FDU 高等线性代数 A. 求根公式

A.1 三次方程

三次方程的求根公式针对的是形如 $x^3+ux+v=0$ 的方程，即缺少二次项的三次方程.
这是因为对于一般的三次方程 $\tilde{x}^3+a\tilde{x}^2+b\tilde{x}+c=0$，可以对 $\tilde{x}$ 做平移消去方程的二次项.
具体来说，可令 $x=\tilde{x}+\frac{1}{3}a$，则 $\tilde{x}=x-\frac{1}{3}a$，
代入 $\tilde{x}^3+a\tilde{x}^2+b\tilde{x}+c=0$ 化简可得 $x^3+(b-\frac{a^2}{3})x+(c-\frac{ab}{3}+\frac{2a^3}{27})=0$.
因此我们只需研究形如 $x^3+ux+v=0$ 的三次方程的求根公式 (其中 $\begin{cases} u=b-\frac{a^2}{3} \\ v=c-\frac{ab}{3}+\frac{2a^3}{27} \end{cases}$)

(1) 因式分解

首先注意到 $C=\begin{bmatrix} 0 & 1 & \\ & 0 & 1 \\ 1 & & 0 \end{bmatrix}$ 是一个 Frobenius 酉型
其特征多项式为 $\det(\lambda I-C)=\lambda^3-1$，其特征值为 $1,\omega,\omega^2$ (其中 $\omega:=\exp\left(\frac{2\pi i}{3}\right)$)
其次注意到 $C^2=\begin{bmatrix} 0 & & 1 \\ 1 & 0 & \\ & 1 & 0 \end{bmatrix}$ 也是一个 Frobenius 酉型
其特征多项式为 $\det(\lambda I-C)=\lambda^3-1$，其特征值为 $1,\omega^2,\omega^4$ (注意 $\omega^4=\omega$)
因此 $xI+yC+zC^2=\begin{bmatrix} x & y & z \\ z & x & y \\ y & z & x \end{bmatrix}$ 的特征值为 $\begin{cases} \lambda_1=x+y+z \\ \lambda_2=x+y\omega+z\omega^2 \\ \lambda_3=x+y\omega^2+z\omega^4 \end{cases}$
于是我们有:

$$\begin{aligned} \det\left(xI+yC+zC^2\right) &= \det\left(\begin{bmatrix} x & y & z \\ z & x & y \\ y & z & x \end{bmatrix}\right) \\ &= x^3+y^3+z^3-3xyz \\ &= \lambda_1\cdot\lambda_2\cdot\lambda_3 \\ &= (x+y+z)(x+y\omega+z\omega^2)(x+y\omega^2+z\omega^4) \end{aligned}$$

这表明 $x^3+y^3+z^3-3xyz$ 可以因式分解为 $(x+y+z)(x+y\omega+z\omega^2)(x+y\omega^2+z\omega^4)$
对应地 $x^3-y^3-z^3-3xyz$ 可以因式分解为 $(x-y-z)(x-y\omega-z\omega^2)(x-y\omega^2-z\omega^4)$

(2) 求解思路

如果我们能将 $x^3+ux+v=0$ 化为
$x^3-y^3-z^3-3xyz=(x-y-z)(x-y\omega-z\omega^2)(x-y\omega^2-z\omega^4)=0$
那么方程的解就是 $\begin{cases} x_1=y+z \\ x_2=y\omega+z\omega^2 \\ x_3=y\omega^2+z\omega^4 \end{cases}$
因此我们只需求解 y,z 使得 $\begin{cases} yz=-\frac{u}{3} \\ y^3+z^3=-v \end{cases}$ 即可，即 $\begin{cases} y^3z^3=-\frac{u^3}{27} \\ y^3+z^3=-v \end{cases}$ (等价变换为 Vieta 定理的形式)

解关于 q 的一元二次方程 $q^2+vq-\frac{u^3}{27}=0$ 即可得 $y^3,z^3=-\frac{v}{2}\pm\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}$
不妨取:

$$\begin{cases} y=\sqrt[3]{-\frac{v}{2}+\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}} \\ z=\sqrt[3]{-\frac{v}{2}-\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}} \end{cases}$$

实际计算时，y,z 会出现不同组合 (因为复数会有 k 个不同的 k 次根)
为避免 y,z 不对应的情况，我们可以先解出一个 y，然后通过 $yz=-\frac{u}{3}$ 解出对应的 z.

