

C++言語による微分代数方程式を解くための ジェネリックプログラム

平 山 弘^{†1} 飯 塚 嵩 文^{†1}

Taylor 級数の四則演算および関数は C++言語によって容易にできる。四則演算、関数、条件文等で記述された C++言語で定義された関数は容易に Taylor 展開できる。次のような微分代数方程式

$$f(\mathbf{y}, \mathbf{y}', x) = 0 \text{ 初期条件 } \mathbf{y}(0) = \mathbf{a}_0, \mathbf{y}'(0) = \mathbf{a}_1$$

の解は、Taylor 展開法によって容易に得ることができる。解は任意の次数まで Taylor 展開できる。したがって、任意次数の Runge-Kutta 公式の代わりとして使うことができる。このプログラムの C++言語によるジェネリックプログラムを作成した。このプログラムを使えば、微分代数方程式を数値的および厳密に計算することができる。

The Generic Programs to Solve the Differential Algebraic Equation by C++ language

HIROSHI HIRAYAMA^{†1} and TAKAFUMI IIZUKA^{†1}

The arithmetic operations and functions of Taylor series can be defined by C++ language easily. The functions represented by C++ language which consist of arithmetic operations, pre-defined functions and conditional statements can be expanded in Taylor series. It is shown that the solution of the following differential algebraic equation

$$f(\mathbf{y}, \mathbf{y}', x) = 0 \text{ initial condition } \mathbf{y}(0) = \mathbf{a}_0, \mathbf{y}'(0) = \mathbf{a}_1$$

can be gotten easily by the Taylor series methods. The solutions can be expanded up to arbitrary order, so the calculation formula of arbitrary order can be used instead of Runge-Kutta formula. A generic program for C++ language is developed for the above program. We can solve the differential algebraic equations numerically and exactly by this program easily.

1. はじめに

オーバーロード、オペレータ・オーバーロード機能を持つ C++言語、Fortran 等を使うと有限項で打ち切った Taylor 級数間の四則演算、Taylor 級数の関数演算を定義することができる。この機能を使うと、プログラムの形で与えられた任意の関数を Taylor 級数に展開することができる。これらの計算は、高精度、複素数、高精度複素数など、いろいろな精度、数値形式で行うことができる。

微分代数方程式 (DAE)⁵⁾ は、微分項を含まない式を含む微分方程式で、拘束条件がある微分方程式が代表的な例となる。これらは、一般に

$$f(x, y, y', \dots, y^{(n)}) = 0$$

と書くことができる。通常微分方程式は、Runge-Kutta 法⁶⁾などで解くが、ここでは、微分方程式の解を Taylor 展開して解く方法およびそのプログラムについて述べる。倍精度実数や倍精度複素数などが利用できるようにするため、プログラムは C++言語のテンプレート⁴⁾機能を使ってジェネリックプログラムとして記述した。これを使えば、数値として有理数を使えば、厳密な Taylor 級数が得られ、多倍長高精度数を利用すれば高精度な Taylor 級数が得られる。倍精度や単精度浮動小数点数を使った Taylor 級数展開の計算は Chang and Corliss²⁾によって行われているが、このように簡単にいろいろな精度、種類の数値で解けるものは今のところ存在しないようである。

2. 微分代数方程式の Taylor 級数解

代数微分方程式⁵⁾⁷⁾は、次の形式を持つものを考える。

$$f(x, y, y') = 0 \tag{1}$$

初期条件は、次のように与える。

$$y(x_0) = y_0, \quad y'(x_0) = y_1 \tag{2}$$

ここで、 f 、 y は、一般にベクトル関数で、十分なめらかで必要な回数だけ微分可能とする。 y_0 、 y_1 は定数ベクトルである。また、初期条件は適切であると仮定する。すなわち、式 (1) を満たすものとする。式 (1) の解は、次のような解を持つと仮定することができる。

