Solving Cubic Equations in Trigonometric/Hyperbolic Method

1. INTRODUCTION

Problems of cubic equations arise in a wide variety of fields, and thus has been studied for long. While Cardano's method is probably the most famous one, it involves imaginary numbers in some cases. It has become prominent that calculation with such numbers is unfriendly to computers, considering there is little need to manually solve equations nowadays.

The trigonometric/hyperbolic method is a good counter to that. However, it is yet to receive the reputation. Actually, searching GitHub for cubic equation solvers yields no result in this method up to January 2019. That composes the motivation of this article.

2. PREREQUISITES

The triple-angle formulae is needed.

Formula 1 – sine:

$$\sin 3\theta = -4\sin^3\theta + 3\sin\theta$$

Formula 2 – cosine:

$$\cos 3\theta = 4\cos^3\theta - 3\cos\theta$$

Formula 3 – hyperbolic sine:

$$\sinh 3\theta = 4\sinh^3\theta + 3\sinh\theta$$

Formula 4 – hyperbolic cosine:

$$\cosh 3\theta = 4\cosh^3\theta - 3\cosh\theta$$

These can be proved using sum/difference identities.

3. SOLUTION

The general idea of this method is to eliminate the cubic term through any of the triple-angle formula. To implement the formula, we must turn the equation into the desired form. The steps are shown below.

Reduction to Depressed Cubic

Typically, a cubic equation has the form

$$ax^3 + bx^2 + cx + d = 0$$

where the square term can always be eliminated by setting

$$t = x + \frac{b}{3a}$$

to obtain

$$t^3 + pt + q = 0 \quad \text{(Eq. 1)}$$

called the depressed form, where

$$p = \frac{3ac - b^2}{3a^2}, q = \frac{2b^3 - 9abc + 27a^2 d}{27a^3}$$

The following steps shall always be concerning this depressed form.

(i) Trigonometric Substitution

By setting

$$t = u \cos\theta$$

from Equation 1 we obtain

$$4u^3 \cos^3\theta + 4pu\cos\theta = -4q \quad \text{(Eq. 2a)}$$

Compare this equation with Formula 2. The goal is to find u to make it coincide with Formula 2. If we set

$$u = \sqrt{-\frac{4}{3}p}$$

equation 2a will be turned into

$$4\cos^3\theta - 3\cos\theta = -\frac{4q}{u^3}$$

Now implementing Formula 2,

$$\cos 3\theta = -\frac{4q}{u^3}$$

Thus,

$$\theta = \frac{1}{3}[\arccos(-\frac{4q}{u^3}) + 2k\pi]$$

Since $t = u\cos\theta$ and $x = t - \frac{b}{3a}$,

$$x_k = u\cos[\frac{1}{3}\arccos(-\frac{4q}{u^3}) + \frac{2k\pi}{3}] - \frac{b}{3a}$$

where $k = 0, 1, 2$.

An alternative form is by setting $t = u\sin\theta$ and using Formula 1. The result is

$$x_k = u\sin[\frac{1}{3}\arcsin(\frac{4q}{u^3}) + \frac{2k\pi}{3}] - \frac{b}{3a}$$

where $k = 0, 1, 2$.

This part works without imaginary numbers if $-1 \le -\frac{4q}{u^3} \le 1$. That is equivalent to the discriminant $4p^3 + 27q^2 \le 0$, which implies $p < 0$, and means there are three real roots.

(ii) Hyperbolic Substitution I

If $4p^3 + 27q^2 > 0$, the argument of the inverse cosine function will go beyond [−1, 1], which introduces imaginary numbers. In this case, we have our tactics adjusted as setting

$$t = u \cosh \theta$$

and

$$u = \sqrt{-\frac{4}{3} p \operatorname{sgn}(-q)}$$

where *sgn* is the signum function placed here to ensure the argument of the inverse hyperbolic cosine function is positive.

Now we go through the exactly same procedure as above. Implementing Formula 4, the result has the form

$$x = u \cosh\left[\frac{1}{3} \operatorname{arcosh}\left(-\frac{4q}{u^3}\right)\right] - \frac{b}{3a}$$

This part works without imaginary numbers if $\left|-\frac{4q}{u^3}\right| \geq 1$. That is equivalent to $p < 0$ and the discriminant $4p^3 + 27q^2 > 0$, meaning there is one real root.

(iii) Hyperbolic Substitution II

If $p > 0$, the argument of the square root function will go negative, which introduces imaginary numbers. In this case, we have our tactics adjusted as setting

$$t = u \sinh \theta$$

and

$$u = -\sqrt{\frac{4}{3} p}$$

so as to ensure the argument of the square root is positive. Implementing Formula 3, the result has the form

$$x = u \sinh\left[\frac{1}{3} \operatorname{arsinh}\left(-\frac{4q}{u^3}\right)\right] - \frac{b}{3a}$$

This part works without imaginary numbers if $p > 0$, which implies $4p^3 + 27q^2 > 0$, meaning there is one real root.

