普通高等教育“十一五”国家级规划教材

计算方法（第2版）

——算法设计及其MATLAB实现

王能超　编著

华中科技大学出版社
http://www.hustp.com

普通高等教育“十一五”国家级规划教材

计算方法

——算法设计及其MATLAB实现

（第2版）

王能超　编著

华中科技大学出版社
中国·武汉

内 容 简 介

本书是计算方法的入门教材，旨在通过一些基本的数值方法来探究数值算法设计的基本技术，诸如缩减技术、校正技术、松弛技术与二分技术等.

本书追求简约，数值算法的设计与分析尽量回避烦琐的数学演绎；本书追求统一，所提供的算法设计技术囊括了快速算法与并行算法等高效算法的设计；本书追求新奇，算法的设计机理扎根于博大精深的中华文化. 讲授本书的基本内容约需 36～40 课时.

本书可作为理工科院校非数学专业，特别是计算机专业的教材，也可供从事科学计算的科技工作者参考.

图书在版编目(CIP)数据

计算方法——算法设计及其 MATLAB 实现(第 2 版)/王能超　编著. —武汉：华中科技大学出版社，2010.12

ISBN 978-7-5609-6700-4

Ⅰ. 计…　Ⅱ. 王…　Ⅲ. ①计算方法-教材　②计算机辅助计算-软件包，MATLAB　Ⅳ. ①O241　②TP391.75

中国版本图书馆 CIP 数据核字(2010)第 212365 号

计算方法——算法设计及其 MATLAB 实现(第 2 版)　　王能超　编著

责任编辑：徐正达
封面设计：潘　群
责任校对：张　琳
责任监印：熊庆玉
出版发行：华中科技大学出版社(中国·武汉)
武昌喻家山　　邮编：430074　　电话：(027)87557437
录　　排：武汉佳年华科技有限公司
印　　刷：湖北新华印务有限公司
开　　本：710mm×1000mm　1/16
印　　张：17
字　　数：343 千字
版　　次：2010 年 12 第 2 版第 1 次印刷
定　　价：26.80 元

追求简易是中华文化的一个重要特色.关于"简易",我国古代经典《周易》有如下精辟的论述:

易则易知,简则易从.
易知则有亲,易从则有功.
有亲则可久,有功则可大.
可久则贤人之德,可大则贤人之业.

何谓"简易"?"易",是指所讲的道理要易于理解;"简",是指所教的方法要易于掌握.

道理易于理解就会使人亲近,彼此亲近就会持久;方法易于掌握才能收到功效,讲究功效就能壮大.

因此,追求简易是科学工作者的重要品德,具备这一品德才能成就伟大的事业.

——摘自本书第一版作者的"自序"

王能超教授的这本书，是一本富于哲学思想和科学方法论精神的著作。书中对各种各样的数值算法提出了几种富于概括性的设计思想和方法原则。这些思想和原则对从事研究和运用计算方法的科技工作者无疑会有深刻的启迪和指导作用。例如，书中所讲述的"缩减技术"、"校正技术"、"松弛技术"和快速算法及并行算法设计等，都是极为重要的方法原则，任何人如能精通并灵活运用这些方法原则，则不仅能圆满地解决实际计算问题，而且还可能有所创新，有所发展。①

徐利治

王能超教授是我国并行算法设计的先驱者之一，他在这方面有许多独特的重要贡献，其中最主要的是他巧妙地运用二分技术于并行算法设计……

王能超教授在并行算法设计中所以能取得巨大进展，主要由于他对算法设计的基本原理有深刻的研究。……正是由于这些独到的论点，他在并行算法设计的研究中取得巨大的、实质性的进展，推动了这门算法设计学的发展。②

程民德

① 引自徐利治先生 1988 年 12 月为《数值算法设计》(王能超著，华中理工大学出版社 1988 年出版)所写的"评审意见书"。

② 引自程民德先生 1992 年 5 月为《数值算法设计》所写的"评审意见书"。

再版前言

1986年夏，笔者应邀在全国计算数学教学研究会（桂林会议）上举办题为“数值算法设计”的学术讲座，讲座中提出用一套数学技术统一计算机上众多数值算法的想法，进而设想构建“数值算法设计学”的学科体系．这一体系撇开繁杂深奥的高等数学知识，其设计思想易于理解，设计方法易于掌握．参加讲座的学术界同行们对这一体系表现出浓厚的兴趣，敦促笔者撰写这方面的专著与教材．

