# 目录

# Use the Newton-Raphson method to solve non-linear equations

phys.2001 孙陶庵 202011010101

2023 年 3 月 6 日

# 1 Newton-Raphson method

## 1.1 code

```
clear
x0 = [0.1,0.1,-0.1];
eps = 1e-5;
maxiter = 10;
F = @(x)([3*x(1) - cos(x(2)*x(3)) - 1/2;
    x(1)^2 - 81*(x(2) + 0.1)^2 + sin(x(3)) + 1.06;
    exp(-x(1)*x(2)) + 20*x(3) + (10*pi - 3)/3]);
J = @(x)([3 -x(3)*sin(x(2)*x(3)) -x(2)*sin(x(2)*x(3));
    2*x(1) -162*(x(2) + 0.1) cos(x(3));
    -x(2)*exp(-x(1)*x(2)) -x(1)*exp(-x(1)*x(2)) 20]);
x = double(x0);
for i = 1:maxiter
    Jx = J(x);
    Fx = F(x);
```

```
        delta = - Jx \ Fx;
        x = x + delta;
        if norm(delta) < eps
            return;
        end
    end
    x


if i == maxiter
    error('This method can''t execute');
end


```

Listing 1: Newton-Raphson method

## 1.2 Principle explanation [1] [2]

Before we get to the code, we need to know how the method works. The Newton-Raphson method is an approximate solution to equations over real numbers and real complex number fields. This method uses the antecedent of the Taylor series of the function $f(x)$ to find the roots The equation $f(x) = 0$.

First choose a $x_0$ close to the zero of the function $f(x)$ and compute the corresponding $f(x_0)$ and the slope of the tangent line $f'(x_0)$ (where $f'$ represents the derivative of the function $f$). Then we calculate the intersection of the line through the points $(x_0, f(x_0))$ and $f'(x_0)$ axis and the $x$-coordinate of the $x$-axis, which is the solution of the following equation.

$$0 = (x - x_0) \cdot f'(x_0) + f(x_0)$$

We name the newly obtained $x$ coordinates of the point as $x_1$. Usually, $x_1$ is closer to the solution of the equation $f(x) = 0$ than $x_0$. So we can now start the next iteration from $x_1$. The iteration equation can be simplified to:

$$x_{n+1} = x_n - \tfrac{f(x_n)}{f'(x_n)}$$

It has been proved that the quadratic convergence of the Newton iteration method must meet the following conditions: $f'(x) \neq 0$; For all $x \in I$, where $I$ is the interval $[\alpha - \gamma, \alpha + \gamma]$, and $x_0$ is in the interval $I$, namely $r \geqslant |a - x_0|$; $f''(x)$ is continuous for all $x \in I$;$x_0$ is close enough to the root $\alpha$. However, the method above is just a function solver. If we want to get the algorithm for nonlinear 'equations' like follow:

$$\begin{cases} 3x_1 - \cos(x_2 x_3) - \tfrac{1}{2} = 0 \\ x_1^2 - 81(x_2 + 0.1)^2 + \sin(x_3) + 1.06 = 0 \\ e^{-x_1 x_2} + 20x_3 + \tfrac{10\pi - 3}{3} = 0 \end{cases}$$

Then we need to expand the definition from $f'(x_n)$ to $\nabla f(x)$. The equations' form will be changed as follow [3]:

$$\begin{bmatrix} \dfrac{\partial \mathbf{f}}{\partial x_1} & \cdots & \dfrac{\partial \mathbf{f}}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \nabla^{\mathrm{T}} f_1 \\ \vdots \\ \nabla^{\mathrm{T}} f_m \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \cdots & \dfrac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_m}{\partial x_1} & \cdots & \dfrac{\partial f_m}{\partial x_n} \end{bmatrix}$$

This matrix are called Jacobian matrix. And we will get:

$$x_{n+1} = x_n - J^{-1}(x_n) f'(x_n)$$

### 1.3 code-explanation

We can see the code above. We use anonymous function to define 2 functions first, F is original function. The next "function" is Jacobi matrix:

$$J(x) = \begin{pmatrix} 3 & -x_3 \sin(x_2 x_3) & -x_2 \sin(x_2 x_3) \\ 2x_1 & -162(x_2+0.1)\cos(x_3) & -x_1 \exp(-x_1 x_2) \\ -x_2 \exp(-x_1 x_2) & -x_1 \exp(-x_1 x_2) & 20 \end{pmatrix}$$

Now, we can get the final result:

$$\begin{pmatrix} 0.5000 & 0.5000 & 0.3000 \\ 0.0000 & 0.0000 & -0.2000 \\ -0.5236 & -0.5236 & -0.7236 \end{pmatrix}$$

## 2 Gauss-Seidel method

```
x = [0.1,0.1,-0.1];
eps = 1e-5;
err = 10;

while(err>eps)
    y = g(x);
    err = norm(y-x);
    x = y;
end

x

function y = g(x)
    y(1) = 1/3*cos(x(2)*x(3))+1/6;
    y(2) = 1/9*sqrt(x(1)^2+sin(x(3))+1.06)-0.1;
```

```
    y(3) = -exp(-x(1)*x(2))/20-(10*pi-3)/60;
end
```

Listing 2: Newton-Raphson method

The above code is derived from the class.

