

# CADCS中的表达式输入与求值问题

王治宝 卢生根

(南开大学计算机与系统科学系, 天津)

## 摘 要

在CADCS的设计中, 线性系统的结构是可以参数表达的, 但非线性系统的表达式却没有这样简单的形式, 最简单且理想的方法是用表达式本身。

编译型的高级语言是不能接受一个表达式的, 为了直接输入非线性函数表达式, 我们引入了一个软体运算器, 它具有一套指令系统, 根据非线性函数表达式可由一个编译系统自动编译计算该表达式及其任意偏导值的程序, 这些程序由软体运算器执行。

**关键词:** 非线性函数表达式; 非线性系统; 控制系统计算机辅助设计。

## 一、引 言

在控制系统计算机辅助设计中, 对线性系统可用一些简单的结构参数来表达, 但非线性系统的结构没有那样好的表达形式, 简单的表达方式就是非线性函数表达式, 因此我们曾探讨了这种输入方法<sup>[1-2]</sup>, 为了求非线性函数表达式及其偏导数的值, 我们输入了一个非线性函数表达式和一组偏导表达式, 这就克服了用户编写程序的困难。但是偏导表达式的输入太麻烦了, 能不能通过非线性函数表达式直接求得其偏导数的值呢? 一般地, 可以考虑数值分析的方法, 或者用符号运算的方法自动推导出偏导表达式, 然而前者精度和数据运用上有问题, 后者在化简上也遇到了困难, 并且均要耗费较多的时间。

本文从程序设计方法本身入手给出了比较好的方法, 轻而易举地解决了这个问题。

## 二、表达式及其求导过程

### 1. 算术运算表达式的求值

非线性系统是由一个非线性函数表达式表示的, 这个表达式可以含有如下算术(包括函数)运算:  $X+Y$ ,  $X-Y$ ,  $X*Y$ ,  $X/Y$ ,  $X \uparrow Y$ ,  $\text{SQRT}(x)$ ,  $\text{SIN}(x)$ ,  $\text{COS}(x)$ ,  $\text{LN}(x)$ ,  $\text{EXP}(x)$ ,  $\text{ATAN}(x)$ , ...。为了对该表达式求值, 我们引入了一个虚拟运算器<sup>[3]</sup>, 其指令系统的指令形式为

$$OP \ P_1 \ P_2 \ R$$

其中 $OP$ 为算术运算符,它可以为 $+$ 或 $-$ 或 $*$ 或 $/$ 或 $\uparrow$ 或 $SQRT$ 等等, $P_1$ 和 $P_2$ 为第1和第2运算对象指针,分别指示运算对象 $X$ 和 $Y$ 的位置, $R$ 为运算结果指针,它指示运算结果的位置。

将表达式输入到一个自动编译系统,得到一个由上述指令系统表示的程序,将该程序输入到虚拟运算器,可得到表达式的值。

## 2. 导数运算的定义与导数运算表达式

相对于表达式中定义的算术运算符 $op$ ,我们来定义导数运算符 $op'$ 。如果 $op'$ 满足

$$X \ op' \ Y = \frac{\partial}{\partial pt} (X \ op \ Y)$$

那么称 $op'$ 为相应于 $op$ 的导数运算符, $X \ op' \ Y$ 为相应于 $X \ op \ Y$ 相对于自变量 $pt$ 的偏导数或导数运算。依该定义显然有

$$X + 'Y = X' + Y'$$

$$X * 'Y = X' * Y + X * Y'$$

$$SIN'(x) = X' \ COS(x)$$

$$SQRT'(x) = X' / (2 * SQRT(x))$$

等等,根据这个导数运算定义,用导数运算符将算术运算表达式中的算术运算符代替后,生成的表达式称为导数运算表达式。例如算术运算表达式为

$$A * B * C + SIN(D/E - F)$$

其相应的导数运算表达式为

$$A * 'B * 'C + 'SIN'(D/'E - 'F)$$

## 3. 导数运算表达式的基本性质

根据导数运算与导数运算表达式的定义,很容易得到导数运算表达式的两条基本性质:

**性质 1** 导数运算表达式中,各导数运算符 $op'$ 的优先级与相应算术运算表达式中算术运算符 $op$ 的优先级完全相同。

**性质 2** 当将导数运算表达式化为算术运算表达式后,作为其运算对象的变量 $X$ , $Y$ , $X'$ , $Y'$ ,除了原算术运算表达式中的运算对象 $X$ 和 $Y$ 以外,就是导数 $X'$ 和 $Y'$ 。

例如算术运算表达式为

$$A * B * C + SIN(D/E - F)$$

相应的导数运算表达式及其等价的算术运算表达式为

$$\begin{aligned} & A * 'B * 'C + 'SIN'(D/'E - 'F) \\ & = (A' * B + A * B') * C + (A * B) * C' \\ & + ((D' * E - D * E') / E \uparrow 2 - F') * COS(D/E - F) \end{aligned}$$

## 4. 导数运算表达式的算法

表达式的计算过程可以用一个分析树表示。为了简明起见,可以从一个例子出发来叙述这一算法,例如对算术运算表达式

$$A * B * C + \text{SIN}(D/E - F)$$

可以用图2.1所示的分析树表示。从树叶到树根从左到右结合，恰好表示该算术运算表达式。其计算过程为

- 1) \* A B T<sub>1</sub>
- 2) \* T<sub>1</sub> C T<sub>2</sub>
- 3) / D E T<sub>3</sub>
- 4) - T<sub>3</sub> F T<sub>4</sub>
- 5) SIN T<sub>4</sub> T<sub>5</sub>
- 6) + T<sub>2</sub> T<sub>5</sub> T<sub>6</sub>