这方面的专著《数值算法设计》于1988年由华中理工大学（现华中科技大学）出版社出版．该书当年获中南地区大学出版社协会优秀学术著作一等奖，1992年又获原国家教委科技进步奖．令笔者深为感动的是，该书得到学术界先辈们的赞赏．当代数学大师徐利治先生，1988年评价该书所提出的数值算法设计技术“都是极为重要的方法原则，任何人如能精通并灵活掌握这些方法原则，则不仅能圆满地解决实际计算问题，而且还可能有所创新，有所发展”．

在先辈们的激励下，多年来，笔者将数值算法学这一体系贯彻于计算方法的教学中，讲授对象包括本科生、硕士生与博士生，普遍反映教学效果良好．

本书第一版于2005年由高等教育出版社出版，第二版被批准为普通高等教育“十一五”国家级规划教材．因笔者已退休多年，加之近年来体质欠佳，修订工作一拖再拖．今年已是“十一五”的最后一年，看来没有退路了．为便于在修订出版过程中联系与配合，征得高等教育出版社有关同志的同意，现将书稿交付华中科技大学出版社．也算是“叶落归根”了．

俗说十年磨一剑．本书从酝酿到这次再版，前后历经20余年的时间，但笔者对这份书稿依然忐忑不安，其中可能还有不少谬误与欠妥之处．笔者将它奉献给广大的读者，希望得到更多的批评指教．

在本书问世之际，笔者特别感激华中科技大学出版社的鼎力支持，感谢鲁建华博士、鲁晓磊博士认真地编写了篇末的附录C《MATLAB文件汇集》．

谨将本书献给我的导师谷超豪教授，衷心感谢恩师多年的培养教育与亲切关怀。

王能超

2011年元旦

目　录

引论　数值算法设计的基本技术

0.1　算法重在设计

电子计算机的问世开创了现代科学的新时代. 随着计算机的广泛应用，科学计算正成为一种新的科学方法，它与科学实验、科学理论并列，构成科学方法论的三大组成部分.

今天，随着科学技术的蓬勃发展，实际课题的规模空前扩大，大型乃至超大型科学计算日益为人们所重视. 与此相适应，巨型计算机在科学计算中正扮演着越来越重要的角色. 计算机的更新换代强有力地推动着算法研究的深入，科学计算正面临蓬勃发展的新机遇.

0.1.1　算法设计关系科学计算的成败

计算机是一种功能很强的计算工具. 现代超级计算机的运算速度已高达每秒万亿次. 今天，运算速度每秒千万亿次的国产巨型机“天河一号”已经问世. 据预测，未来几年内甚至可能研制出运算速度每秒亿亿次的超级计算机系统. 计算机运算速度如此之快，是否意味着计算机上的算法可以随意选择呢?

举个简单的例子.

众所周知，Cramer 法则原则上可用来求解线性方程组. 用这种方法求解一个 n 阶方程组，要计算 $n+1$ 个 n 阶行列式的值，总共要做 $(n+1)n!(n-1)$ 次乘除操作. 当 n 充分大时这个计算量是惊人的. 譬如一个不算太大的 20 阶线性方程组，大约要做 10^{21} 次乘除操作，这项计算即使用每秒三千亿次的巨型计算机来承担，也得要连续工作

$$\frac{10^{21}}{3\times 10^{11}\times 60\times 60\times 24\times 365}\approx 100\ (\text{年})$$

才能完成. 当然这是完全没有实际意义的.

其实，求解线性方程组有许多实用解法(第 5 章与第 6 章). 譬如，运用人们熟悉的消元技术，一个 20 阶的线性方程组即使用普通的计算器也能很快地解出来. 这个简单的例子说明，**能否合理地选择算法是科学计算成败的关键.**

随着计算机的广泛应用与日益普及，算法设计的重要性正越来越为人们所认识.《计算机大百科全书》在其“算法学”词条中指出：“凡与计算机打交道，无不

研究各种类型的算法.”“**算法学是计算机科学最重要的内容,有的计算机学者甚至称,计算机科学就是算法的科学.**”

0.1.2 算法设计追求简单与统一

在知识“大爆炸”的今天,算法的数量也正以“大爆炸”的速度与日俱增,所涉及的文献著作数以千万计,形成浩繁的卷帙.面对这知识的汪洋大海,该如何进行有效的学习呢?许多有志于从事科学计算的青年工作者正为这门学科的知识庞杂所困扰.