# 3 Jacobian method

I'm so sorry that I can't accurately understand the difference between Gauss-Seidel algorithm and Jacobi algorithm.

# 4 Convergence speed for iterative methods

The principle of convergence speed has been paid in the reference [4], so I won't go into details here.

## 4.1 Newton-Raphson method

```
    clear
    x0 = [0.1,0.1,-0.1];
    eps = 1e-5;
    maxiter = 10;
    F = @(x)([3*x(1) - cos(x(2)*x(3)) - 1/2;
        x(1)^2 - 81*(x(2) + 0.1)^2 + sin(x(3)) + 1.06;
        exp(-x(1)*x(2)) + 20*x(3) + (10*pi - 3)/3]);
    J = @(x)([3 -x(3)*sin(x(2)*x(3)) -x(2)*sin(x(2)*x(3));
        2*x(1) -162*(x(2) + 0.1) cos(x(3));
        -x(2)*exp(-x(1)*x(2)) -x(1)*exp(-x(1)*x(2)) 20]);
    x = double(x0);
```

```
    alpha111 = zeros(1, maxiter);

    for i = 1:maxiter
        Jx = J(x);
        Fx = F(x);
        delta = - Jx \ Fx;
        x = x + delta;
        alpha111(i) = norm(Fx);
        if norm(delta) < eps
            break;
        end
    end
    semilogy(1:i, alpha111(1:i), 'o-');
    xlabel('Iteration');
    ylabel('alpha');
```

Listing 3: Newton-Raphson method with Convergence speed for iterative methods

## 4.2 Gauss-Seidel method

```
    x = [0.1,0.1,-0.1];
    eps = 1e-5;
    err = 10;
    order11 = [];

    while(err>eps)
        y = g(x);
        err = norm(y-x);
        order11 = [order11 err];
        x = y;
```

```
    end

    p = polyfit(log(order11(1:end-1)), log(order11(2:end)), 1);
    convergence_rate = p(1);

    semilogy(order11);
    title(['Convergence rate: ', num2str(convergence_rate)]);
    xlabel('Iteration');
    ylabel('Error');


    function y = g(x)
        y(1) = 1/3*cos(x(2)*x(3))+1/6;
        y(2) = 1/9*sqrt(x(1)^2+sin(x(3))+1.06)-0.1;
        y(3) = -exp(-x(1)*x(2))/20-(10*pi-3)/60;
    end
```

Listing 4: Gauss-Seidel method with Convergence speed for iterative methods

### 4.3 Jacobi method

## 5 Convergence Rate Comparison of Different Algorithms

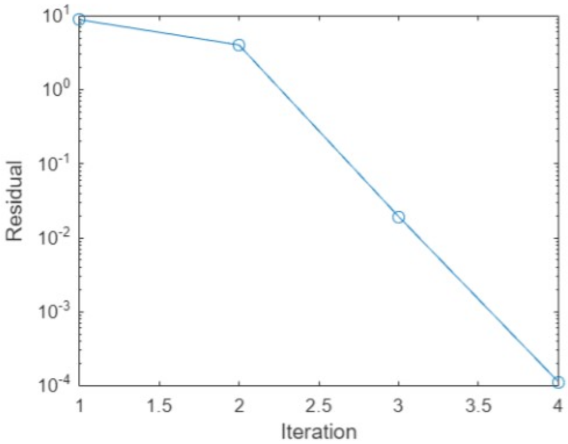


图 1: newtonlaw

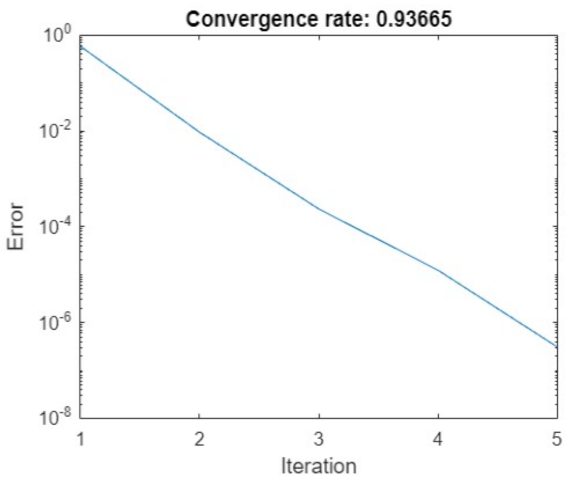


图 2: Gauss

## 参考文献


[1] E. W. Weisstein, “Newton’s method.” [Online]. Available: https://mathworld.wolfram.com/NewtonsMethod.html

[2] X. Wu, "Roots of equations." [Online]. Available: https://www.ece.mcmaster.ca/~xwu/part2.pd

[3] Wikipedia contributors, "Jacobian matrix and determinant — Wikipedia, the free encyclopedia," 2023, [Online; accessed 6-March-2023]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Jacobian_matrix_and_determinant&oldid=1141352419

[4] D. Hundley, "Rate of convergence." [Online]. Available: http://people.whitman.edu/~hundledr/courses/M467F06/ConvAndError.pdf