因此方程 $x^3+ux+v=0$ 的解为:

$$\begin{cases} x_1=y+z=\sqrt[3]{-\frac{v}{2}+\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}}+\sqrt[3]{-\frac{v}{2}-\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}} \\ x_2=y\omega+z\omega^2=\sqrt[3]{-\frac{v}{2}+\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}}\omega+\sqrt[3]{-\frac{v}{2}-\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}}\omega^2 \\ x_3=y\omega^2+z\omega^4=\sqrt[3]{-\frac{v}{2}+\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}}\omega^2+\sqrt[3]{-\frac{v}{2}-\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}}\omega^4 \end{cases}$$

这就是形如 $x^3+ux+v=0$ 的三次方程的求根公式，称为 **Cardano-Tartaglia 公式**.
邵老师: "考试中如果 (注意是如果) 涉及三阶矩阵的求特征值问题，我默认你们是会解三次方程的."

因此一般的三次方程 $\tilde{x}^3+a\tilde{x}^2+b\tilde{x}+c=0$ 的求根步骤为:

- ① 计算 $\begin{cases} u=b-\frac{a^2}{3} \\ v=c-\frac{ab}{3}+\frac{2a^3}{27} \end{cases}$ (化简得到 $x^3+ux+v=0$, 其中 $x=\tilde{x}+\frac{a}{3}$)
- ② 计算 $\begin{cases} y=\sqrt[3]{-\frac{v}{2}+\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}} \\ z=\sqrt[3]{-\frac{v}{2}-\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}} \end{cases}$ (实际计算中先解出一个 y，然后通过 $yz=-\frac{u}{3}$ 解出对应的 z)
- ③ 计算 $\begin{cases} x_1=y+z \\ x_2=y\omega+z\omega^2 \\ x_3=y\omega^2+z\omega^4 \end{cases}$ (其中 $\omega=\exp\left(\frac{2\pi i}{3}\right)$, 注意 $\omega^4=\omega$)
- ④ 计算 $\begin{cases} \tilde{x}_1=x_1-\frac{a}{3} \\ \tilde{x}_2=x_2-\frac{a}{3} \\ \tilde{x}_3=x_3-\frac{a}{3} \end{cases}$

Matlab 代码:

```
function roots = solve_cubic(a, b, c)
    % Step 1: Calculate u and v
    u = b - (a^2) / 3;
    v = c - (a * b) / 3 + (2 * a^3) / 27;

    % Step 2: Calculate y and z
    discriminant = (v^2) / 4 + (u^3) / 27;
    y_all = cube_root(-v / 2 + sqrt(discriminant));
    y = y_all(1);
    z = -(u / 3) / y;

    % Step 3: Calculate the primary roots x1, x2, x3
    omega = exp(2 * pi * 1i / 3); % Cube root of unity
```

```
    x1 = y + z;
    x2 = y * omega + z * omega^2;
    x3 = y * omega^2 + z * omega;

    % Step 4: Adjust for the original variable by shifting with a/3
    root1 = x1 - a / 3;
    root2 = x2 - a / 3;
    root3 = x3 - a / 3;

    % Collect roots
    roots = [root1, root2, root3];

    % Verify the solution
    verify_cubic_solution(a, b, c, roots);
end

% 自定义的计算复数 z 所有三次根的函数
function roots_val = cube_root(z)
    % 输入 z 是复数
    % 输出 roots_val 是一个包含复数三次方根的向量

    % 计算复数的模和辐角
    r = abs(z);           % 复数的模
    theta = angle(z);     % 复数的辐角

    % 计算三次方根的模
    root_mod = r^(1/3);

    % 计算三次方根的所有解
    roots_val = root_mod * exp(1i * (theta + 2 * pi * (0:2)) / 3);
end

function verify_cubic_solution(a, b, c, roots)
    % Verify each root by substituting into the original cubic equation
    fprintf("equation: x^3 + a * x^2 + b * x + c = 0\n");
    fprintf("a = %.4f + %.4fi, b = %.4f + %.4fi, c = %.4f + %.4fi\n", ...
        real(a), imag(a), real(b), imag(b), real(c), imag(c));
    fprintf('Verifying solutions...\n');
    for i = 1:length(roots)
        x = roots(i);
        result = x^3 + a * x^2 + b * x + c;
        fprintf('Root %d: %.4f + %.4fi, Verification Result: %.4f + %.4fi\n',
...
            i, real(x), imag(x), real(result), imag(result));
    end
end
```