学习和研究算法,应当从最简单的做起.每学一个专题,首先剖析一两个最简单、最初等的范例.基于这些极其简单的范例可以提炼出一般性的设计技术,这是个“点石成金”的过程.

要学好算法,关键在于将各种各样的具体算法进行归纳分类,并触类旁通.据我国最古老的一部算书《周髀算经》记载,上古先贤陈子教导后人,**学习算法要有“智类之明”,“问一类而以万事达”**.陈子这种“问一知万”的大智慧,是一服解读各种算法的灵丹妙药.

1976 年,英国著名数学家、菲尔兹奖获得者 Atiyah 在题为“数学的统一性”的演讲[①]中,突出地强调了数学的简单性和统一性.他说:“**数学的目的,就是用简单而基本的词汇去尽可能地解释世界.…… 如果我们积累起来的经验要一代一代传下去的话,我们就必须不断地努力把它们加以简化和统一.**”

算法设计追求简单和统一.后文将基于几个有趣的范例,提炼出算法设计的一条基本原理,进而概括出算法设计的几种基本技术.

0.1.3 Zeno 悖论的启示

古希腊哲学家 Zeno 在两千多年前提出过一个耸人听闻的命题:**一个人不管跑得多快,也永远追不上爬在他前面的一只乌龟.**这就是著名的 **Zeno 悖论.**

Zeno 在论证这个命题时采取了如下形式的逻辑推理:设人与龟同时同向起跑,如果龟不动,那么人经过某个时刻便能赶上它;但实际上在这段时间内龟又爬行了一段路程,从而人又得重新追赶.这样**每追赶一步所归结出的是同样类型的追赶问题,因而这种追赶过程永远不会终结.** Zeno 则据此断言人追上龟是“永远”不可能的.

Zeno 悖论的提出在古希腊学术界引起了轩然大波.在 Zeno 悖论面前,古代的数学逻辑显得无能为力,提供不出有力的论据予以驳斥,从而导致了人类文明史上“第一次数学危机”.

---

① M. Atiyah. 数学的统一性[J]. 数学译林,1980(1):36-43.

耐人寻味的是，尽管 Zeno 悖论的论断极其荒谬，但从算法设计的角度来看它却是极为精辟的.

Zeno 悖论将人龟追赶问题表达为一连串追赶计算的逐步逼近过程. 设人与龟的速度分别为 v_A 与 v_B，记 s_k 表示逼近过程的第 k 步人与龟的间距，另以 t_k 表示相应的时间，相邻两步的时间差 $\Delta t_k = t_k - t_{k-1}$. Zeno 悖论把人与龟追赶过程化归为一追一赶两项计算(图 0-1) 的重复.

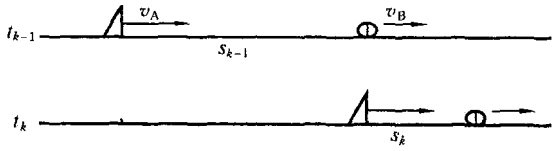

图 0-1

追的计算　先令龟不动，计算人追上龟所费的时间

$$\Delta t_k = \frac{s_{k-1}}{v_A} \tag{1}$$

赶的计算　再令人不动，计算龟在这段时间内爬行的路程

$$s_k = v_B \Delta t_k \tag{2}$$

追的计算与赶的计算都是简单的行程计算. 通过这两项计算加工得出的虽然同样是追赶问题，但问题的"规模" 已被大大地压缩了. 譬如，定义人与龟的间距 s_k 为追赶问题的**规模**，那么，经过上述两项运算手续加工后，问题的规模被压缩了 v_B/v_A 倍，即

$$s_k = \frac{v_B}{v_A} s_{k-1}$$

由于龟的速度 v_B 远远小于人的速度 v_A，压缩系数 v_B/v_A 很小，因而这项计算的逼近效果极为显著. 实际上，设 $s_0 = s$ 为已知，令 $t_0 = 0$(即从人龟起跑开始计时)，则按上述手续做不了几步，追赶问题的规模 s_k 就可以忽略不计，从而得出人追上龟实际所花费的时间 t_k. 这一算法

$$\begin{cases} s_k = \dfrac{v_B}{v_A} s_{k-1}, & k = 1,2,\cdots \\ s_0 = s \end{cases} \tag{3}$$

可称作 **Zeno 算法**，它是 Zeno 悖论的算法描述.