$$y = y_0 + y_1 x + e x^2 \tag{3}$$

^{†1} 神奈川工科大学

Kanagawa Institute of Technology

ここで e は、 y_0 や y_1 と同じ大きさの定数ベクトルで、これから決定するベクトルである。式 (3) を方程式 (1) に代入し、係数に e を含む x の項が現れたら、それ以上高次の項を省略する。このような計算を行うと e に対する連立一次方程式が得られる。この連立一次方程式を解くことによって e を決定できる。すなわち、この場合、解の 2 次の係数を決定することができる。ここで決定した 2 次の係数を y_2 とすると、方程式 (1) の解は、さらに次のような解を持つと仮定することができる。

$$y = y_0 + y_1x + y_2x^2 + ex^3 \quad (4)$$

2 次係数求める時と同じように、式 (4) を方程式 (1) に代入し e の連立一次方程式を計算する。この連立一次方程式を解くことによって e すなわち解の 3 次の係数を決定することができる。このような計算を必要な回数繰り返すことによって、(1) の解の Taylor 級数を任意次数まで計算することができる。

3. 未定係数を含むべき級数の四則演算

べき級数の四則計算のプログラムは、以下のように簡単に作ることができる。通常のべき級数と同じように平行移動によって、展開位置を原点へ移すことができるので一般性を失うことなしに、原点で展開した式だけを扱うことができる。この級数を次のように定義する。

$$f(x) = f_0 + f_1x + f_2x^2 + f_3x^3 + \cdots + (f_i + p_1e_1 + p_2e_2 + \cdots + p_me_m)x^i \quad (5)$$

$$g(x) = g_0 + g_1x + g_2x^2 + g_3x^3 + \cdots + (g_j + q_1e_1 + q_2e_2 + \cdots + q_me_m)x^j \quad (6)$$

$$h(x) = h_0 + h_1x + h_2x^2 + h_3x^3 + \cdots + (h_k + r_1e_1 + r_2e_2 + \cdots + r_me_m)x^k \quad (7)$$

3.1 加 減 算

$h(x)$ が $f(x)$ と $g(x)$ の和差のとき、 f, g および h の係数は、次のような関係になる

$$h(x) = f(x) \pm g(x) \quad h_n = f_n \pm g_n \quad (n = 0, \cdots, k) \quad (8)$$

ここで、 $k = \min(i, j)$ である。未定係数は、 $i = j$ ならば、

$$r_n = p_n + q_n \quad (n = 1, \cdots, m) \quad (9)$$

となる。もし、 $i > j$ および $i < j$ ならば、それぞれ次のようになる。

$$r_n = q_n, \quad r_n = p_n \quad (n = 1, \cdots, m) \quad (10)$$

3.2 乗 算

$h(x)$ が $f(x)$ と $g(x)$ の積のとき、 f, g および h の係数は、次のような関係になる

$$h(x) = f(x)g(x) \quad h_n = \sum_{m=0}^n f_m g_{n-m} \quad (n = 1, \cdots, m) \quad (11)$$

ここで、 $k = \min(i, j)$ である。未定係数は、 $i = j$ ならば、

$$r_n = g_0p_n + f_0q_n \quad (n = 1, \cdots, m) \quad (12)$$

となる。もし、 $i > j$ および $i < j$ ならば、それぞれ次のようになる。

$$r_n = f_0q_n, \quad r_n = g_0p_n \quad (n = 1, \cdots, m) \quad (13)$$

3.3 逆 数

$h(x)$ が $f(x)$ の逆数のとき、次の式が成り立つ。

$$h(x) = \frac{1}{f(x)} \quad (14)$$

この計算の場合、 $h(x)$ と $f(x)$ は同じ次数 $k = i$ になる。係数には、以下のような関係が成り立つ。

$$h_0 = \frac{1}{f_0} \quad h_n = \sum_{m=0}^{n-1} h_m f_{n-m} \quad (n = 1, \cdots, k) \quad (15)$$

未定係数の係数は、以下のようになる。

$$r_n = \frac{p_n}{f_0} \quad (n = 1, \cdots, m) \quad (16)$$

3.4 除 算

$h(x)$ が $f(x)$ と $g(x)$ の商のとき、すなわち

$$h(x) = \frac{f(x)}{g(x)} \quad (17)$$

である。このよう場合、 $g(x)$ の逆数を計算し、 $f(x)$ を掛けて計算する。 $h(x)$ の次数は $k = \min(i, j)$ となる。もし $i < j$ ならば、逆数の計算次数を i 次にするによって不要な計算をなくすることができる。