函数调用:

```
rng(51);
a = rand(1) + rand(1) * 1i; % coefficient for x^2 term
b = rand(1) + rand(1) * 1i; % coefficient for x term
c = rand(1) + rand(1) * 1i; % constant term

roots = solve_cubic(a, b, c);
disp('Calculated roots:');
disp(roots);
```

运行结果:

```
equation: x^3 + a * x^2 + b * x + c = 0
a = 0.6757 + 0.0447i, b = 0.3433 + 0.6440i, c = 0.2842 + 0.9493i
Verifying solutions...
Root 1: -0.9294 + -0.2385i, Verification Result: 0.0000 + -0.0000i
Root 2: -0.3641 + 0.9728i, Verification Result: -0.0000 + 0.0000i
Root 3: 0.6178 + -0.7790i, Verification Result: -0.0000 + 0.0000i
Calculated roots:
  -0.9294 - 0.2385i  -0.3641 + 0.9728i   0.6178 - 0.7790i
```

A.2 四次方程

(1) Ferrari 解法

对于一般的四次方程 $\tilde{x}^4 + q\tilde{x}^3 + r\tilde{x}^2 + s\tilde{x} + t = 0$,
我们同样可以对 $\tilde{x}$ 施加平移 $x = \tilde{x} + \frac{q}{4}$，消去三次项，将方程化为:

$$x^4 + (r - \frac{3q^2}{8})x^2 + (s - \frac{qr}{2} + \frac{q^3}{8})x + (t - \frac{qs}{4} + \frac{q^2r}{16} - \frac{3q^4}{256}) = 0$$

因此我们只需研究形如 $x^4 + ax^2 + bx + c = 0$ 的四次方程的求根公式.
其中:

$$\begin{cases} a = r - \frac{3q^2}{8} \\ b = s - \frac{qr}{2} + \frac{q^3}{8} \\ c = t - \frac{qs}{4} + \frac{q^2r}{16} - \frac{3q^4}{256} \end{cases}$$

我们对 $x^4 + ax^2 + bx + c = 0$ 配方，设 p 为待定系数，将方程化为:

$$x^4 + ax^2 + bx + c = (x^2 + p)^2 + (a - 2p)x^2 + bx + (c - p^2) = 0$$

我们希望取一个特定的 p 使得 $(a - 2p)x^2 + bx + (c - p^2)$ 为一个完全平方式
也就是说，我们需要令这个二次多项式的判别式 $\Delta = b^2 - 4(a - 2p)(c - p^2) = 0$
因此求解四次方程需要求解三次方程 $b^2 - 4(a - 2p)(c - p^2) = 0$ (除非我们能很容易地看出 p 的值)

$$b^2 - 4(a - 2p)(c - p^2) = 0$$
$$\Leftrightarrow$$
$$p^3 - \frac{a}{2}p^2 - cp - \frac{b^2 - 4ac}{8} = 0$$

令 $\tilde{p} = p - \frac{1}{6}a$ 可消除二次项得到 $\tilde{p}^3 + u\tilde{p} + v = 0$，其中 $\begin{cases} u = -\frac{a^2}{12} - c \\ v = -\frac{a^3}{108} - \frac{b^2}{8} + \frac{ac}{3} \end{cases}$

我们需要计算 $\begin{cases} y = \sqrt[3]{-\frac{v}{2} + \sqrt{\frac{v^2}{4} + \frac{u^3}{27}}} \\ z = \sqrt[3]{-\frac{v}{2} - \sqrt{\frac{v^2}{4} + \frac{u^3}{27}}} \end{cases}$ (实际计算中先解出一个 y，然后通过 $yz = -\frac{u}{3}$ 解出对应的 z)

根据 **Cardano-Tartaglia 公式**可取实根 $\tilde{p}_\star = y + z$
因此 $p_\star = \tilde{p}_\star + \frac{1}{6}a$ 可以使得 $(a - 2p)x^2 + bx + (c - p^2)$ 为完全平方式，即:

$$(a-2p_\star)x^2+bx+(c-p_\star^2)=(a-2p_\star)\left(x+\frac{b}{2(a-2p_\star)}\right)^2$$

