尾数. 此外，由于计算机字长的限制，阶码 l 也是有限制的，即满足 $L \leqslant l \leqslant U$，这里 L 和 U 分别称为阶码 l 的下界和上界. 因此，计算机所能表示的全体数的集合被称为计算机的浮点数系，记作 $F(2, t, L, U)$. 易知，尾数部分的 t 越大，所能表示的数的精度越高，故 t 也称为数系 $F(2, t, L, U)$ 的精度.

可以验证，当进行四则运算时，由于计算机字长的限制，通常会产生舍入误差. 而一般的数值计算问题都需要进行大量的四则运算，舍入误差的积累有时会大大影响计算结果的精确度. 因此，必须考虑这些误差在计算过程中是如何传播的，以及是否对最终的计算结果产生影响，影响的程度如何等问题. 这就需要对计算格式进行“舍入误差分析”，并给出算法的数值稳定性及收敛性估计.

1.3.2 数值稳定性

定义 1.3.1 对于某一给定的算法，如果在运算过程中舍入误差在一定条件下能够得到控制，不会对计算结果产生较大的影响，则称该算法是数值稳定的；否则，如果舍入误差得不到有效的控制，使得计算结果与问题的理论解之间产生较大的偏差，则称该算法是数值不稳定的.

由以上定义我们发现，数值稳定的算法一定满足以下性质：原始数据的微小改变只能导致数值计算结果的微小变化. 也就是说，如果原始数据的误差为 ε_0，并且在计算过程中的其他误差都是由 ε_0 引起的，那么对于一个稳定的数值计算方法而言，其计算结果的相对误差至多是 ε_0 相对于原始数据的相对误差的同阶无穷小量.

作为数值稳定性的一个必要条件，上述性质通常被看作是判定某个数值计算方法是否稳定的一个准则，为了保证方法的数值稳定性，必须在设计算法时着重注意这个问题.

此外，为了具体描述误差的传播与算法稳定性之间的关系，我们有以下关于误差传播速度的定义.

定义 1.3.2 设某一给定算法在执行到第 k_0 步时的误差为 e_{k_0}，并且在不引入其他误差的条件下，当执行到第 k 步（$k>k_0$）时，由 e_{k_0} 引起的误差为 e_k，那么

(1) 如果存在与 k 无关的常数 $C>0$，使得

$$|e_k|\approx C(k-k_0)|e_{k_0}| \quad (k=k_0+1,k_0+2,\cdots), \tag{1.3.2}$$

则称误差的增长是线性级的；

(2) 如果存在大于 1 的常数 M，使得

$$|e_k|\approx M^{(k-k_0)}|e_{k_0}| \quad (k=k_0+1,k_0+2,\cdots), \tag{1.3.3}$$

则称误差的增长是指数级的.

注意到，由于误差的传播是不可避免的，因此能否控制误差的传播，就成为算法稳定性的决定性条件. 通常情况下，误差按线性级增长的算法是数值稳定的，误差按指数级增长的算法是数值不稳定的. 另外，如果式(1.3.3)中的 $M\in(0,1)$，即误差是按指数级减小的，则算法也是数值稳定的.

例 1.3.1 为了构造数列 $\left\{\frac{1}{3^n}\right\}_{n=0}^{\infty}$，可以采用两种不同的数值计算格式：

$$x_0=1, x_n=\frac{1}{3}x_{n-1} \quad (n=1,2,\cdots) \tag{1.3.4}$$

和

$$\begin{cases} y_0=1, y_1=\dfrac{1}{3}, \\ y_n=\dfrac{10}{3}y_{n-1}-y_{n-2} \quad (n=2,3,\cdots) \end{cases} \tag{1.3.5}$$

分别进行计算，并得到两个不同的数列 $\{x_n\}_{n=0}^{\infty}$ 和 $\{y_n\}_{n=0}^{\infty}$.

事实上，若令 $y_n=\lambda^n(\lambda\neq 0)$，则由式(1.3.5)知，$\lambda$ 满足

$$\lambda^2-\frac{10}{3}\cdot\lambda+1=0,$$

上述方程的两个根分别为 $\frac{1}{3}$ 和 3，故二阶线性递推方程(1.3.5)的通解