高等学校教材

计算方法
——算法设计及其MATLAB实现

王能超 编著

高等教育出版社
HIGHER EDUCATION PRESS

内容提要

本书是从《计算方法》(人民教育出版社,1978年)一书几经改版而成的,各种版本都受到读者广泛的欢迎,累计已发行数十万册。这次再版在内容处理上有创新。本书坚持"简单的重复生成复杂"的理念,运用某种算法设计技术统一了各种数值算法,其设计原理容易理解,设计方法容易掌握。为便于读者自学,本书附加了"例题选解"以及"常用算法的MATLAB文件汇集"等有关材料。

本书可供本科、专科各类院校的不同专业作为普及计算方法知识的教材,亦可供工程技术人员阅读参考。

图书在版编目(CIP)数据

计算方法——算法设计及其MATLAB实现/王能超编著.北京:高等教育出版社,2005.1
ISBN 7-04-016060-9

Ⅰ.计… Ⅱ.王… Ⅲ.①电子计算机-算法设计②计算机辅助计算-软件包,MATLAB Ⅳ.①TP301.6②TP391.75

中国版本图书馆CIP数据核字(2004)第114877号

策划编辑 康兆华 **责任编辑** 康兆华 **市场策划** 陈 振 **封面设计** 王凌波
责任绘图 宗小梅 **版式设计** 张 岚 **责任校对** 康晓燕 **责任印制** 孔 源

出版发行	高等教育出版社	**购书热线**	010-64054588
社 址	北京市西城区德外大街4号	**免费咨询**	800-810-0598
邮政编码	100011	**网 址**	http://www.hep.edu.cn
总 机	010-58581000		http://www.hep.com.cn
经 销	新华书店北京发行所		
印 刷	北京铭成印刷有限公司		
开 本	787×960 1/16	**版 次**	2005年1月第1版
印 张	15.25	**印 次**	2005年1月第1次印刷
字 数	280 000	**定 价**	17.90元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换。

物料号:16060-00

自　序

我们迎来了新世纪的黎明。21世纪犹如喷薄欲出的一轮红日呈现在人们面前。在这世纪更迭的神圣时刻，人们正满怀激情地发问：新世纪的科学会有什么样的新风采？

新世纪呼唤新科学

回顾已经过去的20世纪，谁也不会否认这样的事实：这个世纪的科学技术取得了惊人的成就，在科学史上谱写了辉煌的篇章。

在20世纪末，具有远见卓识的学者们意识到人类科学正面临着一个新的转折点。1984年，在诺贝尔奖获得者P.W.Anderson、M.Gell-Mann和K.S.Arrow等人的支持下，美国一批从事物理、经济、生物和计算机等学科研究的学者们创建了著名的桑塔费研究所SFI，试图探索未来科学的思维方式。SFI首任所长G.A.Cowan尖锐地指出：

“通往诺贝尔奖的堂皇道路，通常是由简化论和还原论的思维方式取得的，这就造成了科学上越来越多的碎裂片，而真实的世界要求我们用更加整体的眼光去看问题。”

SFI认为，从局部到整体必然会导致问题的复杂化，他们将未来科学命名为“复杂科学”。

科学研究离不开数学。伽里略有句名言：宇宙这本大书是用数学语言写成的。数学是科学的世界语。数学是协助科学探索的有力工具。缺乏数学思维指导的科学活动必然是盲目的、肤浅的。

众所周知，近代科学诞生于有着雄厚数学基础的17世纪。17世纪被誉为数学史上“天才的世纪”。这个世纪取得了三项伟大的数学成就：对数方法将困难的乘除运算化归为简易的加减运算；解析几何方法将玄奥的几何命题化归为浅显的代数命题；微积分方法用简单的代数多项式逼近一般的复杂函数。总之，数学思维的基本特征是将复杂转化为简单。

数学的目的是追求简单，然而当今的科学却片面地强调“复杂”，炫耀“复杂”。复杂与简单果真是不可调和吗？数学与当今科学能够相互沟通交融吗？

新科学需要新观念

什么是“复杂”？在某种意义上，复杂意味着知识的缺乏。一个命题，在没有解决之前是复杂的，解决之后它就变得简单了；一种规律，在没有掌握之前是玄奥的，掌握之后就变得浅显了。