其中T<sub>i</sub>指示第*i* (*i*=1, 2, ..., 6)条指令的运算结果, T<sub>6</sub>指示算术运算表达式的值。

相应的导数运算表达式

$$A * ' B * ' C + ' \text{SIN}' (D / ' E - ' F)$$

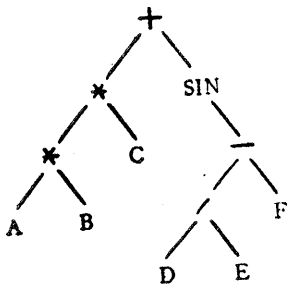


图 2.1 算术运算表达式树

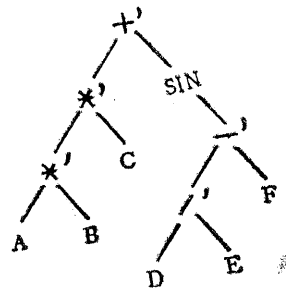


图 2.2 导数运算表达式树

可以用图2.2所示的分析树作类似地表示。从树根到树叶，从右到左依次递归由图2.3

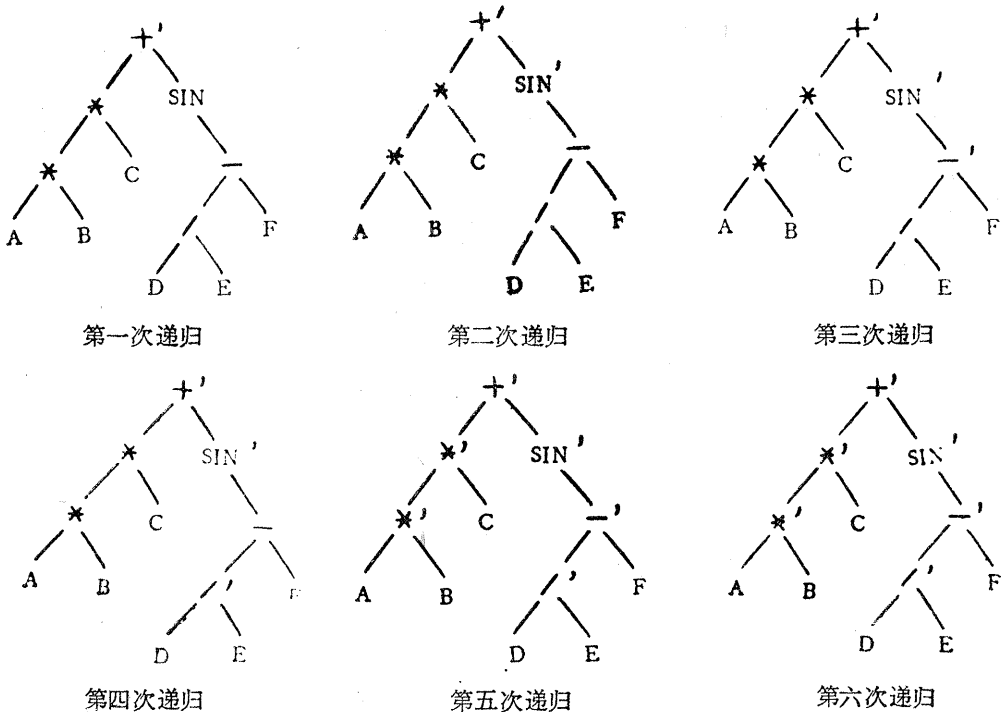


图 2.3 导数运算表达式的递归过程

所示。

其计算过程为

- |                            |                            |
|----------------------------|----------------------------|
| 1) *' A B T <sub>1</sub> ' | 4) -' ③ F T <sub>4</sub> ' |
| 2) *' ① C T <sub>2</sub> ' | 5) SIN' ④ T <sub>5</sub> ' |
| 3) /' D E T <sub>3</sub> ' | 6) +' ② ⑤ T <sub>6</sub> ' |

其中  $T_i'$  指示第  $i$  条指令的运算结果,  $T_i'$  指示导数运算表达式的值。① 表示  $T_1$  和  $T_1'$  两个指针, 具体地说, ①, ②, ③, ④ 和 ⑤ 依次表示图 2.3 中的相应子树, 如图 2.4 所示。

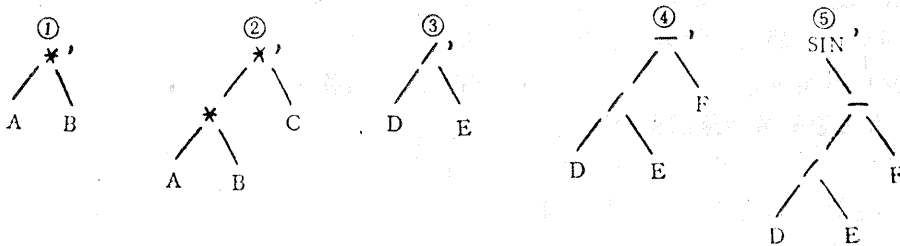


图 2.4 递归过程的子树

依导数运算的定义, 将这个导数运算表达式的计算过程化为一个等价的算术运算表达式的计算过程为

- |   |   |
|---|---|
| 1) * A' B T <sub>1</sub> '                              | 9) - T <sub>7</sub> ' T <sub>8</sub> ' T <sub>9</sub> '     |
| 2) * A B' T <sub>2</sub> '                              | 10) * E E T <sub>10</sub> '                                 |
| 3) + T <sub>1</sub> ' T <sub>2</sub> ' T <sub>3</sub> ' | 11) / T <sub>9</sub> ' T <sub>10</sub> ' T <sub>11</sub> '  |
| 4) * T <sub>3</sub> ' C T <sub>4</sub> '                | 12) - T <sub>11</sub> ' F' T <sub>12</sub> '                |
| 5) * T <sub>1</sub> ' C' T <sub>5</sub> '               | 13) COS T <sub>4</sub> ' T <sub>13</sub> '                  |
| 6) + T <sub>4</sub> ' T <sub>5</sub> ' T <sub>6</sub> ' | 14) * T <sub>12</sub> ' T <sub>13</sub> ' T <sub>14</sub> ' |
| 7) * D' E T <sub>7</sub> '                              | 15) + T <sub>6</sub> ' T <sub>14</sub> ' T <sub>15</sub> '  |
| 8) * D E' T <sub>8</sub> '                              |   |

其中  $T_1$  和  $T_4$  为原算术运算表达式计算过程的第 1 和第 4 条指令的结果指针。  $\bar{T}_i'$  ( $i=1, 2, \dots, 15$ ) 为第  $i$  条指令的结果指针, 其中的  $\bar{T}_3'$ ,  $\bar{T}_6'$ ,  $\bar{T}_{11}'$ ,  $\bar{T}_{12}'$ ,  $\bar{T}_{14}'$  和  $\bar{T}_{15}'$  与导