因此四次方程 $x^4+ax^2+bx+c=0$ 可化为:

$$\begin{aligned}x^4+ax^2+bx+c&=(x^2+p_\star)^2+(a-2p_\star)x^2+bx+(c-p_\star^2)\\&=(x^2+p_\star)^2+(a-2p_\star)\left(x+\frac{b}{2(a-2p_\star)}\right)^2\\&=0\end{aligned}$$

即 $(x^2+p_\star)^2=(2p_\star-a)\left(x+\frac{b}{2(a-2p_\star)}\right)^2$
当 $2p_\star-a=0$ 时我们有 $x_{1,2}=\sqrt{-p_\star}$ 和 $x_{3,4}=-\sqrt{-p_\star}$
当 $2p_\star-a\neq 0$ 时等价于求解两个二次方程:

- $x^2+p_\star=\sqrt{2p_\star-a}\left(x+\frac{b}{2(a-2p_\star)}\right)=\sqrt{2p_\star-a}x-\frac{b}{2\sqrt{2p_\star-a}}$
- $x^2+p_\star=-\sqrt{2p_\star-a}\left(x+\frac{b}{2(a-2p_\star)}\right)=-\sqrt{2p_\star-a}x+\frac{b}{2\sqrt{2p_\star-a}}$

得到四个解为:

$$\begin{aligned}x_{1,2}&=\frac{1}{2}\left\{\sqrt{2p_\star-a}\pm\sqrt{(2p_\star-a)-4\left(p_\star+\frac{b}{2\sqrt{2p_\star-a}}\right)}\right\}\\x_{3,4}&=\frac{1}{2}\left\{-\sqrt{2p_\star-a}\pm\sqrt{(2p_\star-a)-4\left(p_\star-\frac{b}{2\sqrt{2p_\star-a}}\right)}\right\}\end{aligned}$$

因此一般的四次方程 $\tilde{x}^3+a\tilde{x}^2+b\tilde{x}+c=0$ 的求根步骤为:

- ① 计算 $\begin{cases}a=r-\frac{3q^2}{8}\\b=s-\frac{qr}{2}+\frac{q^3}{8}\\c=t-\frac{qs}{4}+\frac{q^2r}{16}-\frac{3q^4}{256}\end{cases}$ (化简得到 $x^4+ax^2+bx+c=0$, 其中 $x=\tilde{x}+\frac{q}{4}$)
- ② 计算 $\begin{cases}u=-\frac{a^2}{12}-c\\v=-\frac{a^3}{108}-\frac{b^2}{8}+\frac{ac}{3}\end{cases}$
- ③ 计算 $\begin{cases}y=\sqrt[3]{-\frac{v}{2}+\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}}\\z=\sqrt[3]{-\frac{v}{2}-\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}}\end{cases}$(实际计算中先解出一个 y，然后通过 $yz=-\frac{u}{3}$ 解出对应的 z)
- ④ 计算 $\tilde{p}_\star=\sqrt[3]{-\frac{v}{2}+\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}}+\sqrt[3]{-\frac{v}{2}-\sqrt{\frac{v^2}{4}+\frac{u^3}{27}}}$
- ⑤ 计算 $p_\star=\tilde{p}_\star+\frac{1}{6}a$
- ⑥ 若 $2p_\star-a=0$，则计算 $\begin{cases}x_{1,2}=\sqrt{-p_\star}\\x_{3,4}=-\sqrt{-p_\star}\end{cases}$

  若 $2p_\star-a\neq 0$，则计算 $\begin{cases}x_{1,2}=\frac{1}{2}\left\{\sqrt{2p_\star-a}\pm\sqrt{(2p_\star-a)-4\left(p_\star+\frac{b}{2\sqrt{2p_\star-a}}\right)}\right\}\\x_{3,4}=\frac{1}{2}\left\{-\sqrt{2p_\star-a}\pm\sqrt{(2p_\star-a)-4\left(p_\star-\frac{b}{2\sqrt{2p_\star-a}}\right)}\right\}\end{cases}$

- ⑦ 计算 $\begin{cases} \tilde{x}_1 = x_1 - \frac{q}{4} \\ \tilde{x}_2 = x_2 - \frac{q}{4} \\ \tilde{x}_3 = x_3 - \frac{q}{4} \\ \tilde{x}_4 = x_4 - \frac{q}{4} \end{cases}$