我们正处在计算机时代。计算机的广泛应用日益改变着世界的面貌，也深刻地影响着人们的思维方式。计算机的工作原理是简单的，它只会做加减乘除的二进制运算，然而依赖各式各样的算法，它却能承担极其复杂的计算任务。计算机上的算法究竟是怎样设计的呢？

本书以大量的算法案例透析出一个发人深省的事实：算法设计的基本理念是通过简单的重复生成复杂，或者说，将复杂转化为简单的重复。在算法设计过程中，重复就是力量。这里所说的“重复”本质上是某种演化过程。

我们深信，复杂的事物可能具有简单的演化机制，而简单的模型则可能具有复杂的演化形态。如果发展到极致，复杂与简单就可能合于一体：极端复杂就可能等于极端简单。

复杂和简单是相互变通的。它们两者是矛盾的统一体。

新观念仰赖大智慧

所谓“大智慧”，是那些历经数千年考验的人类智慧。

在2500年前，古希腊哲学家Zeno提出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。这个“人龟追赶问题”就是所谓的Zeno悖论，它的提出触发了人类历史上“第一次数学危机”。

表面上看Zeno悖论极端荒谬，其实它同时又是极端深刻。本书揭露出一个奇妙的事实，剖析Zeno悖论所归纳出的几种算法设计技术，可用来设计出大量的计算机算法。

纵观数值算法学的发展史，中华先贤做出过杰出的贡献。祖冲之的圆周率计算曾千年称雄于世界。刘徽的千年古术至今仍熠熠生辉，指引人们探索新的计算领域。中华民族是个智慧的民族。算法设计是中华数学的强项。科学探索要国际化，首先要民族化。为要同国际先进水平接轨，先要向智慧的中华先贤讨教。在中华民族正和平崛起的今天，我们肩负着伟大而神圣的历史使命：复兴先贤伟业，重振中华雄风！

王能超

2004年9月18日

《周易》论“简易”

算法设计的基本思想是简朴的。算法设计的基本技术是简单的。算法设计追求简易。

追求简易是中华传统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。

解释这番话的含义。

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一项重要品德，具备这一品德才能成就伟大的事业。

前　言

自1978年以来，作者在高等教育出版社多次出版有关计算方法(数值分析)的教材，其中包括：

[1] 计算方法(1978年)

[2] 数值分析简明教程(1984年;2003年第二版)

[3] 计算方法简明教程(2003年)

这项工作是从1977年开始的。这一年恢复了高考，高等院校又焕发出生机。这一年的年底，我受命编写计算方法的“统编教材”。当时困难很多：给的学时少，仅仅提供22学时；编写时间短，要求在半年之内完成。压力变成了动力，一份结构紧凑、内容简约的教材如期“逼”了出来。

次年(1978年)5月中旬在上海召开了这份教材的审稿会，由上海交通大学孙增光教授主审，参加评审的有清华大学孙念增教授、西安交通大学游兆永教授等知名学者。与会专家对教材给予了充分的肯定。会后不久教材[1]就面世了。

为充实教材[1]，1984年又出版了教材[2]。该书以泰勒展开作为主线，被同行们评价为“泰勒公式包打天下”。该书荣获原国家教委优秀教材二等奖。

纵观形形色色的众多算法，其设计机理均可概括为“简单的重复生成复杂”。基于这一理念又编写出教材[3]。这份教材回避了泰勒展开方法，代之以几种简约的算法设计技术。内容更为简明，方法更易掌握。该书基于这些技术统一了众多常用算法，并自然地跨越到高效算法设计的学科前沿。

人类已进入计算机时代，计算机的广泛应用迫切要求普及有关计算方法的基本知识。编写本书的目的是为了进一步适应形势发展的需要，同时作者也希望为自己20余年计算方法(数值分析)的教材探索做个小结。

近两三年来作者明显地加快了教材编写工作的进度，这主要归功于高等教育出版社领导和有关编辑同志的鼎力支持，作者对此表示衷心的感谢！在此还要感谢鲁晓磊同志协助编写了篇末的附录MATLAB文件。

“谁言寸草心，报得三春晖。”作者谨将本书献给导师谷超豪教授，感谢他多年的培养、教育和关怀！

王能超

2004年8月28日

目　录

引　论

0.1 算法重在设计

0.1.1 计算机开创了现代科学的新时代

纵观上下数千年的科学史，科学的发展大致经历了古代科学、近代科学和现代科学三个历史阶段.

