

微型计算机算法与程序

扩展 BASIC

上海科学技术文献出版社

微型计算机算法与程序

· 扩展 BASIC ·

王 林 张晓卫 编著

上海科学技术文献出版社

内 容 简 介

本书通俗地介绍了微型计算机算法与程序。内容包括: 扩展 BASIC 语言简介; 复数运算; 代数方程与超越方程; 插值、样条插值与数值微商; 线性代数计算与矩阵计算; 数值积分; 常微分方程的解法以及数理统计方法等。每种算法都给出了算法功能, 计算方法简介, 使用说明, 标准算法子程序, 例题和例题的主程序, 用户能方便地使用和套改。

本书可供使用微处理机和微型计算机的科技人员参阅, 也可供大专院校师生参考。

前 言

科学技术和国民经济的发展要求从事各类专业的科技人员,经济管理和金融商业人员等都能够使用计算机。

使用计算机的一项重要工作是编写计算问题的程序。编写程序所使用的语言主要有面向机器的机器语言和面向问题的高级语言。机器语言是由机器的数字代码组成,用它编写的程序叫手编程序。手编程序不便于阅读和检查,不同型号的机器,其机器语言不同,不能相互通用,因而手编程序经常是由专业的程序员来编写的。显然这种语言的局限性很大。高级语言的出现,使得编制程序一般不必需要专业程序员来编写。高级语言面向问题,与我们平常书写的数学公式相近,形式直观,概念明确,便于掌握。用同种语言编写的程序,在具有该种语言编译程序的其它机器上都可以通用。

目前普遍使用的高级语言有 FORTRAN, ALGOL, BASIC, COBOL, PL/A, PASCAL 等。其中 BASIC 语言易于学习,便于较快掌握。特别是 BASIC 语言经过扩展以后形成的扩展 BASIC 语言,功能显著增强,能满足目前科学计算的需要。

扩展 BASIC 语言编译程序对用户程序边解释边执行,可以进行人机对话,这是它的特有的优点。程序输入过程中能即时发现错误,修改手续方便。扩展 BASIC 带有检查、显示出错类型、程序跟踪等手段,所以调试程序比其它高级语言简单、迅速。

鉴于扩展 BASIC 的功能和它特有的优点,我们将科学计算中的常用算法用扩展 BASIC 语言编写出程序,并全部在微处

理机上进行了实算。只要计算量是在机器内存容量内,字长在允许的范围内,即可由 BASIC 语言来求解。现将这些算法、程序汇编成册,供大家使用参考。

在每个算法中都分别给出了功能、计算方法简介、使用说明、算法的标准子程序、例题及例题的主程序。功能是指该计算方法的使用范围和解决问题的能力。在计算方法简介中,没有进行繁琐的数学论证和推导,仅介绍了算法本身的数学过程和计算公式。我们在写法上尽量简明清楚,它不同于计算方法的教科书,但又有本身的计算完整性,力求使读者方便地使用和套改。在使用说明中,详细介绍了输入输出参数,介绍了程序中所用的数据及它的输入格式,对主程序和辅程序进行了必要的说明。标准子程序是不需读者修改的,在使用过程中只需直接调用。读者只需按所计算的题目,根据例题主程序、辅程序的格式套改即可使用。

鉴于我们的水平所限,对书中的不足之处,请读者批评指正。

编 者

1983 年 10 月

目 录

绪 论	1
第一章 复数运算	26
一、复数的除法	26
二、 e^z (z 为复数)	28
三、复数的模	29
四、复数的平方根	31
五、复变量的三角函数	34
六、复变量的幂指函数	38
七、复变量的自然对数	40
第二章 求解代数方程与超越方程	44
一、二次、三次和四次方程的代数解法	45
二、二分法	58
三、弦截法	61
四、优选法	65
五、二次插值法(1)	69
六、二次插值法(2)	74
七、牛顿法	80
八、贝努利法	84
九、下降法解非线性方程组	90
十、线性插值法	94
十一、阻尼最小二乘法	98
第三章 插值与数值微商	110
一、线性插值	111
二、一元全区间不等距插值	114

三、一元三点等距插值、微商	117
四、一元三点不等距插值	120
五、一元三点不等距分段插值	123
六、一元三点不等距成组插值	126
七、埃特金插值	129
八、爱尔米特插值	133
九、第一种边界条件三次样条函数插值、微商与积分	136
十、第二种边界条件三次样条函数插值、微商与积分	145
十一、第三种边界条件三次样条函数插值、微商与积分	151
十二、有理函数插值	158
十三、分段有理函数插值	164
十四、二元不等距插值	169
第四章 线性代数计算	174
一、列主元高斯-约当消去法	176
二、高斯消去法	181
三、列主元高斯消去法	187
四、全主元高斯消去法	191
五、解三对角线型方程组的追赶法	196
六、克劳特分解法	200
七、正定对称方程组的平方根法	206
八、正定对称方程组的改进平方根法	211
九、线性对称方程组的分解法	216
十、高斯-赛德尔迭代法求解线性代数方程组	220
十一、赛德尔法解线性代数方程组	224
十二、高斯消去法求实行列式值	228
十三、求正定对称矩阵行列式的值	231
第五章 数值积分	235
一、梯形法求积分	237
二、定步长辛普生法求积分	239
三、变步长辛普生法求积分	242