Matlab 代码为:

```
function roots = Ferrari_solve_quartic(q, r, s, t)
    % Display the original equation
    fprintf('Original equation: x^4 + q * x^3 + r * x^2 + s * x + t = 0\n');
    fprintf('q = %.4f + %.4fi, r = %.4f + %.4fi, s = %.4f + %.4fi, t = %.4f +
%.4fi\n', ...
        real(q), imag(q), real(r), imag(r), real(s), imag(s), real(t), imag(t));

    % Step 1: Calculate the transformed coefficients a, b, and c
    a = r - (3 * q^2) / 8;
    b = s - (q * r) / 2 + (q^3) / 8;
    c = t - (q * s) / 4 + (q^2 * r) / 16 - (3 * q^4) / 256;

    % Display transformed equation
    fprintf('Transformed equation: x^4 + a * x^2 + b * x + c = 0\n');
    fprintf('a = %.4f + %.4fi, b = %.4f + %.4fi, c = %.4f + %.4fi\n', ...
        real(a), imag(a), real(b), imag(b), real(c), imag(c));

    % Step 2: Calculate u and v
    u = -(a^2) / 12 - c;
    v = -(a^3) / 108 - (b^2) / 8 + (a * c) / 3;

    fprintf('u = %.4f, v = %.4f\n', u, v);

    % Step 3: Calculate p_star_tilde
    discriminant = (v^2) / 4 + (u^3) / 27;
    y_all = cube_root(-v / 2 + sqrt(discriminant));
    y = y_all(1);
    z = -(u/3) / y;
    p_star_tilde = y + z;
    fprintf('p_star_tilde = %.4f\n', p_star_tilde);

    % Step 4: Calculate p_star
    p_star = p_star_tilde + (a / 6);
    fprintf('p_star = %.4f\n', p_star);

    % Verification of the equation b^2 - 4(a - 2p_star)(c - p_star^2) = 0
    verify_equation(b, a, p_star, c);

    % Step 5: Calculate x1, x2, x3, x4 using the formula
    if abs(2 * p_star - a) < 1e-6
        x1 = sqrt(-p_star);
        x2 = x1;
        x3 = -x1;
        x4 = -x1;
    else
        sqrt_term = sqrt(2 * p_star - a);
        part1 = (2 * p_star - a) - 4 * (p_star + b / (2 * sqrt_term));
        part2 = (2 * p_star - a) - 4 * (p_star - b / (2 * sqrt_term));

        % Roots calculation
```

```
        x1 = (1 / 2) * (sqrt_term + sqrt(part1));
        x2 = (1 / 2) * (sqrt_term - sqrt(part1));
        x3 = (1 / 2) * (-sqrt_term + sqrt(part2));
        x4 = (1 / 2) * (-sqrt_term - sqrt(part2));
    end

    % Display calculated roots
    fprintf('x1 = %.4f + %.4fi, x2 = %.4f + %.4fi, x3 = %.4f + %.4fi, x4 = %.4f
+ %.4fi\n', ...
        real(x1), imag(x1), real(x2), imag(x2), real(x3), imag(x3), real(x4),
imag(x4));

    % Step 6: Adjust for the original variable by shifting with q/4
    tilde_x1 = x1 - q / 4;
    tilde_x2 = x2 - q / 4;
    tilde_x3 = x3 - q / 4;
    tilde_x4 = x4 - q / 4;

    % Collect roots
    roots = [tilde_x1, tilde_x2, tilde_x3, tilde_x4];

    % Verify the solution
    verify_quartic_solution(q, r, s, t, roots);
end

function verify_quartic_solution(q, r, s, t, roots)
    % Verify each root by substituting into the original quartic equation
    fprintf('Verifying solutions...\n');
    for i = 1:length(roots)
        x = roots(i);
        result = x^4 + q * x^3 + r * x^2 + s * x + t;
        fprintf('Root %d: %.4f + %.4fi, Verification Result: %.4f + %.4fi\n',
...
            i, real(x), imag(x), real(result), imag(result));
    end
end

% Verification function for the equation b^2 - 4(a - 2p_star)(c - p_star^2) = 0
function verify_equation(b, a, p_star, c)
    left_side = b^2 - 4 * (a - 2 * p_star) * (c - p_star^2);
    % fprintf('Verifying b^2 - 4(a - 2p_star)(c - p_star^2) = 0: %.4f\n',
left_side);
    if abs(left_side) < 1e-6
        fprintf('The equation b^2 - 4(a - 2p_star)(c - p_star^2) is
satisfied.\n\n');
    else
        fprintf('The equation b^2 - 4(a - 2p_star)(c - p_star^2) is not
satisfied.\n\n');
    end
end

% 自定义的计算复数 z 所有三次根的函数
function roots_val = cube_root(z)
    % 输入 z 是复数
    % 输出 roots_val 是一个包含复数三次方根的向量

    % 计算复数的模和辐角
    r = abs(z);           % 复数的模
```

```
    theta = angle(z);      % 复数的辐角

    % 计算三次方根的模
    root_mod = r^(1/3);

    % 计算三次方根的所有解
    roots_val = root_mod * exp(1i * (theta + 2 * pi * (0:2)) / 3);
end
```