上述追的计算和赶的计算都是简单的行程计算，**Zeno 算法的设计思想是，将人与龟的追赶计算化归为简单的行程计算的重复.**

总之，上述 Zeno 算法的每一步，都是将原先的追赶问题加工成同样类型的追赶问题，但加工后的追赶问题，其规模(如人与龟的间距) 已被大大地压缩了. 这样，当规模变得足够小时，即可认为人已追上了龟，从而求得问题的解 —— 人追

上龟实际花费的时间.

这种反复缩减问题规模的设计策略称作**规模缩减技术**,简称**缩减技术**. 缩减技术是一种基本的算法设计技术. 下面介绍这种技术在直接法设计中的应用.

0.2 直接法的缩减技术

所谓直接法是这样一类算法,它通过有限步计算可以直接得出问题的精确解(如果不考虑舍入误差的话).

0.2.1 数列求和的累加算法

下述数列求和问题是人们所熟知的:

$$S = a_0 + a_1 + \cdots + a_n \tag{4}$$

这个计算模型有两个简单的特例. 当 $n = 0$ 即为一项和式 $S = a_0$ 时,所给计算模型就是它的解,这时不需要做任何计算. 这表明,对于数列求和问题,它的解是计算模型**退化**的情形. 又当 $n = 1$ 即计算两项和式 $S = a_0 + a_1$ 时,计算过程是**平凡**的,这时不存在算法设计问题.

现在基于这两种简单情形考察所给和式(4) 的累加求和算法. 设 b_k 表示前 $k+1$ 项的部分和 $a_0 + a_1 + \cdots + a_k$,则有

$$\begin{cases} b_0 = a_0 \\ b_k = b_{k-1} + a_k, \quad k = 1,2,\cdots,n \end{cases} \tag{5}$$

而计算结果 b_n 即为所求的和值

$$S = b_n \tag{6}$$

上述数列求和的累加算法,其设计思想是将多项求和(式(4)) 化归为两项求和(式(5)) 的重复. 而依式(5) 重复加工若干次,最终即可将所给和式(4) 加工成一项和式(6) 的退化情形,从而得出和值 S.

再剖析计算模型自身的演变过程. 按式(5) 每加工一次,所给和式(4) 便减少一项,而所生成的计算模型依然是数列求和. 反复施行这种加工手续,计算模型不断变形为

$$\underset{\text{(计算模型)}}{n+1 \text{ 项和式}} \Rightarrow n \text{ 项和式} \Rightarrow n-1 \text{ 项和式} \Rightarrow \cdots \Rightarrow \underset{\text{(所求结果)}}{1 \text{ 项和式}}$$

这里,符号"⇒" 表示重复施行两项求和的加工手续.

这样,如果定义和式的项数为数列求和问题的**规模**,则所求和值可以视作规模为 1 的退化情形. 因此,只要令和式的规模(项数) 逐次减 1,最终当规模为 1 时即可直接得出所求的和值. 这样设计出的算法就是累加求和算法(式(5)).

上述累加求和算法可以视作规模缩减技术的一个范例.

0.2.2　缩减技术的设计机理

许多数值计算问题可以引进某个实数，所谓问题的**规模**来刻画其“大小”，而问题的解则是其规模为足够小，譬如规模为1或0的退化情形. 求解这类问题，一种行之有效的办法是通过某种简单的运算手续逐步缩减问题的规模，直到加工得出所求的解. 算法设计的这种技术称作**规模缩减技术**，简称**缩减技术**.

缩减技术适用的一类问题是，求解这类问题的困难在于它的规模（适当定义）比较大. 针对这类问题运用缩减技术，就是设法逐步缩减计算问题的规模，直到规模变得足够小时直接生成或方便地求出问题的解.

缩减技术的设计机理可用“**大事化小，小事化了**”这句俗话来概括.

所谓“大事化小”，意即逐步压缩问题的规模. 在运用缩减技术时，“大事”是如何“化小”的呢？这个处理过程具有如下两项基本特征.

1° **结构递归**. “大事化小”是逐步完成的，其每一步将所考察的计算模型加工成同样类型的计算模型，因而这类算法具有明晰的递归结构.

2° **规模递减**. 每一步加工前后的计算模型虽然从属于同一类型，但其规模已被压缩了. 压缩系数愈小，算法的效率愈高.

再考察“小事化了”的处理过程. 所谓“小事化了”，是指当问题的规模变得足够小时即可直接或方便地得出问题的解.

