

Julia 集的分析和探索

吴泓鹰

数学与应用数学 (强基计划) 3210101890

2022 年 7 月 4 日

摘要

本文介绍了 Julia 集的数学理论，并得到其图像的生成算法，进而使用 Python 程序实现生成不同条件下的 Julia 集图像的程序，探索得到 Julia 集在数学中的特征性质以及美丽之处，此外还对 Mandelbrot 集与 Julia 集的关系进行了简单分析。

1 引入：问题背景

Julia 集由法国数学家 Gaston Julia¹发现并命名，其是在复平面上构成分形点的集合。在复动力系统²的背景下，Julia 集由使得任意小的扰动就会使迭代函数序列发生剧烈变化的值组成，这体现了 Julia 集行为上的混沌性³。关于 Julia 集有很多有趣和美丽的地方，下面将对 Julia 集进行简单的分析和探索。^[1]

2 分析：生成 Julia 集

2.1 数学理论

Julia 集一般可以定义为使得复函数 $f_c(z) = z^2 + c$ 经过无数次迭代后能够不发散的复数 z 的集合，其中 $c \in \mathbb{C}$ ，我们一般取 $|c| < 2$ 。^[2]

进一步的，令 $f(z)$ 为有理函数，其定义为 $f(z) := \frac{p(z)}{q(z)}$ ，其中 $z \in \mathbb{C}^*, \mathbb{C}^*$ 为黎曼球面 $\mathbb{C} \cup \{\infty\}$, p, q 为没有公因式的多项式。则拓展后的 Julia 集 J_R 为在经过无数次 $f(z)$ 迭代后不发散的 z 的集合，而真正的 Julia 集 J 为 J_R 的边界 ∂J_R 。^[4]

¹详见Gaston Julia

²一个数学分支，详见Complex dynamics。

³混沌理论相关，详见Chaos theory。

2.2 算法实现

为了在计算机上生成 Julia 集的图像，我们首先需要一个算法来实现这一过程。考虑到计算机能够处理的迭代次数是有限的，因此我们把“不发散”理解为“ $|z| < 2$ ”，并且每经过一次迭代，我们将其中“不发散”点的颜色改变一点，“发散”点则保留其原来的颜色，这样我们就能清晰地观察到生成图像的特征。具体算法的流程图如下：[3][5]

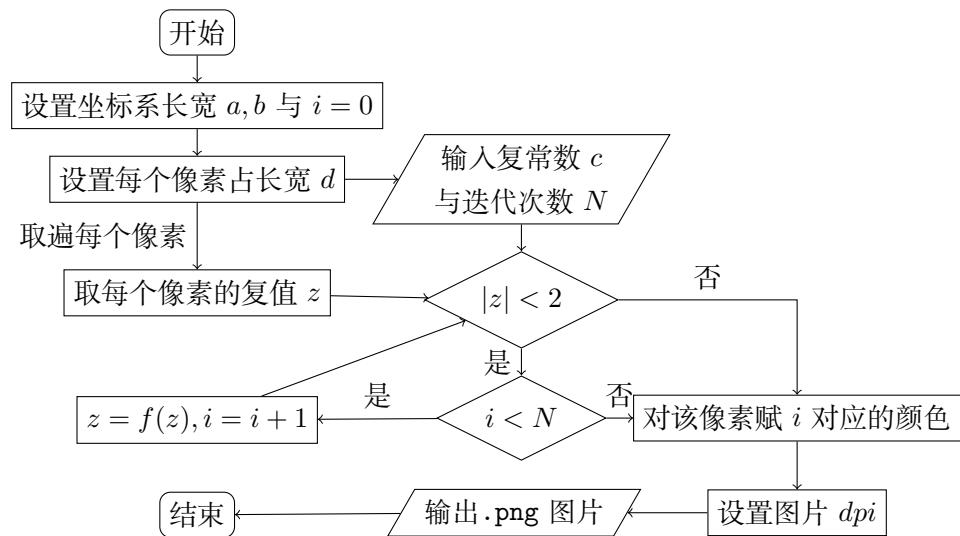


图 1: Junia 集图像生成的算法流程图

具体而言，为了实现该算法，我使用了 Python 编程语言进行程序设计⁴，其中使用了库numpy与matplotlib，并且添加了一些细节，使用 ipython3 编译得到各式各样的 Junia 集的图像。