4. 未定係数を含むべき級数の関数演算

べき級数の関数計算も通常のべき級数のように定義できる。ここでは指数関数の計算法を示す。

4.1 指数関数

$h(x) = e^{f(x)}$ とおくと $\frac{dh}{dx} = h \frac{df}{dx}$ が成り立つ。この微分方程式から、次のような関係が得られる。この計算の場合、 $h(x)$ と $g(x)$ は同じ次数 $k = i$ になる。

$$h_0 = e^{f_0} \quad h_n = \frac{1}{n} \sum_{m=1}^n m h_{n-m} f_m \quad (n = 1, \dots, k) \quad (18)$$

未定係数部分は、次のようになる。

$$r_n = h_0 p_n \quad (n = 1, \dots, m) \quad (19)$$

4.2 対数関数 $h(x) = \log f(x)$

この関数は、次の微分方程式

$$f(x) \frac{dh(x)}{dx} = \frac{df(x)}{dx} \quad (20)$$

を満たす。この式から、指数関数計算の場合と同様な方法で、次のような関係式が得られる。

$$h_0 = \log f_0 \quad h_n = \frac{1}{n f_0} \left(n f_n - \sum_{k=1}^{n-1} k h_k f_{n-k} \right) \quad (n = 1, \dots, m) \quad (21)$$

未定係数部分は、次のようになる。

$$r_n = h_0 p_n \quad (n = 1, \dots, m) \quad (22)$$

4.3 べき乗 $h(x) = f(x)^\alpha$ (α は定数)

この関数は、次の微分方程式

$$f(x) \frac{dh(x)}{dx} = \alpha h(x) \quad (23)$$

を満たす。この式の両辺に、(5)、(6)、(7) の式を代入して、各次数の x の係数を等しいと置いて、次の関係式が得られる。

$$h_0 = f_0^\alpha \quad h_n = \frac{1}{n f_0} \sum_{k=1}^n ((\alpha + 1)k - n) f_k h_{n-k} \quad (n = 1, \dots, m) \quad (24)$$

未定係数部分は、次のようになる。

$$r_n = h_0 p_n \quad (n = 1, \dots, m) \quad (25)$$

$\alpha = \frac{1}{2}$ とおけば、平方根を計算するためのプログラムになる。

4.4 三角関数

$f(x)$ の三角関数 $g(x) = \sin f(x)$ 、 $h(x) = \cos f(x)$ は、次の微分方程式を満たす

$$\frac{dg(x)}{dx} = h(x) \frac{df(x)}{dx} \quad \frac{dh(x)}{dx} = -g(x) \frac{df(x)}{dx} \quad (26)$$

この式から、係数に対する次のような関係式が得られる。この計算の場合、 $h(x)$ と $g(x)$ は同じ次数 $j = i$ 、 $k = i$ になる。

$$g_0 = \sin f_0 \quad g_n = \frac{1}{n} \sum_{j=1}^n j f_j h_{n-j}$$

$$h_0 = \cos f_0 \quad h_n = -\frac{1}{n} \sum_{j=1}^n j f_j g_{n-j} \quad (n = 1, \dots, k) \quad (27)$$

未定係数部分は、次のようになる。

$$q_n = h_0 p_n \quad r_n = g_0 q_n \quad (n = 1, \dots, m) \quad (28)$$

三角関数は、 $\sin x$ と $\cos x$ を同時に計算すると、計算式が単純で見易い公式となる。これは $\sinh x$ や $\cosh x$ の場合も同様である。

5. ジェネリックプログラム

作成したジェネリックプログラムは、C++言語のテンプレート機能を使用し作成した。約 3000 行のプログラムで、出来るだけ簡単に使えるようにした。前章に示した未定係数を含む計算は意識しないで済むように作成した。数値例の最初の簡単な例 6.1 で使い方を示す。この場合、準備しなければならないプログラムは、方程式と初期条件だけである。