四、辛普生法求成组积分	245
五、变步长辛普生法求二重积分	250
六、自适应辛普生法求积分	256
七、龙贝格法求积分(1)	262
八、龙贝格法求积分(2)	265
九、高斯-勒让德法求积分	270
十、高斯法求定积分	275
十一、高斯法求多重积分	277
十二、高斯-切比雪夫方法求积分	282
第六章 常微分方程组的数值解法	287
一、改进欧拉方法	288
二、定步长四阶龙格-库塔法(1)	292
三、定步长四阶龙格-库塔法(2)	295
四、定步长四阶龙格-库塔法(3)	298
五、变步长龙格-库塔法	302
六、变步长四阶单步方法	310
七、定步长五阶单步方法	316
八、变步长墨森方法	321
九、定步长哈明方法	328
十、定步长预报-校正法	333
十一、吉尔方法	338
十二、双边法	343
第七章 数理统计方法	349
一、单因素方差分析	350
二、双因素方差分析	355
三、一元线性回归及预测	362
四、多元线性回归	371
五、逐步回归分析	387
六、二次分段移动平均法	408
七、三次指数平滑预测法	412

绪 论

一、扩展 BASIC 语言简介

BASIC 这个词是英语 Beginner's All-purpose Symbolic Instruction Code (即初学者通用符号指令代码) 的缩写, 由于它简明易学而使用甚广。但过去的 BASIC 语言的解译程序简单, 因而对于解决较为复杂的运算问题在功能、精度和程序设计上都有许多困难。但由于 BASIC 语言所特有的许多优点和易学性, 使人们不断地扩充了 BASIC 语言的功能, 增加了矩阵和字符串运算的指令, 增加了许多控制语句, 程序调试语句和输入输出语句等等, 使 BASIC 语言的功能大大增强。近年来我国生产的 100 系列小型通用机上和国外引进的微处理机上 (例如 Z-80, TRS-80 等) 都配有扩展 BASIC 语言。

不同机种和不同文本的扩展 BASIC 语言略有些差异, 但基本内容和主要语句都是相同的。鉴于本书中的程序均在 Z-80 微处理机上进行实算并打印出结果, 因而我们将以 CROMEMCO 公司的 16K 扩展 BASIC 语言为基础, 介绍扩展 BASIC 语言, 读者掌握了该文本的扩展 BASIC 语言, 就已基本掌握了扩展 BASIC 语言。当使用不同机种或不同文本的扩展 BASIC 语言时, 只要看一下有关的说明书, 就可以顺利地工作了。

CROMEMCO 公司 16K 扩展 BASIC 是由 Shepardson 微处理机公司专门为 CROMEMCO 设计的, 由于充分利用了 Z-80 微处理机扩展的 158 条指令, 因而它具有很高的计算精度, 较好

的程序设计效能和较快的计算速度。

16 K 扩展 BASIC 的一个主要特点是利用 Z-80 微处理机特有的 BCD(二~十进制)算术指令进行高速的 14 位的算术运算。

16 K 扩展 BASIC 的半编译程序的编译工作是在程序每输入完一行后进行的,句法上的错误是输入后即时检查出来的,而不是在程序执行时才检查出来的。因此,引入和调试一个程序所化的时间显著地减少。它不仅给出错误信息,而且它重新把该错误语句显示出来,指出错误所在。

16 K 扩展 BASIC 还提供了一种跟踪程序的方法,以帮助确定错误的来源。这种错误在程序中往往是难以发现的,用这种方法时,已执行过的语句标号被记录下来。在跟踪状态中,FOR 循环会自动地按照嵌套的情况记录已执行过的行标号。

在 16 K 扩展 BASIC 中包括了许多扩展 BASIC 版本的特点。这些特点包括,一行可写多个语句;数组可以是一维、二维和三维的;允许自定义 26 个函数。

16 K 扩展 BASIC 的指令、语句较多,内容丰富,功能较强,但是由于篇幅有限,不能一一介绍(例如各种打印格式语句)。下面主要将科学计算中所用到的指令、语句作一介绍,便于阅读程序。

二、扩展 BASIC 语言的基本概念

1. BASIC 语言程序的基本结构

BASIC 语言程序是由一系列语句构成的,一般一个语句占一行,扩展 BASIC 语言允许部分语句可以几个语句在一行中。语句由标号;指令、语句或命令;语句的工作部分;结束符四部分组成。

