4 次方程式の解とフェラーリの方法の計算プログラム

2016.7.19　　鈴木　実

1　フェラーリの方法による 4 次方程式の解

4 次方程式を

$$x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 = 0 \tag{1}$$

とする．x^3 の係数を 0 とするため，

$$x = y - b_3 \tag{2}$$

$$b_3 = \frac{a_3}{4} \tag{3}$$

とおくと，

$$y^4 + py^2 + qy + r = 0 \tag{4}$$

となる．ただし，

$$p = a_2 - 6b_3^2 \tag{5}$$

$$q = a_1 - 2a_2b_3 + 8b_3^3 \tag{6}$$

$$r = a_0 - a_1b_3 + a_2b_3^2 - 3b_3^4 \tag{7}$$

である．ここで新しい変数 u を導入して，式 (4) を次の連立方程式に書き換えることができる．

$$\left(y^2 + \frac{p+u}{2}\right)^2 - u\left(y - \frac{q}{2u}\right)^2 = 0 \tag{8}$$

$$u(p+u)^2 - 4ru = q^2 \tag{9}$$

式 (9) は 3 次方程式であるからカルダノの方法で解くことができる（たとえば，[1]）．得られた u を用いると式 (8) は次の 2 つの 2 次方程式に帰結できる．すなわち，

$$\left(y^2 + \frac{p+u}{2}\right) + \sqrt{u}\left(y - \frac{q}{2u}\right) = 0 \tag{10}$$

$$\left(y^2 + \frac{p+u}{2}\right) - \sqrt{u}\left(y - \frac{q}{2u}\right) = 0 \tag{11}$$

したがって，4 次方程式の解は，

$$x = -\frac{a_3}{4} + \frac{\sqrt{u}}{2} \pm \sqrt{\left(\frac{\sqrt{u}}{2}\right)^2 - \frac{p+u}{2} + \frac{q}{2\sqrt{u}}} \tag{12}$$

$$x = -\frac{a_3}{4} - \frac{\sqrt{u}}{2} \pm \sqrt{\left(\frac{\sqrt{u}}{2}\right)^2 - \frac{p+u}{2} - \frac{q}{2\sqrt{u}}} \tag{13}$$

となる．

参考文献

[1] 「3 次方程式の解とカルダノの方法の計算プログラム」(2016/7/19 のエントリー)

プログラムソース

1. フェルミ・ディラック積分の数値計算プログラム

```
/* solve_quartic_eq.c  -o  solve_quartic_eq */
/* 2016.7.16 by M. Suzuki */

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

double cube_root(double x)
{
    double z;
    z=pow(fabs(x), 1.0/3.0);
    if(x<0) return -z;
    else return z;
}

/*                                                                              */
/*  Solve the cubic equation by Cardano's method.                               */
/*                                                                              */
int cubic_equation(double a2, double a1, double a0, double *solution)
{
    int i;
    double u, v, x, y;
    double p, q, theta;
    double A, B, C, D, F, R, Q, S, T;

    p=3*a1-a2*a2;
    q=27*a0-9*a1*a2+2*a2*a2*a2;
    D=q*q+4*p*p*p;
    D/=(54*54);
    D*=-1;
    p/=3;
    q/=27;
    A=-a2/3;
    B=-q/2;
    C=sqrt(fabs(D));
    for(i=0;i<5;i++) solution[i]=0;

    if(D==0.0)
    {
        if(q==0)
        {
            for(i=0;i<3;i++) solution[i]=A;
        }
        else
        {
            F=cube_root(B);
            solution[0]=A+2*F;
            solution[1]=solution[2]=A-F;
        }
    }
    else if(D>0)
```

```
    {
        x=B*B+D;
        y=pow(x, 1.0/6.0);
        theta=atan2(C, B);
        theta/=3;
        R=y*cos(theta);
        Q=y*sin(theta)*sqrt(3);
        solution[0]=A+2*R;
        solution[1]=A-R-Q;
        solution[2]=A-R+Q;
    }
    else
    {
        S=cube_root(B+C);
        T=cube_root(B-C);
        solution[0]=A+S+T;
        solution[1]=solution[2]=A-(S+T)/2;
        solution[3]=S-T;
        y=sqrt(3)/2;
        solution[3]*=y;
        solution[4]*=-1;
    }
    return 0;
}


/*                                                                              */
/*  Solve the quartic equation by Ferrari's method.                             */
/*                                                                              */
double quartic_equation(double a3, double a2, double a1, double a0, double *solution4)
{
    int i;
    double b3, p, q, r;
    double u, x, y, z;
    double Dp, Dm, rtDp, rtDm, R, S, T;
    double solution3[5];


    b3=a3/4;
    p=a2-6*b3*b3;
    q=a1-2*a2*b3+8*pow(b3, 3);
    r=a0-a1*b3+a2*b3*b3-3*pow(b3,4);

    x=2*p;
    y=p*p-4*r;
    z=-q*q;

    i=cubic_equation(x, y, z, solution3);
    u=solution3[0];

    R=-sqrt(u)/2;
    S=(p+u)/2;
    T=-q/(4*R);

    Dp=R*R-S+T;
    Dm=R*R-S-T;
    rtDp=sqrt(fabs(Dp));
```

```
    rtDm=sqrt(fabs(Dm));

    if(Dp>=0)
    {
        solution4[0]=-b3+R-rtDp;
        solution4[1]=-b3+R+rtDp;
        solution4[4]=0;
        solution4[5]=0;
    }
    else
    {
        solution4[0]=-b3+R;
        solution4[1]=-b3+R;
        solution4[4]=-rtDp;
        solution4[5]=rtDp;
    }

    if(Dm>=0)
    {
        solution4[2]=-b3-R-rtDm;
        solution4[3]=-b3-R+rtDm;
        solution4[6]=0;
        solution4[7]=0;
    }
    else
    {
        solution4[2]=-b3-R;
        solution4[3]=-b3-R;
        solution4[6]=-rtDm;
        solution4[7]=+rtDm;
    }
    return 0;
}

int main(int argc, char *argv[])
{
    int i;
    double a3, a2, a1, a0, x, y;
    double solution4[8];

    a3=atof(argv[1]);
    a2=atof(argv[2]);
    a1=atof(argv[3]);
    a0=atof(argv[4]);

    printf("a3 %lf\t a2 %lf\t a1 %lf\t a0 %lf\n", a3, a2, a1, a0);

    i=quartic_equation(a3, a2, a1, a0, solution4);

    for(i=0;i<4;i++) printf("%lf\t%lf\n", solution4[i], solution4[i+4]);
}
```