函数调用:

```
rng(51);
q = rand(1) + rand(1) * 1i; % coefficient for x^3 term
r = rand(1) + rand(1) * 1i; % coefficient for x^2 term
s = rand(1) + rand(1) * 1i; % coefficient for x term
t = rand(1) + rand(1) * 1i; % constant term

roots = Ferrari_solve_quartic(q, r, s, t);
disp('Calculated roots:');
disp(roots);
```

运行结果:

```
Original equation: x^4 + q * x^3 + r * x^2 + s * x + t = 0
q = 0.6757 + 0.0447i, r = 0.3433 + 0.6440i, s = 0.2842 + 0.9493i, t = 0.1577 +
0.3880i
Transformed equation: x^4 + a * x^2 + b * x + c = 0
a = 0.1728 + 0.6214i, b = 0.2207 + 0.7317i, c = 0.1252 + 0.2434i
u = -0.0955, v = 0.0194
p_star_tilde = 0.4214
p_star = 0.4502
The equation b^2 - 4(a - 2p_star)(c - p_star^2) is satisfied.

x1 = 0.8970 + -0.7495i, x2 = -0.0314 + 0.8966i, x3 = -0.4047 + 0.2010i, x4 =
-0.4609 + -0.3481i
Verifying solutions...
Root 1: 0.7280 + -0.7607i, Verification Result: -0.0000 + 0.0000i
Root 2: -0.2003 + 0.8855i, Verification Result: 0.0000 + 0.0000i
Root 3: -0.5736 + 0.1898i, Verification Result: -0.0000 + 0.0000i
Root 4: -0.6298 + -0.3593i, Verification Result: -0.0000 + 0.0000i
Calculated roots:
   0.7280 - 0.7607i  -0.2003 + 0.8855i  -0.5736 + 0.1898i  -0.6298 - 0.3593i
```

(2) Descartes-Euler 解法

(TODO)

A.3 历史趣闻

16世纪初，意大利数学家 https://en.wikipedia.org/wiki/Scipione_del_Ferro (可能是在另一位数学家 Pacioli 的启发下)
解决了形如 $x^3+ux+v=0$ 的三次方程的求根问题，
这在求解一元三次方程问题的道路上是一个突破性的进展.
然而他没有发现对于一般的三次方程 $x^3+ax^2+bx+c=0$，
可以通过平移变换成缺二次项的三次方程的形式.

当年数学的绝招都是独门秘籍不公布的，因此 Ferro 并没有发表自己的成果，
选择对解法保密——反正你只要把三次方程给我，我帮你解出来，把解代入方程验证你总是会的吧?

后来 Ferro 把解法传授给了自己的弟子 **Fiore** (而 Ferro 的手稿由女婿 **Nave** 继承)，
告诉他只要你学会这个东西，你就有口饭吃了.
Fiore 学得一知半解，但却对外吹嘘自己能够解所有的一元三次方程.
另一位意大利数学家 https://en.wikipedia.org/wiki/Nicolo_Tartaglia (原名 Fontana，"Tartaglia" 是他的外号，意为"口吃的人")，
他比较缺钱，当时有一种赚钱的方法，就是去大学里挑战大学教授，互相给对方出题，
谁解出来的题多谁就能赢得奖金，甚至可以赢得对方的职位.
他听说 Fiore 号称会解三次方程，心想为什么别人会解自己却不会解，
于是熬了一个晚上 (According to Professor Meiyue Shao) 想出了三次方程的解法.
而且他不但想出来 $x^3+ux+v=0$ 的解法，
他还发现 $x^3+ax^2+bx+c=0$ 可以等价变换为形如 $x^3+ux+v=0$ 的方程.
比赛的结果可想而知，"Tartaglia" 以 $30:0$ 的比分战胜了 Fiore，
因为"Tartaglia" 给 Fiore 出的都是一般的三次方程，
(当然，由于当时还没有负数的概念，所以这些方程的系数都是正数)
而 Fiore 给 "Tartaglia" 出的都是缺二次项的三次方程.
于是 "Tartaglia" 一战成名，还被聘为大学教授，生活贫困的问题一下子就解决了.
和 Ferro 一样，"Tartaglia" 也选择将自己的解法保密，毕竟这是吃饭的本事.