3 探索：不同条件下的 Julia 集

以下的所有数值算例均使用matplotlib库中的imshow函数生成，并且一般而言，我们取 $f(z) = z^2 + c$ 。设置的部分参数⁵如下：

标清	a	b	d	dpi	高清	a	b	d	dpi
	3.0	3.0	0.003	300		3.0	3.0	0.001	600

⁴源代码见src目录下的文件。

⁵以下生成的图片均根据标清参数生成，高清图片可到png目录下查阅。

3.1 改变迭代次数 N

取 $c = -0.4 + 0.6i$ [1]，再分别取 $N = 20, 50, 100, 200$ ，如图2所示。可以观察到：随着迭代次数的增加，Julia 集的图像的精细程度越来越高，同时也体现出了更明显的自相似性。

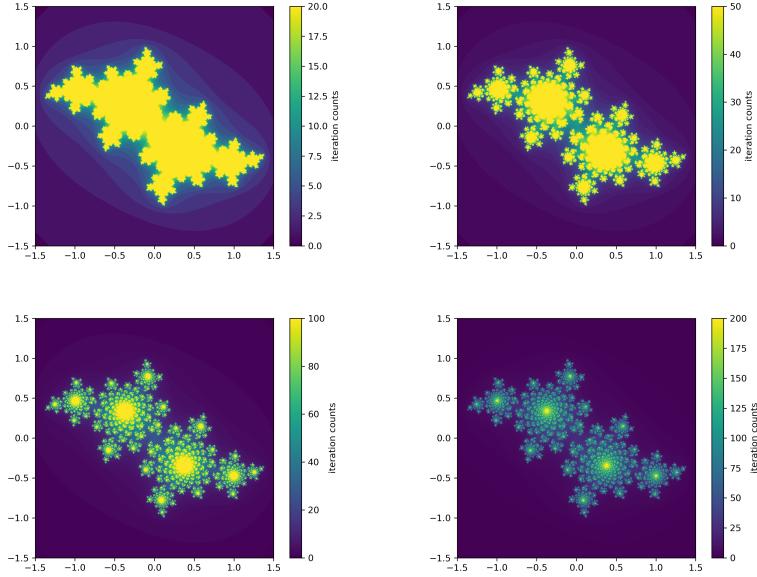
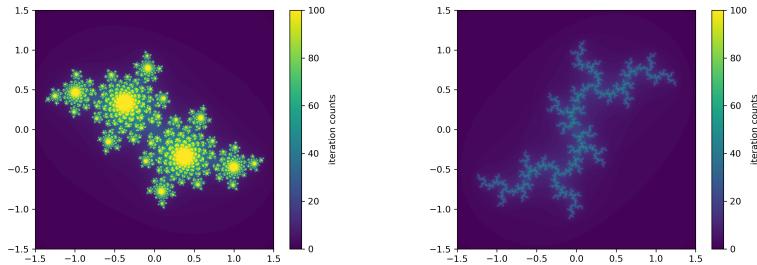
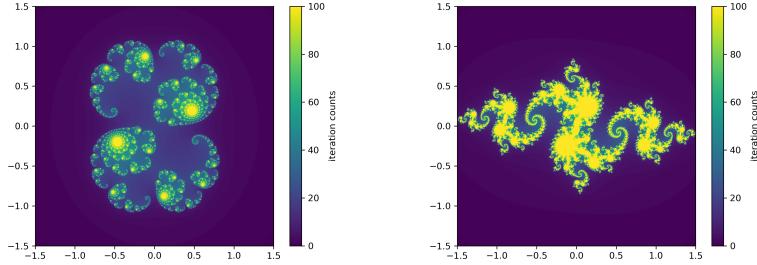