“小事”是如何“化了”的呢？

对于某些计算模型，如前面讨论过的数列求和问题，它们的规模为正整数，而其解则是规模为0或1的退化情形. 这时只要设法使规模逐次减1，加工若干步后即可直接得出所求的解. 这里“小事化了”是直截了当的.

这样设计出的一类算法统称**直接法**. 前述数列求和的累加算法以及下面将要讲到的多项式求值的秦九韶算法都是直接法.

0.2.3　多项式求值的秦九韶算法

微积分方法的核心是逼近法. 多项式是微积分学中最为基本的一种逼近工具，因而多项式求值算法在微积分计算中具有重要意义.

设要对给定的 x 计算下列多项式的值：

$$P = a_0x^n + a_1x^{n-1} + \cdots + a_{n-1}x + a_n = \sum_{k=0}^{n} a_kx^{n-k} \tag{7}$$

由于计算每一项 a_kx^{n-k} 需做 $n-k$ 次乘法，如果先逐项计算 a_kx^{n-k}，然后再累加求和得出多项式的值 P，这种**逐项生成算法**所要耗费的乘法次数为

$$Q = \sum_{k=0}^{n}(n-k) \approx \frac{n^2}{2}$$

当 n 充分大时这个计算量是相当大的.

现在设法改进这一算法. 类似于数列求和计算,首先考察两个特例:当 $n=0$ 时,所给计算模型即为所求的解

$$P=a_0$$

这时不需要做任何计算;又当 $n=1$ 时,计算模型

$$P=a_0x+a_1$$

为简单的一次式,这时虽然需要进行计算,但不存在算法设计问题.

注意,当 $x=1$ 时多项式(7)便退化为和式(4),可以类比数列求和算法的设计过程讨论多项式求值算法的设计问题.

设将多项式的次数规定为多项式求值问题的规模,如果从式(7)的前两项中提出公因子 x^{n-1},则有

$$P=(a_0x+a_1)x^{n-1}+\sum_{k=2}^{n}a_kx^{n-k}$$

这样,如果算出一次式

$$v_1=a_0x+a_1$$

的值,则所给 n 次式的计算模型(7)便化归为 $n-1$ 次式

$$P=v_1x^{n-1}+\sum_{k=2}^{n}a_kx^{n-k}$$

的计算,从而使问题的规模减少了 1 次. 不断地重复这种加工手续,使计算问题的规模逐次减 1,则经过 n 步即可将所给多项式的次数降为 0,从而获得所求的解. 这样设计出算法 0.1.

算法 0.1 (秦九韶算法)

令 $v_0=a_0$,对 $k=1,2,\cdots,n$ 计算

$$v_k=xv_{k-1}+a_k \tag{8}$$

则结果 $P=v_n$ 即为所给多项式(7)的值.

容易看出,按递推算式(8)计算多项式(7)的值,总共只要做 n 次乘法,其计算量远比前述逐项生成算法的计算量小. 这是一种优秀算法.

这一优秀算法称作**秦九韶算法**. 它是我国南宋大数学家秦九韶(公元 13 世纪)最先提出来的. 需要注意的是,国外文献常称这一算法为 **Horner 算法**,其实 Horner 的工作比秦九韶晚了五六百年.

秦九韶算法说明,n 次式(7)的求值问题可化归为一次式(8)求值计算的重复. 设以符号"⇒"表示一次式的求值手续,则秦九韶算法的模型加工流程如下:

n 次式求值 ⇒ $n-1$ 次式求值 ⇒ $n-2$ 次式求值 ⇒ ⋯ ⇒ 0 次式求值

(计算模型) (计算结果)

0.3　迭代法的校正技术

上一节介绍了设计直接法的缩减技术. 缩减技术针对这样的问题，它的规模为正整数，而解则是规模足够小(通常规模为 0 或 1) 的退化情形. 这样，只要设法令规模逐次减 1，即可将计算模型逐步加工成解的形式. 这种加工过程可用“大事化小，小事化了”这句俗话来概括.

有些问题的“大事化小”过程似乎无法了结. Zeno 悖论强调人“永远”追不上龟正是为了突出这层含义. 这是一类无限的逼近过程，计算问题的规模通常是实数. 正如前述 Zeno算法所看到的，如果逼近过程的规模(适当定义) 按某个比例常数一致地缩减，那么，适当提供某个精度即可控制计算过程的终止. 这样设计出的算法通常称作**迭代法**.