数运算表达式计算过程的第1, 2, 3, 4, 5和6条指令的结果指令有相同的作用。

### 三、软体运算器与自动编译

#### 1. 四元组软体运算器的应用

在二.1中, 我们曾提到过一个虚拟运算器, 也称为软体运算器, 由于它的指令系统的指令

$$OP \ P_1 \ P_2 \ R$$

由 $OP$ ,  $P_1$ ,  $P_2$ 和 $R$ 四元组成, 故称为四元组软体运算器。它与相应的自动编译系统结合, 对有定义的任意算术运算表达式的求值都是可行的, 因此当将导数运算表达式化为等价的算术运算表达式后, 只要考虑到运算过程中有的运算对象依赖于原算术运算表达式的中间结果, 那么导数运算表达式的值也是可以由四元组软体运算器求得的。为了该求值过程可以实现, 其相应的自动编译系统与四元组软体运算器必须作适当的修改。

#### 2. 六元组软体运算器指令系统

在二.4中, 我们没有直接用导数运算表达式的计算过程求其值, 而是将其化为它的等价形式的算术运算表达式, 借助四元组的软体运算器求值, 但是我们完全可以将导数运算表达式的计算过程表达为

$$\begin{array}{ll} 1) \ *' \ A \ A' \ B \ B' \ T'_1 & 4) \ -' \ T_3 \ T'_3 \ F \ F' \ T'_4 \\ 2) \ *' \ T_1 \ T'_1 \ C \ C' \ T'_2 & 5) \ \text{SIN} \ T_4 \ T'_4 \\ 3) \ /' \ D \ D' \ E \ E' \ T'_3 & 6) \ +' \ T_2 \ T'_2 \ T_5 \ T'_5 \ T'_6 \end{array}$$

一般地, 可以考虑一个导数软体运算器, 其指令系统可以表示为

$$OP' \ P_1 \ P'_1 \ P_2 \ P'_2 \ R$$

其中 $OP'$ 为导数运算符,  $P_1$ 为第1运算对象指针,  $P'_1$ 为第1运算对象导数的指针,  $P_2$ 为第2运算对象指针,  $P'_2$ 为第2运算对象导数指针,  $R$ 为运算结果指针。自然地, 我们称该指令系统为六元组指令系统, 相应地称软体运算器为六元组软体运算器。

#### 3. 六元组软体运算器

六元组软体运算器计算步骤如下:

- 1) 将导数运算程序长度送入 $\text{II}$ ;
- 2) 将指令计数器 $\text{I}$ 置1;
- 3) 取第 $\text{I}$ 条导数运算指令 $OP' \ P_1 \ P'_1 \ P_2 \ P'_2 \ R$ ;
- 4) 依导数运算符 $OP'$ 选择导数运算并将结果送入 $R$ 指定的变量;
- 5) 将 $\text{I} + 1$ 送入 $\text{I}$ ;
- 6) 如果 $\text{I} \leq \text{II}$ , 那么转向3) 否则继续;

7) 结果送入R.

### 4. 六元组自动编译系统与剪枝技术的应用

类似于四元组指令的程序自动编译系统，可以构造六元组程序的自动编译系统。

按上述运算过程，不管运算对象是否是被导变量的函数一律进行求导运算，这样显然会有多余的运算，有时甚至大部分运算是不必要的，例如表达式

$$X - f(Y)$$

对X求偏导，而f(Y)是一个很复杂的函数，显然对f(Y)的求导过程会浪费很多时间。如果能在表达式的自动编译过程消除这种情况，那是很必要的。

为了确定运算对象是否是被导变量的函数，在分析树的每个节点上设置一个指示器，约定0表示与被导变量无关，1表示是被导变量的函数。显然每个树叶的指示器不是就是0，例如表达式