まず、倍精度を計算に使用するので、未定係数を含むべき級数を

```
typedef epower_template<double> epower ;
```

と宣言する。これを宣言すると内部で、以下のべき級数の宣言も自動的に行われる。

```
typedef power_template<double> power ;
```

宣言された epower は未定係数を含むべき級数 (Taylor 級数)、power は通常のべき級数である。方程式は、以下のように与える。

```
void func( const double x, const epower *u, epower *s )
{
    s[0]=diff(u[0])+square(u[0])+2.0*square(u[1]) ;
    s[1] = -u[0]+(1+x)*u[1] ;
}
```

ここで、関数 square は 2 乗関数、関数 diff は微分関数である。初期条件は、初期におけるべき級数 (Taylor 級数) を与える。

```
v[0] = power(1); // 初期状態では v[0]=1
v[1] = power(); // 初期状態では何も入っていないことを示す。
```

関数 v は u の初期条件を表すべき級数である。このように設定してから、微分代数方程式を解く関数 DAE_solver を呼ぶ。結果は、解はべき級数として得られる。この級数にステップ幅を代入すれば次のステップにおける初期値を計算することができる。その初期値を利用して次のステップの初期値を求める。これを繰り返せば、微分代数方程式を解くことができる。

有理数で計算するためには、有理数の計算できるプログラムを作成し、次のように宣言すれば、倍精度と同様に計算できる。

```
typedef epower_template<long_rational>          epower ;
```

ここで使用した long_rational は無限精度有理数計算プログラムである。また複素数で計算するには

```
typedef epower_template<complex<double> >      epower ;
```

とすれば計算できる。この例では倍精度の複素数で計算するための宣言である。

6. 数 値 例

6.1 非線形微分代数方程式

微分代数方程式とは、微分項を含まない式を含む微分方程式である。以下では次の非線形微分代数方程式¹⁾ を解く。

$$\begin{cases} \frac{du_1}{dx} &= -u_1^2 + 2u_2^2 \\ 0 &= -u_1 + (1+x)u_2 \end{cases} \quad u_1(0) = 1 \quad (29)$$

厳密解

$$\begin{cases} u_1(x) &= \frac{1+x}{1+x^2} \\ u_2(x) &= \frac{1}{1+x^2} \end{cases} \quad (30)$$

$u_1(x)$ に関しては x^2 以上の高次項を省略し、 $u_2(x)$ に関して x の 1 次以上の項を省略すると、初期条件から、解は次の形で表されることがわかる。 e_1 、 e_2 を未知数とすると

$$u_1(x) = 1 + e_1 x \quad u_2(x) = e_2 \quad (31)$$

と書ける。式 (31) を式 (29) に代入して、 x の 1 次以上の項を省略すると

$$\begin{cases} e_1 &= -1 + 2e_2^2 \\ 0 &= -1 + e_2 \end{cases} \quad (32)$$

これを解いて、 $e_1 = 1$ 、 $e_2 = 1$ となる。式 (32) のように、一番最初の計算は一般に非線形方程式となる。非線形方程式を解く必要がないようにするには、方程式 (29) に現れる未知関数とその微分係数を初期値として与える。そのように与えると、それ以降の方程式は連立一次方程式だけを解くことによって計算を進めることができる。方程式 (29) では、 $u_1(0)$ 、 $u_2(0)$ 、 $u_1'(0)$ の 3 つの初期値を与えれば、それ以降すべて連立一次方程式を解くことによって、Taylor 級数が計算できる。