0.3.1　Zeno 悖论中的“Zeno 钟”

Zeno 悖论所表述的人龟追赶问题其实是容易求解的. 设人与龟起初相距 s，两者速度分别为 v_A 与 v_B，则可列出方程

$$v_A t - v_B t = s \tag{9}$$

因此人追上龟实际所花费的时间

$$t^* = \frac{s}{v_A - v_B}$$

我们再运用所谓预报校正技术处理这个简单问题，以为将来求解一般非线性方程做准备.

设有解 t^* 的某个**预报值** t_0，希望提供校正量 Δt，使**校正值**

$$t_1 = t_0 + \Delta t$$

能更好地满足所给方程(9)，即尽可能准确地成立

$$v_A(t_0 + \Delta t) - v_B(t_0 + \Delta t) \approx s$$

注意到 v_B 是个小量，设校正量 Δt 也是个小量，上式舍弃高阶小量 $v_B \Delta t$，得

$$v_A(t_0 + \Delta t) - v_B t_0 = s \tag{10}$$

求解这个方程，所定出的校正值 $t_1 = t_0 + \Delta t$ 为

$$t_1 = \frac{s + v_B t_0}{v_A}$$

进一步视 t_1 为新的预报值重复施行上述手续求出新的校正值 t_2，依 t_2 再定出 t_3，反复施行这种预报校正手续，即可生成一个近似值序列 $t_1, t_2, \cdots$. 这就规定了一个迭代过程，其迭代公式为

$$t_{k+1} = \frac{s + v_B t_k}{v_A}, \quad k = 0,1,2,\cdots \tag{11}$$

Zeno 悖论所表述的逼近过程正是这种迭代过程,当 $k \to \infty$ 时,式(11)的迭代结果 t_k 将收敛到人追上龟所需的时间 t^*.

那么,Zeno 强调人"永远"追不上龟试图表达什么含义呢?

我们知道,任何形式的重复均可作为时间的量度. Zeno 在刻画人龟追赶过程时实际上设置了两个"时钟":一个是日常钟 t_k,其含义无须解释;Zeno 又将迭代次数 k—— 一追一赶过程(图 0-1)的重复次数设定为另一种"时钟",不妨将这一时钟称作 **Zeno 钟**. Zeno 钟 k 采取离散的计数方式,它仅仅取正整数值. Zeno 公式(11)表明,当 Zeno 钟 $k \to \infty$ 时人才能追上龟. Zeno 正是依据这一事实断言"人永远追不上龟".

再举一个有实用价值的迭代算法.

0.3.2 开方算法

四千多年前,在亚洲西南部的古巴比伦(现伊拉克境内)就已经萌发出数学智慧的幼芽. 古巴比伦数学取得了一系列重要成就,譬如制成了有关平方根的数表. 古巴比伦人制造开方表的方法难以考证,不过可以想象其计算方法必定相当简单.

现代电子计算机上又是怎样计算方根值的呢?

相对于加减乘除四则运算来说,开方运算无疑是复杂的. 人们自然希望将复杂的开方运算归结为四则运算的重复,为此需要设计某种算法.

给定 $a > 0$,求方根值 $\sqrt{a}$ 的问题就是要解方程

$$x^2 - a = 0 \tag{12}$$

这是个非线性的二次方程,从初等数学的角度来看,它的求解有难度. 该如何化难为易呢?

设给定某个预报值 x_0,我们希望借助于某种简单方法确定校正量 Δx,使校正值 $x_1 = x_0 + \Delta x$ 能够比较准确地满足所给方程(12),即有

$$(x_0 + \Delta x)^2 \approx a$$

假设校正量 Δx 是个小量,为简化计算,舍弃上式中的高阶小量 $(\Delta x)^2$,而令

$$x_0^2 + 2x_0\Delta x = a$$

这是关于 Δx 的一次方程,据此定出 Δx,从而对校正值 $x_1 = x_0 + \Delta x$ 有

$$x_1 = \frac{1}{2}\left(x_0 + \frac{a}{x_0}\right)$$

反复施行这种预报校正手续,即可导出**开方公式**

$$x_{k+1} = \frac{1}{2}\left(x_k + \frac{a}{x_k}\right), \quad k = 0,1,2,\cdots \tag{13}$$

从给定的某个初值 $x_0 > 0$ 出发,利用上式反复迭代,即可获得满足精度要求