在遥远的古代，虽然人们在长期的社会实践中积累了不少知识，但这些知识是零碎的、不系统的和没有经过严格论证的.古人所获取的知识大都表现为经验性的总结或猜测性的思辨，其研究方法实际上是不科学的.在这个意义上，古代科学只是科学的萌芽，还不是真正的科学.

近代科学蓬勃兴起于17世纪，其奠基工作从Galileo(伽利略，1564—1642)开始，而由Newton(牛顿，1642—1727)所完成.近代科学方法强调实验和理论的紧密结合，即以实验的事实(数据和资料)为依据，通过严密的论证(数学推理)形成系统的理论.这种科学方法促进了科学的繁荣与发展.

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，科学计算正逐步上升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.

在今天，随着科学技术革命的蓬勃发展，实际课题的规模空前扩大，所谓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，科学计算正处于蓬勃发展的新时代.

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.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，行列式解法的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，这个计算量是相当惊人的.譬如一个20阶不算太大的方程组，大约要做 10^{21} 次乘除操作.这项计算即使用每秒

3千亿次的巨型计算机来承担，也得要连续工作

$$\frac{10^{21}}{3\times10^{11}\times60\times60\times24\times365}\approx100\ (\text{年})$$

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(参看本书第五章与第六章).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.这个简单的案例说明，**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."

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，如何才能进行有效的学习呢？许多有志于从事科学计算的青年科学工作者正为这门学科的知识庞杂所困扰.

1976年，英国著名数学家Atiyah在他就任伦敦数学学会主席时，发表了题为"数学的统一性"的演讲[①].在这次演讲中，他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多地解释世界.……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.后文将基于一个有趣的范例，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.

0.1.2 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，如图0.1所示.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掀起轩然大波.在Zeno悖论面前，古代的

① M Atiyah.数学的统一性.数学译林，1980年第1期

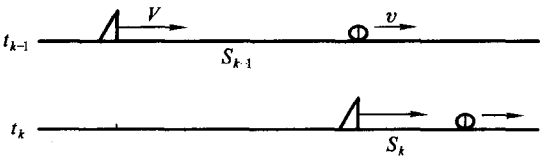

图 0.1 人龟追赶过程

数学逻辑显得无能为力,提供不出有力的论据给予驳斥,从而导致了人类文明史上"第一次数学危机".

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步的逐步逼近过程.设人与龟的速度分别为 V 与 v,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 悖论把人与龟的追赶问题分解为一追一赶两个过程(参看图 0.1):

追的过程 先令龟不动,计算人追上龟所费的时间

$$\Delta t_k = \frac{S_{k-1}}{V} \tag{1}$$

赶的过程 再令人不动,计算龟在这段时间内爬行的路程

$$S_k = v\Delta t_k \tag{2}$$

无论是追的过程还是赶的过程,它们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,设以人与龟的间距 S_k 定义为追赶问题的**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/V 倍:

$$S_k = \frac{v}{V}S_{k-1}$$

由于龟的速度 v 远远小于人的速度 V,压缩系数 v/V 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

$$\begin{cases} S_k = \dfrac{v}{V}S_{k-1}, \quad k = 1,2,\cdots \\ S_0 = S \end{cases} \tag{3}$$

可称为 **Zeno 算法**,它是 Zeno 悖论的算法描述.

上述追的过程(1)和赶的过程(2)都是简单的行程计算,Zeno 算法(3)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,参看图 0.1)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为**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.现在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:

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

上述数列求和的累加算法,其设计思想是将多项求和(4)化归为两项求和(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**,则所求和值可以视为规模为 1 的退化情形.因之,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值.这样设计出的算法就是累加求和算法(5).

上述累加求和算法可以视为规模缩减技术的一个范例.

0.2.2 缩减技术的设计思想

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为**规模缩减技术**,简称**缩减技术**.

缩减技术所适用的一类问题是,求解这类问题的困难所在是它的规模(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.

缩减技术的设计思想可用**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.压缩系数愈小则算法的效率愈高.

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述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：

$$P = a_0 x^n + a_1 x^{n-1} + \cdots + a_{n-1} x + a_n = \sum_{k=0}^{n} a_k x^{n-k} \tag{7}$$

由于计算每一项 $a_k x^{n-k}$ 需做 $n-k$ 次乘法，如果先逐项计算 $a_k x^{n-k}$，然后再累加求和计算多项式的值 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

$$P = a_0 x + a_1$$

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.