与 "Tartaglia" 同时代的还有一位意大利数学家，名叫
[https://en.wikipedia.org/wiki/Gerolamo_Cardano]
(https://en.wikipedia.org/wiki/Gerolamo_Cardano)，
他是一名百科全书式的学者，兼具多重身份：
数学家、生物学家、物理学家、化学家、占星术士、天文学家、哲学家、作家和赌博爱好者
(是的，他是概率学的奠基人之一，提出了二项式定理)，
这和他父亲的朋友 https://en.wikipedia.org/wiki/Leonardo_da_Vinci 非常相像.
Cardano 对三次方程的解法很感兴趣，因此花重金向 "Tartaglia" 购买解法.
"Tartaglia" 禁不住金钱的诱惑，在 Cardano 发誓保密的前提下，
将三次方程的解法以口诀的形式告诉了 Cardano.
然而该口诀晦涩难懂，Cardano 和他的学生 https://en.wikipedia.org/wiki/Lodovico_Ferrari 花了十年才从中破解出三次方程的解法，
Ferrari 还在此基础上发明了四次方程的解法，
但碍于对 "Tartaglia" 的誓言，三次方程、四次方程的解法都不能发表.
于是 Cardano 想到从 Ferro 的女婿 Nave 那里买来 Ferro 的手稿，
并宣称 Cardano 是从 Ferro 的手稿习得并完善了三次方程的求根公式，
终于得以将数年成果公之于众.
在他1545年的著作 ***Ars Magna* (大艺术)** 中，Cardano 开创性地使用了负数的概念，
同时还提出了对负数开方的观点，这些都是天才般的构想，尽管他对复数的性质没有深入研究.

不出所料，"Tartaglia" 感受到了深深的背叛，
尽管 Cardano 在书中将三次方程的解法归功于 "Tartaglia"，
但他依然痛斥 Cardano 背信弃义，并提出司法诉讼.
眼看事情即将一发不可收拾，
Cardano 的那位学生 Ferrari 向 "Tartaglia" 提出挑战，希望以此结束这场口舌之争.

"Tartaglia" 当然不想应战，他已经功成名就了，没有必要冒险与一个穷学生比试，
但大学校方逼他应战，如果不应战就要开除他.
这场比试的结果可想而知，"Tartaglia" 给 Ferrari 出的都是三次方程，
但 Ferrari 给 "Tartaglia" 出的都是四次方程，
面对这种降维打击，"Tartaglia" 只得认输，还因此丢了自己的职位，
他的晚年穷困潦倒，最终病死于威尼斯.
然而，广泛流传的 "Tartaglia" 余生致力于诋毁 Cardano 的故事似乎完全是捏造的.
数学历史学家现在将三次方程的求根公式同时归功于 Cardano 和 "Tartaglia"，
称之为 Cardano-Tartaglia 公式.