上の計算結果から、初期条件は以下ようになる。

$$u_1(0) = 1 \quad u_2(0) = 1 \quad u_1'(0) = 1 \quad (33)$$

この初期条件を使って、方程式 (29) を解く。このような操作を繰り返し、9 次まで $u_1(x)$ の Taylor 展開式を求める。この結果から $u_1(x)$ の微分 $u_1'(x)$ と $(u_2(x))$ は 8 次式となり以下のようになる。

$$u_1(x) = 1 + x - x^2 - x^3 + x^4 + x^5 - x^6 - x^7 + x^8 + x^9 \quad (34)$$

$$u_1'(x) = 1 - 2x - 3x^2 + 4x^3 + 5x^4 - 6x^5 - 7x^6 + 8x^7 + 9x^8 \quad (35)$$

$$u_2(x) = 1 - x^2 + x^4 - x^6 + x^8 \quad (36)$$

これらの式が十分速く収束すると仮定する。その収束半径を h とする。この収束半径内では、これらの級数が十分速く収束し、十分な精度で関数値、微分係数が計算できる。このような場合、最大次数の項が許容誤差程度の大きになると仮定できる。許容誤差を ϵ としたとき、式 (34)、(35)、(36) から、それぞれ

$$h^9 \leq \epsilon \quad 9h^8 \leq \epsilon \quad h^8 \leq \epsilon \quad (37)$$

となる。 $\epsilon = 1.0 \times 10^{-16}$ としたとき、(37) のすべての条件を満たす最大の h は $h = 0.00759835$ である。 $x = h$ は次の展開位置になる。この $h = 0.00759835$ で Taylor 展開すると

$$u_1(x) = 1.00754 + 0.984632x - 1.02244x^2 - 0.969038x^3 + 1.03711x^4 \\ + 0.953222x^5 - 1.05154x^6 - 0.937188x^7 + 1.06572x^8 + 0.92094x^9 \quad (38)$$

$$u_1'(x) = 0.984632 - 2.04489x - 2.90711x^2 + 4.14844x^3 + 4.76611x^4 \\ - 6.30921x^5 - 6.56032x^6 + 8.52573x^7 + 8.28846x^8 \quad (39)$$

$$u_2(x) = 0.999942 - 0.015195x - 0.999654x^2 + 0.0303847x^3 + 0.999134x^4 \\ - 0.0455656x^5 - 0.998384x^6 + 0.0607342x^7 + 0.997404x^8 \quad (40)$$

となる。ここでの x は h だけ並行移動しているので $x - 0.00759835$ を意味する。この方法で $x = 0$ から $x = 5$ まで計算すると下のような表になる。9 次の Taylor 展開を利用すると、254 回 Taylor 展開しなければならないことがわかる。もし 16 次式を利用すると同程度の精度で 32 回の Taylor 展開を行うことによって、同様な結果が得られる。

N	x	$u_1(x)$	$u_2(x)$	$u_1(x)$ の誤差	$u_2(x)$ の誤差
0	0.00000	1.0000000000000000	1.0000000000000000	0.000	0.000
1	0.00760	1.007540186496727	0.999942268306222	8.023e-18	2.765e-17
2	0.01528	1.015038498238999	0.999766718317692	8.305e-17	3.746e-17
3	0.02305	1.022503542278010	0.999469134227284	3.263e-17	1.480e-17
4	0.03093	1.029946728502382	0.999044114746433	7.785e-17	3.377e-17
		...			
253	4.98986	0.231280447914390	0.038612013275414	1.580e-15	2.087e-16
254	5.00000	0.230769230769229	0.038461538461538	1.597e-15	2.060e-16

6.2 高階微分代数方程式（3重振り子）

厳密に計算できる例として 3 重振り子の解を Taylor 展開する。3 重振り子の方程式は、以下のようになる。第 i 番目の質点の座標 (x_i, y_i) ($i = 1, 2, 3$)、質量はすべて 1 すなわち $m_1 = m_2 = m_3 = 1$ 、質点間の距離は 1 すなわち $l = 1$ とする。