(iv) Special Cases

If $p = 0$, all the substitutions fail due to a division by zero. Nonetheless, the equation in this case can be solved directly and substitution is not needed. See the summary below.

4. SUMMARY

The following two cases are of THREE real solutions.

(i) If $4p^3 + 27q^2 \le 0$ and $p = 0$,

$$x_k = -\frac{b}{3a} \quad \text{where} \quad k = 0, 1, 2 \ .$$

(ii) If $4p^3 + 27q^2 \le 0$ and $p \ne 0$,

$$x_k = u\cos\left[\frac{1}{3}\operatorname{arccos}\left(-\frac{4q}{u^3}\right) + \frac{2k\pi}{3}\right] - \frac{b}{3a} \quad \text{where} \quad u = \sqrt{-\frac{4}{3}p} \quad \text{and}$$

$k = 0, 1, 2$.

The following three cases are of ONE real solution.

(iii) If $4p^3 + 27q^2 > 0$ and $p = 0$,

$$x = -\sqrt[3]{q} - \frac{b}{3a} \ .$$

(iv) If $4p^3 + 27q^2 > 0$ and $p < 0$,

$$x = u\cosh\left[\frac{1}{3}\operatorname{arcosh}\left(-\frac{4q}{u^3}\right)\right] - \frac{b}{3a} \quad \text{where} \quad u = \sqrt{-\frac{4}{3}p}\,\operatorname{sgn}(-q) \ .$$

(v) If $4p^3 + 27q^2 > 0$ and $p > 0$,

$$x = u\sinh\left[\frac{1}{3}\operatorname{arsinh}\left(-\frac{4q}{u^3}\right)\right] - \frac{b}{3a} \quad \text{where} \quad u = -\sqrt{\frac{4}{3}p} \ .$$

APPENDIX 1. CREDITS

Key Authors:
Flora Canou
Contributors:
Alexander Zheng

Reference Materials
This article uses material from the Wikipedia article *Cubic Function*, which is released under the Creative Commons Attribution-ShareAlike 3.0 Unported License.

APPENDIX 2. EXAMPLE PROGRAM IN C

Note: these codes are licensed under the Mozilla Public License 2.0.
It is recommended to copy them to a text editor to read these codes.

```
/* Copyright 2019 Flora Canou | V. C0-1.0.0 | Cubic Equation Solver
 * This Source Code Form is licensed under the Mozilla Public License, v. 2.0.
 * If a copy of the MPL was not distributed with this file, you can obtain one at https://mozilla.org/MPL/2.0/.
 * The program solves cubic equation in trigonometric/hyperbolic method.
 */

 #include <math.h>
 #include <float.h>
 #define pi  3.14159265359

int solvecubic (double a, double b, double c, double d, double *x)
{
    int n = 0; //number of real solutions
    if (fabs(a) < DBL_EPSILON)
    {
        if (fabs(b) < DBL_EPSILON)
        {
            if (fabs(c) < DBL_EPSILON)
            {
                if (fabs(d) < DBL_EPSILON) // zero
                    n = -1; //infinitely many solutions
                else // non-zero constant
                    n = 0;
            }
```

```
        else // linear
        {
            n = 1;
            x[0] = -d/c;
        }
    }
    else //quadratic
    {
        double discriminant = c*c - 4*b*d;
        if (discriminant >= 0)
        {
            n = 2;
            x[0] = (-c + sqrt (discriminant)) / (2*b);
            x[1] = (-c - sqrt (discriminant)) / (2*b);
        }
        else
            n = 0;
    }
}
else //cubic
{
    double p, q, discriminant, u;
    p = (3*a*c - b*b) / (3*a*a);
    q = (2*b*b*b - 9*a*b*c + 27*a*a*d) / (27*a*a*a);
    discriminant = 4*p*p*p + 27*q*q;
    if (discriminant <= 0) // three real solutions
    {
        n = 3;
        if (fabs(p) < DBL_EPSILON)
        {
            x[0] = -b / (3*a);
            x[1] = x[0];
            x[2] = x[0];
        }
        else
        {
            u = sqrt (-4*p/3);
            x[0] = u * cos (acos (-4*q/(u*u*u)) / 3) - b /
(3*a);
            x[1] = u * cos ((acos (-4*q/(u*u*u)) - 2*pi) / 3) -
b / (3*a);
            x[2] = u * cos ((acos (-4*q/(u*u*u)) + 2*pi) / 3) -
b / (3*a);
        }
    }
    else // one real solution
```

```
        {
            n = 1;
            if (fabs(p) < DBL_EPSILON)
                x[0] = -cbrt(q) - b / (3*a);
            else if (p < 0)
            {
                u = copysign (sqrt (-4*p/3), -q);
                x[0] = u * cosh (acosh (-4*q/(u*u*u)) / 3) - b /
(3*a);
            }
            else
            {
                u = -sqrt (4*p/3);
                x[0] = u * sinh (asinh (-4*q/(u*u*u)) / 3) - b /
(3*a);
            }
        }
    }
    return n;
}
```