至于 Cardano，根据邵老师的讲述，他曾预言自己会在75岁的某一天逝世，
但当那一天真的到来之时，Cardano 依旧生龙活虎，
他为了使自己的预言成真，于是在那一天自杀了.
https://hsm.stackexchange.com/questions/15690/did-cardano-predict-the-date-of-his-death-then-commit-suicide-on-that-date，实际上文艺复兴时期的历史记录往往充满了神秘色彩，
尤其是涉及重要人物的个人生活和死亡细节的时候.
不过比起预言失败，Cardano 在晚年遭遇了更大的挫折，
他的大儿子 Giovanni (同时也是他最喜欢的儿子) 因谋杀不忠的妻子而被处死，
起因是 Giovanni 发现自己的三个孩子都不是自己的后代.
Cardano 的二儿子 Aldo 是个赌徒，因为总是从 Cardano 那里偷钱而被剥夺了继承权.
Cardano 的女儿 Chiara 死于梅毒.
儿女的不幸和不肖让 Cardano 伤心欲绝，
他曾在书中回忆他与亡妻 Lucia 1531至1546年的生活，称那是他一生中最快乐的时光.
Cardano 还因 ***De rerum varietate (万物的变化)*** 一书被宗教裁判所逮捕，
他被迫公开放弃信仰，他所有的非医学作品都被列为禁书.
后来 Cardano 移居罗马，在那里他被教皇 Gregory XIII 给予了终身年金，并完成了他的自传.
之后的日子里，他任教于皇家医师学院，继续从事他的医学和哲学研究，
直至1576年去世，终年75岁.

我们的另一位主角，Ferrari，他在 Cardano 研究三次方程和四次方程解法的过程中起到了重要作用，且四次方程解法主要是 Ferrari 发现的.
Cardano 后来辞去教职时推荐了 Ferrari，他也因此在十几岁的年纪便获得了大学教职.
Ferrari 在42岁那年退休，而且非常富有，
他随后回到了家乡博洛尼亚，并与他守寡的表妹 Maddalena 生活在一起.
1565年10月 Ferrari 死于砷中毒，终年43岁.
据传说，是 Maddalena 所为.

从数学史上来看，"Tartaglia"、Cardano、Ferrari 的成果是革命性的，
因为他们第一次解出了古希腊数学家没有解出来的问题.
从古希腊到文艺复兴前的1000多年里，整个欧洲的数学基本上没有任何进展，
期间例如 Fibonacci 数列之类的问题，如果古希腊的数学家还活着，也能得到解答，
只是后世的数学家想到去研究这样一个增长的问题.
但三次方程、四次方程的问题古希腊是铁定解不了的，其他古文明也都解不了，
例如中国古代解高阶方程是近似求解，即数值解，但这里给出的求根公式是精确解.
这样的成果在当时是革命性的，使得数学家们的思想开始转变了，
因为以前他们认为老祖宗留下的东西都是好的，数学中该研究的东西老祖宗都研究透了，
但文艺复兴时期他们解出求根公式之后发现，科学还是要往前发展的，
例如 Cardano 就认为负数乃至负数的开方都是值得研究的.
包括 Cardano 的著作***Ars Magna (大艺术)*** 对后世的影响也很大，
就是大家做出成果不是藏在手里找别人决斗去的，而是拿出来发表，
发表之后依然承认这是你的东西，这对整个科学界 (不仅仅是数学界) 具有深远的影响.

同时我们也不能忘记 **Ferro**，他是第一个有记录的解出缺二次项的三次方程求根公式的人，
尽管他没有将结论推广到一般的三次方程，尽管他的求根公式是不完备的，我们依然要清楚 "第一个" 和 "第二个" 的区别——有人告诉你这个问题解得出来，和没人告诉是有区别的.
Ferro 提供了第一个三次方程解法 (缺二次项)，
"Tartaglia" 熬了一晚上想出了一般的三次方程的解法，
Cardano 完善了求根公式，系统性地使用了负数和负数开方的概念，
Ferrari 在三次方程解法的基础上想到了四次方程的解法，
这四人的天才程度与现如今高中生推出三次方程求根公式的天才程度是不可同日而语的.

短短30年间，三次方程、四次方程都找到了相应的解法，那么五次方程是不是不远了？
这给当时的数学界带来了极大的鼓舞，
一代又一代的数学家尝试解出五次方程的求根公式，都没能解出来.
直到 https://en.wikipedia.org/wiki/Joseph-Louis_Lagrange 的时代 (18世纪后半叶)，
Lagrange 总结前人的成果，把二次方程、三次方程、四次方程的求根公式整理在一起，
发现三次方程本质上只有一种解法，四次方程有不同解法.
经过对这些方法的整理，Lagrange 发现要解一个五次方程，先要解一个六次方程，
这基本上预示着五次方程没有一般的求根公式.
同时，他发现解的地位都是平等的，用置换的概念去研究是可行的，
他的思想为后来的 **Abel** 和 **Galois** 采用并发展，演变为 **Galois 理论**，
终于证明了高于四次的一般方程不能用代数方法求解.

THE END