第 i 質点と第 j 質点の張力は $t_{i,j}$ とする。厳密に計算するため $g = \frac{98}{10}$ とする。初期条件として、質点の距離を 3, 4, 5 になる三角形になるように一平面上に静止した状態で配置した。このとき方程式は次のようになる。

$$\begin{aligned} x_1^2 + y_1^2 - 1 &= 0 \\ (x_1 - x_2)^2 + (y_1 - y_2)^2 - 1 &= 0 \\ (x_2 - x_3)^2 + (y_2 - y_3)^2 - 1 &= 0 \\ m_1 \frac{d^2 x_1}{dt^2} + t_{01} \frac{x_1}{l} + t_{12} \frac{x_1 - x_2}{l} &= 0 \end{aligned}$$

$$\begin{aligned} m_1 \frac{d^2 y_1}{dt^2} + m_1 g + t_{01} \frac{y_1}{l} + t_{12} \frac{y_1 - y_2}{l} &= 0 \\ m_2 \frac{d^2 x_2}{dt^2} + t_{12} \frac{x_2 - x_1}{l} + t_{23} \frac{x_2 - x_3}{l} &= 0 \\ m_2 \frac{d^2 y_2}{dt^2} + m_2 g + t_{12} \frac{y_2 - y_1}{l} + t_{23} \frac{y_2 - y_3}{l} &= 0 \\ m_3 \frac{d^2 x_3}{dt^2} + t_{23} \frac{x_3 - x_2}{l} &= 0 \\ m_3 \frac{d^2 y_3}{dt^2} + m_3 g + t_{23} \frac{y_3 - y_2}{l} &= 0 \end{aligned}$$

初期条件は静止した状態から開始するようにした。したがって、1 階微分である速度は 0 となる。

$$\begin{aligned} x_1(0) &= \frac{4}{5} & y_1(0) &= -\frac{3}{5} & \frac{dx_1}{dt} &= 0 & \frac{dy_1}{dt} &= 0 \\ x_2(0) &= \frac{8}{5} & y_2(0) &= -\frac{6}{5} & \frac{dx_2}{dt} &= 0 & \frac{dy_2}{dt} &= 0 \\ x_3(0) &= \frac{12}{5} & y_3(0) &= -\frac{9}{5} & \frac{dx_3}{dt} &= 0 & \frac{dy_3}{dt} &= 0 \end{aligned}$$

この代数微分方程式の解は有理数で計算すると、厳密に計算でき、次のような解が得られる。この計算は前例題のように何回も繰り返すことはできるが、次のステップの初期条件の計算は、無限級数を有限項で打ち切って計算することになるため、厳密な値にはならない。このため、厳密な計算はそこで終わりとなる。近似で計算するならば、時間の掛かる有理数計算は無意味で無駄な計算となる。有理数の厳密な計算は 120 桁の多倍長計算よりも時間のかかる計算となった。得られた解は、方程式に代入することによって、確かめることができる。完全なゼロにはならないが t の高次の項だけが残るようになる。

3 重振り子の解の 14 次までの Taylor 展開式は次のようになる。 t_{01} 、 t_{12} 、 t_{23} は 12 次までの Taylor 展開式になる。

$$\begin{aligned} x_1 &= \frac{4}{5} - \frac{294}{125}t^2 - \frac{16807}{6250}t^4 + \frac{56824467}{1562500}t^6 - \frac{18355949927}{312500000}t^8 - \frac{60159520498063}{78125000000}t^{10} \\ &\quad + \frac{1313976234092138393}{25781250000000}t^{12} + \frac{2510582471240281958639}{167578125000000000}t^{14} \\ y_1 &= -\frac{3}{5} - \frac{392}{125}t^2 + \frac{28812}{3125}t^4 + \frac{4235364}{390625}t^6 - \frac{7839305817}{39062500}t^8 + \frac{1875514592539}{7324218750}t^{10} \\ &\quad + \frac{220921419737045503}{32226562500000}t^{12} - \frac{124407041676788879351}{3491210937500000}t^{14} \end{aligned}$$