100 LET $D = B * B - 4 * A * C$
 语句标号 赋值指令 语句工作部分

第一部分是语句标号。它可以是 1~999 之间的任一整数。语句标号不要求连续，也不要求按顺序排列，但在计算机执行时，一般按标号从小到大的顺序执行。

第二部分是指令、语句或命令。它表示语句的性质。各种语句、指令或命令都有固定的字符代号和严格的格式、意义和用法。只有详细地了解各种指令、语句和命令的概念，才能正确地使用。上例中 LET 表示赋值，即是将 B 的平方减去 4 乘 A 乘 C 的值赋给 D。

第三部分是语句的工作部分。它通常由运算对象、运算符、标准函数以及打印对象格式等内容组成。上例中 $D = B * B - 4 * A * C$ 就是工作部分。

第四部分是结束符。它在书写程序时并不书写，但在上机操作时，每一语句输入完后，要打一个回车换行符号 RETURN 作为结束符号。

2. 指令的句法

BASIC 语言的所有字母必须大写。

指令包括命令和语句两类。

命令是立即执行的指令，没有行号，不被 BASIC 所储存。

语句是 BASIC 的一个程序行，它包括一个或多个指令。这些指令是被存储的，只有在机器上给出 RUN 命令或其它可以使程序开始执行的命令后才执行。每一语句行有一个行标号与其它语句相区别。

3. 数在机器内部的表示形式

整数

整数是不带小数点的数。在 16 K BASIC 中，它的范围是

在 +32767 到 -32768 之间,在内存中一个整数占两个字节,如果高字节的最高值为 1,此数为负;若为零,此数为正。正数以数的二进制形式在机内存放,负数以二进制的补码形式在机内存放。

例如 12385, 55 都是整数。

定点数和浮点数

BASIC 语言使用通常意义下的十进制数,但书写形式分为定点数和浮点数两种表示形式。定点数是指一个数的允许使用数码 0~9,小数点和正负号。浮点数表示是在定点数的基础上增加指数部 $E \pm e$,它表示 10 的次方, e 表示 10 的次方数,它是一位或二位整数。浮点数有短浮点数和长浮点数。

例如 -1.2385 55.0 是定点数。

-1.2385 E+21 0.55 E+2 是浮点数。

短浮点数

一个短浮点数在 BASIC 语言中占有 4 个字节,它有六位有效数,数的范围在 $\pm 9.99 E+62$ 至 $\pm 9.99 E-65$ 之间。

例如 -1.2385 E+21 是短浮点数。

长浮点数

一个长浮点数在 BASIC 语言中占有 8 个字节的内存,它可以表示 14 位有效数,数的范围在 $\pm 9.99 E+62$ 至 $\pm 9.99 E-65$ 之间。

例如 -1.238500000000 E+21 是长浮点数。

常数

常数在算术运算中是不会改变其值的,常数有三种类型:整数、实数(浮点数)和 16 进制数。

4. 变量的表示方法

变量的值是变化的,当程序执行时,变量被赋予新的值。扩

展 BASIC 的变量分为数值变量和字符串变量。

(1) 数值变量

如果一个量不写成数的形式,而是以一个名字来表示,这个量就是变量。数值变量就是要被赋予数值的变量。

数值变量的格式

它是用英语 26 个字母中的一个字母或一个字母跟一个数字来表示的。字母都必须大写,例: A5, Y1。

整型变量

用 INTEGER 指令可以使被指定的变量为整型变量。在 RUN 指令之前给出 IMODE 指令,则所有变量取整数值。

短浮点变量

用 SHORT 指令可以使某些变量为短浮点变量。

长浮点变量

用 LONG 指令可以使被指定的某些变量为长浮点变量。

矩阵和向量

矩阵是按规定的格式组成的数值变量的数组。在 BASIC 语言中,如果一组有序的数或变量作为一个整体来处理,这组数或变量便称为数组矩阵的格式。矩阵取名的方法与数值变量相同。矩阵中的元素用矩阵名后面跟以括弧括起来的一个、两个或三个下标来表示。这些下标可以是表达式和变量。

例如

$$a = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix}。$$

表达式

表达式是由数、变量、函数用运算符及圆括号连接起来的有

意义的式子。由数值变量和函数组成的表达式的运算结果是一个数。

(2) 字符串变量

字符串变量被赋予字符值。

字符串变量的格式, 以一个字母跟一个 \$ 或一个字母跟一个数字再跟一个 \$ 来表示。BASIC 语言规定, 只有大写字母才能用做字符串名字。

5. 运算符

运算符是表示对一项或二项进行一次操作的符号或字母的组合。

(1) 算术运算符

算术运算符和平常的代数运算符是很相似的, 其中有

+ - * / 表示加, 减, 乘, 除。

+ - 表示正、负号。

** 或 ^ 表示指数。

它的运算顺序和通常的算术一样, 括号内先算, 然后是指数, 乘除, 最后是加减运算, 自左向右顺序进行。

例如 $A + B * C ** N - (E - F) * M$ 。

运算步骤是:

第一步 算出 $E - F$ 的值;

第二步 算出 O^N 的值;

第三步 算 $(E - F) * M$ 和 $B * C ** N$;

第四步 $A + B * C ** N - (E - F) * M$ 。

(2) 赋值运算符

等号(=)是一个赋值运算符, 用它来给(等号左边的)一个数值变量赋予一个算术表达式, 关系表达式, 布尔表达式或一个函数的值。

赋值符也可用来给一个字符串变量赋予一个字符串文字,字符串变量或字符串函数的值。

BASIC 语言也可以把等号做为关系符使用,作为合法的赋值符,等号只能用于 LET 指令、隐含的 LET 指令或 MAT 指令中第一个等号的地方。

例

100 A=2。

110 PRINT A。

120 A=A*2。

130 IF A=16384 THEN STOP。

140 GOTO 110。

100 句 把 2 这个值赋予 A。

110 句 打印 A。

120 句 把变量 A 的当前值乘以常数 2 的结果赋给变量 A。

130 句 此句中等号是个关系符。当关系表达式 $A=16384$ 的值是“真”时,则程序停止执行,否则执行下一句,140 句。

140 句 控制转移到 110 句,继续执行。

(3) 关系运算符

关系运算符是用来比较两个表达式的。关系运算符的结果是“真”(=1)或“假”(=0)。

关系运算符 =, <, >, <=, >=, <> 或 # 的意义分别表示等于,小于,大于,小于等于,大于等于和不等。

例如 10 IF A<=0 GOTO 50。

第 10 句可读作,如果 A 小于等于零,则程序控制转移到第 50 句。

(4) 布尔运算符

布尔运算是对于一个或两个表达式进行逻辑运算。在一个布尔运算中表达式取一个或两个值(=0)或(=1)。在本语言中,在使用布尔运算符时所有不等于零的值都被认为是真。

布尔运算符 AND、OR、XOR、NOT 的意义分别是逻辑与、逻辑或、逻辑异或、逻辑非或反。

6. 标准函数

对于数学中一些常用的算术和三角函数,它们实现公共的常用的计算,把它们编成一个个独立的子程序,并预先定义,存在 BASIC 程序中,供程序员随时调用,而不必每次都来写计算这些函数的程序。它包括以下函数。

(1) 算术函数

ABS(x)	x 的绝对值。
BINAND(x, y)	二进制逻辑与。
BINOR(x, y)	二进制逻辑或。
BINXOR(x, y)	二进制逻辑异或。
EXP(x)	e 值的 x 次方。
FRA(x)	x 的小数部分。
INT(x)	x 的整数部分。
IRN(x)	值为 0 和 32767 之间的整随机数。
LOG(x)	x 的自然对数。
MAX($x_1, \dots, x_n$)	测定在表达式列中具有最大值的数字表达式。
MIN($x_1, \dots, x_n$)	测定在表达式列中具有最小值的数字表达式。
RANDOMIZE	和 RND 和 IRN 联用产生不同组的随机数。

RND(x) 产生 0 和 1 之间的随机数。

SGN(x) x 的代数符号。

SQR(x) x 的平方根。

(2) 三角函数

ATN(x) x 的反正切。

COS(x) x 的余弦。

SIN(x) x 的正弦。

TAN(x) x 的正切。

(3) 字符串函数

ASC($\$X$) 提供与 $X\$$ 第一个字符等价的 ASCII 码的数值。

CHR\$(X) 取单个的字符, 这个字符的 ASCII 码值等价于 X 。

LEN($X\$$) 给出 $X\$$ 字符串的长度。

POS($X\$, Y\$, N$) 这个函数测定子字符串($Y\$$)在字符串($X\$$)中的位置, 子字符串的第一个字符由第 N 个字符开始定位。

STR\$(X) 把任一数字表达式 X 转换成字符串, 这个字符串是 X 的字符描述。

VAL($X\$$) 把任一 $X\$$ 串表达式转换成 $X\$$ 所描述的数字。

(4) 程序员自定义函数

BASIC 语言允许用户自定义 26 个函数。

格式: DEF FNS($avar-1, avar-2, \dots, avar-n$) = $aexp$ 。

其中 S 是 ASCII 码字符(A-Z)中的任意一个;

$avar1-n$ 是自变量;

$aexp$ 是算术表达式, 变量或常数。