$$A * B * C + SIN(D/E - F)$$

对D求偏导，其指示器的状态如图3.1所示，显然可从这个分析树中，将其中与求偏导无关的两枝剪掉，如图3.2圆圈所示。

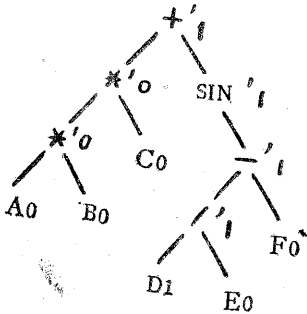


图 3.1 指示器的标志状态

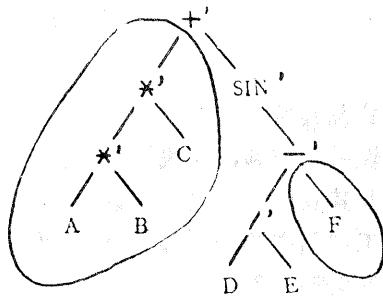


图 3.2 剪枝状态

在自动编译过程中，运用这种剪枝技术可以简化求导计算过程，以提高计算速度。

## 四、应用例子

我们已将表达式的输入方法应用到 CSCAD 软件包系统辨识子包中<sup>[8]</sup>，并在 IBM PC/XT 微型计算机上投入运行。为了说明问题，我们给出一个例子。

静态非线性系统模型的形式为

$$Y = P_1 * U_1 + P_2 * EXP(P_3 * U_2)$$

如果  $P_1 = 100$ ,  $P_2 = 0.1$ ,  $P_3 = 0.5$ , 当给定输入  $U_1$  和  $U_2$  时，可以得到输出  $Y$ 。这样可以得到一系列（例如100组）输入输出数据。

将输入输出数据和非线性函数表达式

$$P_1 * U_1 + P_2 * \text{EXP}(P_3 * U_2)$$

作为CSCAD软件包的输入信息,当要求用麦夸脱方法估计参数 $P_1, P_2, P_3$ 时,迭代4次得到结果为

$$P_1 = 99.9999$$

$$P_2 = 0.1000185$$

$$P_3 = 0.4998982$$

众所周知,用麦夸脱方法估计参数 $P_1, P_2, P_3$ 时必须反复求非线性函数表达式

$$P_1 * U_1 + P_2 * \text{EXP}(P_3 * U_2)$$

及其对 $P_1, P_2, P_3$ 偏导表达式

$$U_1$$

$$\text{EXP}(P_3 * U_2)$$

$$P_2 * \text{EXP}(P_3 * U_2) * U_2$$

的值。

## 五、结 束 语

表达式及其偏导值的计算问题,不但在系统辨识上得到应用,还可以应用到最优控制,非线性系统仿真,非线性系统的分析与综合和自适应预报的CAD中。同时考虑到一般方法及其应用的方便,表达式二次偏导及复合函数求导问题也是很有意义的。关于这方面的工作已有结果<sup>[5]</sup>。表达式及其诸求导问题对于一般CAD及其语言<sup>[4]</sup>的设计都是很有意义的。

## 参 考 文 献

- [1] 王厦生、王治宝、卢桂章, ICS语言和LANMIP软件包, 信息与控制, 1, 6, (1982), 67—74。
- [2] 王治宝、卢桂章、王秀峰, 控制系统CAD软件包的设计——CSCAD的总体设计, 控制理论与应用, 1, 4, (1984), 28—34。
- [3] 王治宝、卢桂章、王秀峰, 控制系统CAD软件包的设计——CSCAD的实现及其应用, 控制理论与应用, 2, 2, (1985), 12—21。
- [4] 王治宝, 控制系统CAD语言的设计——CSCADL语言及其实现, 南开大学学报, 自然科学版, 1, (1986), 63—70。
- [5] Wang Zhibao, Lu shenggen, Man-computer Interface Handling System on Expression in CADCS, 4th IFAC Symposium on Computer Aided Design in Control System CADCS' 88, 23—25, August, (1988), 252—255。

## Input and Evaluating Problem of Expression in CADCS

Wang Zhibao, Lu Shenggan

(Department of Computer and System  
Sciences Nankai University, Tianjin)

### Abstract

In CADCS, the construction of linear system can be expressed with parameters, but nonlinear system can not

An expression can not be accepted by compiled high level language. To be able to input nonlinear system expression, we introduce an arithmetic unit by software, which has the following instruction system,

$$OP \quad P_1 \quad P_2 \quad r$$
$$OP' \quad P_1 \quad P_1' \quad P_2 \quad P_2' \quad r$$

An nonlinear expression is compiled automatically by a compiler system, thus, evaluation of the expression and its partial derivatives can be achieved, which is executed on the software arithmetic unit.

The system has been implemented on IBM PC/XT. These results can be used in CAD for optimal control, nonlinear system control and simulation and adaptive prediction.

**Key words**—Non-linear function expression, Non-linear system, CAD of control system.