$$\begin{aligned}
 x_2 &= \frac{8}{5} - \frac{294}{125}t^2 - \frac{19208}{3125}t^4 + \frac{15176721}{781250}t^6 + \frac{1176842947}{19531250}t^8 - \frac{15290829118047}{19531250000}t^{10} \\
 &\quad + \frac{4232291911311373}{32226562500000}t^{12} + \frac{612014950318767821}{16757812500000}t^{14} \\
 y_2 &= -\frac{6}{5} - \frac{392}{125}t^2 + \frac{14406}{3125}t^4 - \frac{4588311}{390625}t^6 - \frac{560832783}{39062500}t^8 + \frac{9963184507479}{19531250000}t^{10} \\
 &\quad - \frac{2453362224397021}{2014160156250}t^{12} - \frac{10706809080660950229}{558593750000000}t^{14} \\
 x_3 &= \frac{12}{5} - \frac{294}{125}t^2 - \frac{19208}{3125}t^4 + \frac{1411788}{78125}t^6 + \frac{7124470493}{156250000}t^8 - \frac{29045275356783}{39062500000}t^{10} \\
 &\quad + \frac{17227168581356279}{25781250000000}t^{12} + \frac{2715952532852056887949}{83789062500000000}t^{14} \\
 y_3 &= -\frac{9}{5} - \frac{392}{125}t^2 + \frac{14406}{3125}t^4 - \frac{1058841}{78125}t^6 - \frac{331064286}{9765625}t^8 + \frac{10987439760353}{19531250000}t^{10} \\
 &\quad - \frac{1069858339547809}{2148437500000}t^{12} - \frac{342857460770508015703}{13964843750000000}t^{14} \\
 t_{01} &= \frac{441}{25} + \frac{172872}{625}t^2 - \frac{29647548}{15625}t^4 - \frac{17086870164}{1953125}t^6 + \frac{41987887729893}{195312500}t^8 \\
 &\quad - \frac{3011411287113453}{12207031250}t^{10} - \frac{1003401402873151529457}{53710937500000}t^{12} \\
 t_{12} &= \frac{294}{25} + \frac{115248}{625}t^2 - \frac{4235364}{3125}t^4 - \frac{12428910956}{1953125}t^6 + \frac{1640858367834}{9765625}t^8 \\
 &\quad - \frac{5257943156865931}{5257943156865931}t^{10} - \frac{119349791054624545051}{119349791054624545051}t^{12} \\
 t_{23} &= \frac{147}{25} + \frac{57624}{625}t^2 - \frac{2117682}{3125}t^4 - \frac{6214455478}{1953125}t^6 + \frac{6555806639613}{78125000}t^8 \\
 &\quad - \frac{671397340932164}{6103515625}t^{10} - \frac{2384929385164768789151}{322265625000000}t^{12}
 \end{aligned}$$

7. ま と め

ここで作成したプログラムを利用すると、代数微分方程式を倍精度、複素数などいろいろな型・精度で解くことができ、大変便利である。

多くの場合、数値の型が変わってもアルゴリズムは大きくは変わらないが、浮動小数点数では誤差が入るため、誤差を少なくする計算処理を行う。有理数では厳密な計算であるため、そのような計算を必要としない。複素数では収束判定に絶対値を計算し、その値によって判定するが、実数計算では絶対値計算が不要な場合がある。若干プログラムが異なる場合

が多い。

このような違いがあるプログラムを1個のプログラムにするため、次期 C++ の標準ライブラリ候補の Boost ライブラリ³⁾の使用を試みている。

参 考 文 献

- 1) アヒニアズナルメット、森 正武, "二重指数変換による微分代数方程式の数値解法", 第34回数値解析シンポジウム講演予稿集, (2005), 1-4
- 2) Chang Y. F. and Corliss G., "ATOMFT: Solving ODEs and DAE Using Taylor Series", Computers Math. Applic., 28(1994), 209-233
- 3) ビョルン・カールソン (村上雅章訳), "Boost", ピアソン・エデュケーション, (2008)
- 4) David Abrahams, Aleksey Gurtovoy, "C++ Template Metaprogramming", Addison Wesley, (2005)
- 5) Hairer E., Wanner G., "Solving Ordinary Differential Equations II", Springer-Verlag, (1991)
- 6) 三井斌友, "数値解析入門", 朝倉書店, (1985)
- 7) 渡辺, 高木, "常微分方程式初期値問題の数値計算プログラム HIDMAS", 日本応用数理学会論文誌, 1(1991), 135-163