图 2: $c = -0.4 + 0.6i, N = 20, 50, 100, 200$

3.2 改变复常数 c

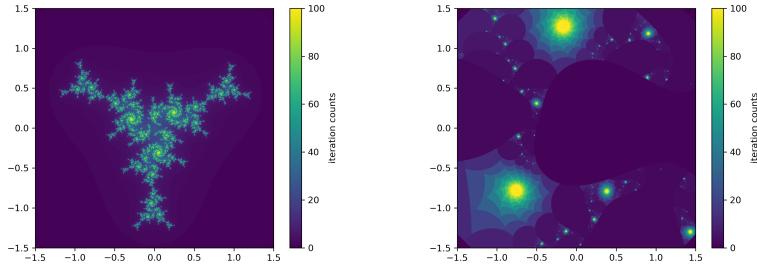
取 $N = 100$ ，再分别取 $c_1 = -0.4 + 0.6i, c_2 = -0.8i, c_3 = 0.285 + 0.01i, c_4 = -0.1 + 0.651i$ [1]，如图3所示。可以观察到，对于不同的 c 值而言，Julia 集的图像差异极大，这正体现了前文中所提到的 Julia 集具有的“混沌”性质；同时无论 c 值如何变化，得到的图像总是自相似且对称的，这也是为什么我们还会将 Julia 集描述为复平面上分形点的集合的原因。



图 3: $N = 100, c = c_1, c_2, c_3, c_4$

3.3 改变迭代公式 $f(z)$

取 $N = 100$, 令 $f_1(z) = z^3 - 0.071 - 1.134i$ [3], $f_2(z) = \frac{(1 - \frac{z^3}{6})}{(z - \frac{z^2}{2})^2} - 0.616 + 0.45i$ [1], 如图4所示。可以观察的到, 即使 $f(z)$ 改变了, 生成的 Julia 集的图像仍然具有分形的性质, 在这之中体现出了数学之美。

图 4: $N = 100, f(z) = f_1(z), f_2(z)$

4 拓展: Julia 集和 Mandelbrot 集之间的关系

实际上 Julia 集和 Mandelbrot 集就是同一函数 $f(z)$ 对不同参数的递归过程, Julia 集是固定复常数 c , 取遍复数 z 进行递归, 而 Mandelbrot 集则是固定递归初值 z_0 取遍复数 c 进行递归。而且更深入的来看, Mandelbrot 集就是对 Julia 集的一个概括 [3], 这是因为在 Mandelbrot 集中所选取的 c 值对应到 Julia 集时, 这个 Julia 集是连通的, 反之亦然。[2]

这里在 src 目录下提供了一个 python 代码 `man_julia.py`[3], 运行后可以得到一个窗口 (如图5所示), 点击左侧的 Mandelbrot 集的图像的某个点 c 的位置即可在右侧生成相应的 Julia 集的图像。可在根目录终端下执行命令 `make relation` 得到。

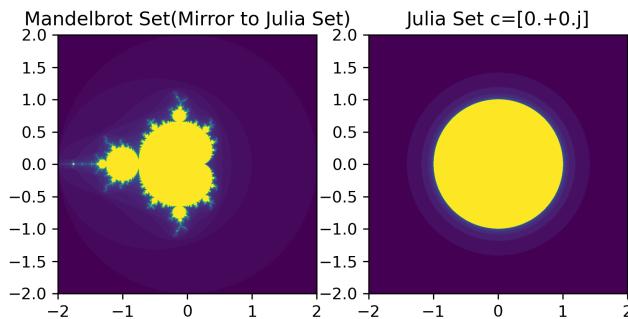


图 5: 代码经 ipython3 运行后窗口展示

5 总结与思考

通过以上简单的分析与探索，我对 Julia 集的性质与特征有了更深入的了解，并且通过 Python 语言仿照编写了程序对分析和探索的过程进行了可视化，使得这一过程更为清晰自然。于此同时，还对 Mandelbrot 集与 Julia 集的关系进行了浅显的分析与观察，而这一联系方法想必能够适用于其他的 $f(z)$ ，这将帮助我们寻找不同 $f(z)$ 下更能凸显图像性质的 c 值。

参考文献

- [1] Julia set. [EB/OL]. https://en.wikipedia.org/wiki/Julia_set.
- [2] Juan Carlos Ponce Campuzano. The julia set. [EB/OL]. https://complex-analysis.com/content/julia_set.html.
- [3] Re De. Python 绘制 mandelbrot set 与 julia set. [EB/OL]. <https://zhuanlan.zhihu.com/p/32788146>.
- [4] Eric W Weisstein. Julia set. [EB/OL]. From MathWorld—A Wolfram Web Resource.<https://mathworld.wolfram.com/JuliaSet.html>.
- [5] 陈少文. Mandelbrot set. [EB/OL]. <https://www.chenshaowen.com/blog/drawing-2d-fractal-graph-using-python.html>.