注意到当 $x=1$ 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)的前面两项中提出公因子 x^{n-1}，则有

$$P = (a_0 x + a_1) x^{n-1} + \sum_{k=2}^{n} a_k x^{n-k}$$

这样，如果算出一次式

$$v_1 = a_0 x + a_1$$

的值，则所给计算模型(7)便化归为 $n-1$ 次式

$$P = v_1 x^{n-1} + \sum_{k=2}^{n} a_k x^{n-k}$$

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.这样设计出的算法是算法 0.1.

算法 0.1 令 $v_0 = a_0$，对 $k = 1, 2, \cdots, n$ 计算

$$v_k = x \cdot v_{k-1} + a_k \tag{8}$$

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少.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.需要提醒注意的是,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:

$$\begin{matrix} n\text{ 次式求值} \\ \text{(计算模型)} \end{matrix} \Rightarrow n-1\text{ 次式求值} \Rightarrow n-2\text{ 次式求值} \Rightarrow \cdots \Rightarrow \begin{matrix} 0\text{ 次式求值} \\ \text{(计算结果)} \end{matrix}$$

0.2.4 算法的框图描述

同人工手算的计算方法意义不同,将要研究的算法是为计算机提供的,因此,解题方案中的每个细节都必须加以定义,并且要完整地描述整个计算过程.这就是说,所谓计算机算法不仅仅是单纯的数学公式,而是指解题方案的准确而完整的描述.

刻画计算流程有多种方式,本书常用框图直观地显示算法的全貌.

本书将使用两种形式的框:一种是叙述框□,计算公式就填在这种框内;另一种是检查框◇,它表示算法的判断部分.检查框有两个出口,究竟选择哪个出口,要视框内的判断条件是否满足而定.

今后所有的框图均以㊙框标志计算过程开始启动,而以㊛框表示计算过程的最终结束.另以箭头"→"指明各框执行的顺序.

现在考察秦九韶算法的计算流程.

按秦九韶算式(8)进行计算,每求出一个"新值"v_k 以后,"老值"v_{k-1} 便失去继续保存的价值,因此可将新值 v_k 存放在老值 v_{k-1} 所占用的单元内.这样,只要设置一个单元 v 进行累算,而将算式(8)表为如下动态形式:

$$v \Leftarrow x \cdot v + a_k, \quad k = 1, 2, \cdots, n$$

这里箭头"⇐"用以刻画赋值手续.执行这组算式之前,应先送初值 a_0 到单元 v 中:

$$v \Leftarrow a_0$$

图 0.2 描述了多项式(7)求值的秦九韶算法,其中:

框 1 为**准备部分**.单元 v 中送初值 a_0,单元 k 中送计数值 1.

框 2 为**计算部分**.每循环一次,单元 v 中的老值 v_{k-1} 为新值 v_k 所替换.

框 3 为**控制部分**.检查单元 k 中的计数值以判断循环应否结束.当计数值为 n 时输出 v 中结果,否则转框 4.

框 4 为**修改部分**.修改单元 k 中计数值,然后转框 2 再做下一步的计算.

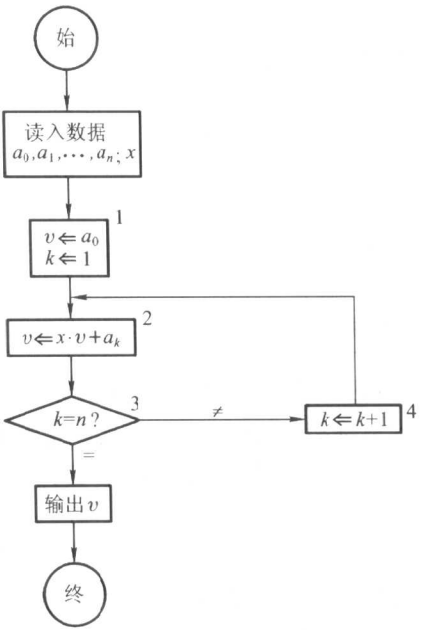

图 0.2 秦九韶算法的计算流程

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,它的解是规模足够小(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,其问题的规模通常是实数.正如前述 Zeno 算法所看到的,如果所设计出的逼近过程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 与 v,则容易列出方程