数学应用软件大型实验实验报告

实验序号：B63　　日期：

班级		姓名		学号	
实验名称	牛顿迭代收敛域绘图				

问题背景描述：

牛顿迭代公式可以直接用来求解复数方程 $z^3-1=0$，在复平面上他的三个根是 $z_1^*=1$，$z_{2,3}^*=-\frac{1}{2}\pm\frac{\sqrt{3}}{2}i$。选择中心位于坐标原点边长为 2 的正方形内的点为初始值，把收敛到三个不同根的初始值分别标上不同的颜色。只要计算足够多的点，你将得到牛顿法收敛域的彩色图形。

实验目的：

1．能使用牛顿迭代公式进行迭代；

2．能通过迭代的结果进行判断，并画出收敛域。

实验原理与数学模型：

1．因为牛顿迭代法必须构造收敛的迭代公式，令 $f(z)=z^3-1, f'(z)=3z^2$，迭代公式为

$$z_{n+1}=z_n-\frac{z_n^3-1}{3z_n^2};$$

2．由于中心位于坐标原点边长为 2 的正方形内的点是初始值，不妨建立一种取点方式为

$\begin{cases}x_m=-1+m\times h\\ y_n=-1+n\times h\end{cases}$，其中 h 是隔多少距离取点，可设为 0.002，那么 $m,n=0,1,2,\cdots,\frac{2}{h}=1000$，

于是复平面上的点就是 $z_{mn}=x_m+iy_n$ 或者表示为 $z_{mn}(x_m,y_n)$；

3．每个点代入迭代公式，依经验最多迭代 20 次，收敛的点即可收敛到对应的收敛点；

4．那么，在这片复平面上的点有三种情况，要么不收敛，要么收敛到 z_1^* 或 z_2^* 或 z_3^*，我们可以把这个数据放入一个 1001×1001 的矩阵 Z 中，对其中的某一个元素 a_{ij} 来说，0 表示不收敛，1、2、3 分别表示收敛到 z_1^*、z_2^*、z_3^*；

5．最后只要针对矩阵 Z 中的不同值绘图时使用不同的颜色即可。

实验所用软件及版本：

Matlab R2016b

主要内容（要点）：

1. 生成 1001×1001 的复数矩阵 G，G(i, j) = complex(-1 + h * i, -1 + h * j)；
2. 生成 1001×1001 的复数矩阵 Z；
3. 对矩阵 Z 的每个元素进行牛顿迭代 20 次；
4. 通过迭代的结果，若 abs(G(i, j) - x(1)) < h，则说明 G(i, j)收敛到根 x(1)，收敛到其他根也是同理，若均不满足，就可认为是不收敛；
5. 根据结果，先利用 find 命令将收敛到不同点的位置依次赋给 OPQR，再使用 plot 命令进行绘图，并选择点图与不同颜色加以区分。

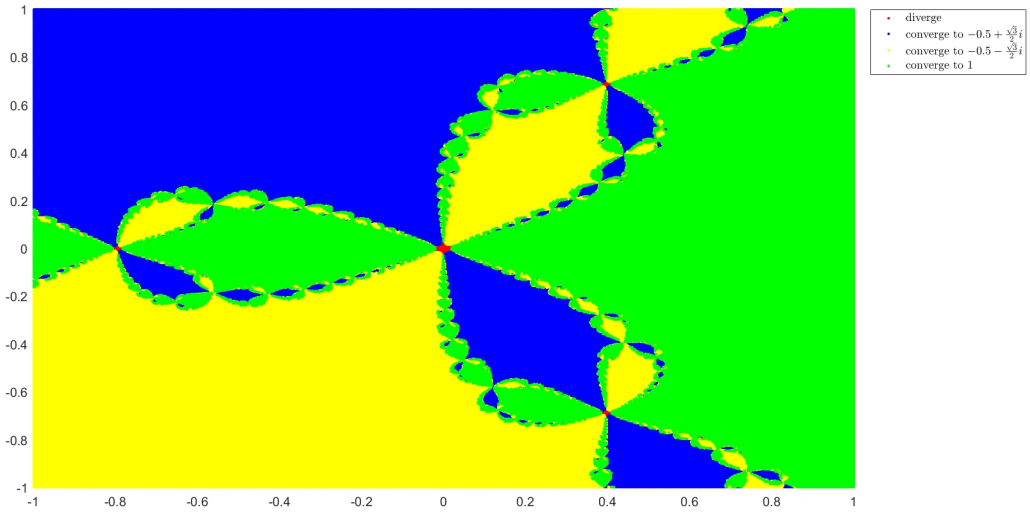
实验过程记录（含：基本步骤、主要程序清单及异常情况记录等）：

```
% x 是 z^3 - 1 = 0 的根，h 是步长，m + 1 是总点数
% G 用来存储每个点迭代后的值，Z 用来存储收敛情况
% f 是迭代公式对应的函数句柄
% O、P、Q、R 用来存储 Z 中 0、1、2、3 的位置，并转换为平面中的位置
% 绘图时，红色不收敛，蓝色收敛到 z_1，黄色收敛到 z_2，绿色收敛到 z_3
x = roots([1, 0, 0, -1]);
h = 0.002;
m = 2 / h;
G = zeros(m + 1, m + 1);
Z = zeros(m + 1, m + 1);
f = @(z) z - (z.^3 - 1) ./ (3 * z.^2);

for i = 1:m + 1

    for j = 1:m + 1
        G(i, j) = complex(-1 + h * i, -1 + h * j);

        for k = 1:20
            G(i, j) = feval(f, G(i, j));
        end

        if abs(G(i, j) - x(1)) < 0.001
            Z(i, j) = 1;
        elseif abs(G(i, j) - x(2)) < 0.001
            Z(i, j) = 2;
        elseif abs(G(i, j) - x(3)) < 0.001
            Z(i, j) = 3;
        end

    end

end
```

```
[O1, O2] = find(Z == 0);
[P1, P2] = find(Z == 1);
[Q1, Q2] = find(Z == 2);
[R1, R2] = find(Z == 3);
O1 = -1 + O1 .* h;
O2 = -1 + O2 .* h;
P1 = -1 + P1 .* h;
P2 = -1 + P2 .* h;
Q1 = -1 + Q1 .* h;
Q2 = -1 + Q2 .* h;
R1 = -1 + R1 .* h;
R2 = -1 + R2 .* h;

plot(O1, O2, '.r', P1, P2, '.b', Q1, Q2, '.y', R1, R2, '.g')
axis([-1, 1, -1, 1])
l = legend('diverge', 'converge to $-0.5+\frac{\sqrt{3}}{2} i$', 'converge
to $-0.5-\frac{\sqrt{3}}{2} i$', 'converge to 1');
set(l, 'Interpreter', 'latex', 'Location', 'NorthEastOutside')
```

实验结果报告与实验总结：

1．输出结果：



2．实验总结：

(1) 本实验的主要难点有两个，一个是如何进行迭代计算某个点的收敛情况，另一个难点是如何在得到结果矩阵 Z 的基础上进行绘图；

(2) 通过结果的收敛图，能较清楚地知道每个点的收敛情况。

思考与深入：

1．本题通过嵌套循环完成对点的迭代计算，当点的数量较多时，速度会较慢，如何改进计算速度是值得思考的；