

混沌游戏与分形

李梓丰

南京大学物理学院 2022 级，南京 210046

论文是否发表：否

2023 年 4 月 9 日

摘要 我们都知道诸如牛顿迭代法， $Z_{n+1} = Z_n^2 + C$ 等非线性动力系统可以构造出多种分形结构，其中最著名的当属芒德布罗集和茱莉亚集。非线性现象导致的混沌和分形结构我们已经司空见惯。自然界中几乎没有线性变换产生分形结构的例子，因为他们的行为一定是可以预测的。然而如果将线性变换与随机性结合，却能得到不一样的结果，混沌游戏便是随机性产生分形最经典的例子。本文分析了混沌游戏中随机行为导致有序分形结构的根本原因，并得出一套可以解释并预测混沌游戏行为的理论，最后分析分形结构在物理学中的应用。

关键词 混沌游戏，分形

论文创新性

完整阐述了作者独立建立混沌游戏理论的过程，并分析分形的应用价值

引言

看似随机的混沌游戏却会出现分形图案，其实随机并不重要，重要的是背后的规则

1 混沌游戏的唯象研究

游戏内容阐述：设置有规则的一些点，例如单位圆周上的 3 个等分点作为目标点，标记为 1 到 3。在平面上任意处设置母点，随机生成 $[1 - 3]$ 的随机整数，若得到随机整数 n ，母点向第 n 号目标点前进二者距离的 β 倍 ($0 < \beta < 1$)，在此处标记并作为新的母点。随机操作 N 趋于无穷次。得到的图形会是一个“分形”即“Sierpinski triangle”¹。类似的，还可以构建出“Koch curve”等著名的图案。

混沌游戏可以由玩家设置的参数有：标记点数目 n ，和移动比例 β ，玩家也可以调整 n 个目标点的位置，不过这只会影响图案的宏观形状，其微观的自相似性不受影响。

1.1 用语声明

1. 分形等级

我们使用零级分形指代最原始的分形图案，并定义： $n+1$ 级分形是 n 级分形的一个最大子分形。由分形的自相似性，分形的大小关系具有相对性，如果说某个子块是相对 m 级分形的 n 级分形，那么它是一个 $(m+n)$ 级分形。本文中的分形等级，如果不加说明都默认是相对零级分形的等级。

2. 分形定位

我们使用一个有序数组来对分形进行定位：依旧以刚才的 Sierpinski triangle 为例，对这个 0 级分形的三个角依次编号为 1, 2, 3 并按此规律标记其内部所有的高级分形三角形的顶点。由分形图案的自相似性，

¹ 虽然看起来很像，但是我们还没有证明，在证明前都会使用引号标记

由归纳法定义有序数组: 定义 $[]$ 表示为 0 级分形, $[\alpha_1, \alpha_2, \dots, \alpha_n, \alpha_{n+1}]$ 定义为分形 $[\alpha_1, \alpha_2, \dots, \alpha_n]$ 靠近它标记点 α_{n+1} 的那个最大子分形。由我们的定义可知:

1. “相等定理”: 数组 “[]” 中数字个数, 与分形等级是相等的
2. “决定定理”: α_i 决定了在目标分形 $[\alpha_1, \alpha_2, \dots, \alpha_n]$ 在分形 $[\alpha_1, \alpha_2, \dots, \alpha_{i-1}]$ 中的哪一个子块内。

最后, 为了表示方便, 我们定义:

$$[\alpha_1, \alpha_2, \dots, \alpha_{n-1}, \alpha_n]^{-1} = [\alpha_n, \alpha_{n-1}, \dots, \alpha_2, \alpha_1]$$

3. 操作

母点按游戏规则随机移动并在新位置生成标记。这称为一次操作。

4. 示例:

如图-2, 图中完整的 Sierpinski triangle 是零级分形, 数组 [1,1] 定位图中红色边框的小 Sierpinski triangle, 数组 [2,1] 定位图中蓝色边框的小 Sierpinski triangle, 它们都是二级分形。数组 [1,2,2] 则定位图中粉红色边框的小 Sierpinski triangle。数组 $[1, 2, 2, \dots]$ 定位的所有分形, 都会落在这个粉色的 Sierpinski triangle 内

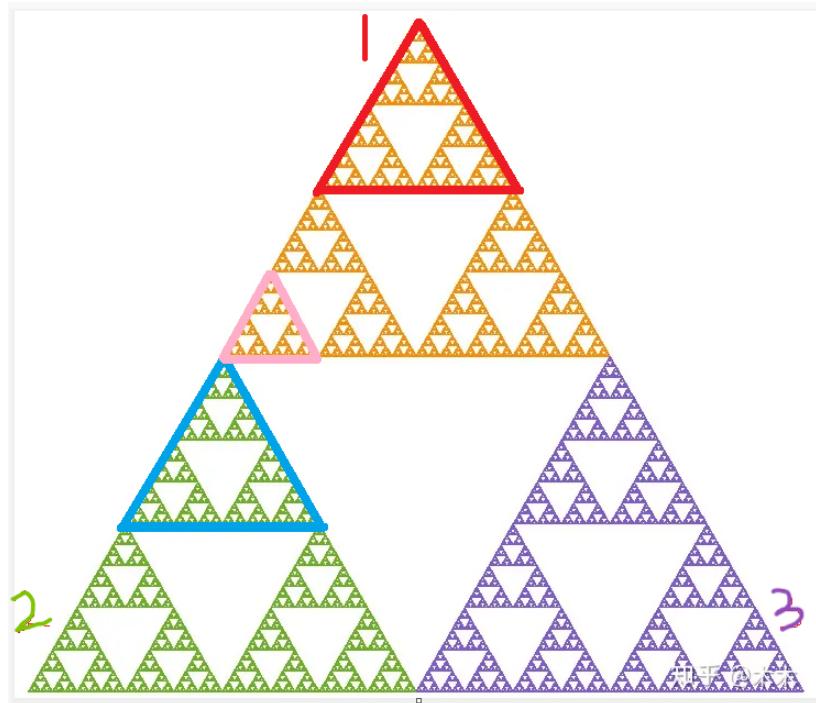


图 2: 分形等级与分形定位示例

1.2 混沌游戏的现象

(1). 增加随机到某个目标点的概率, 发现所有高阶分形相对自身对应编号的点的密度都增加相同比例, 如图-3

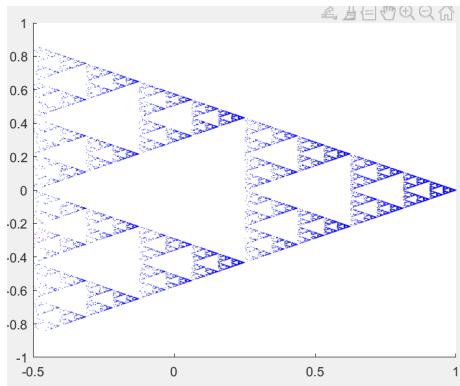


图 3: 混沌实验结果 1

(2). 混沌游戏的图案对于初始点的位置不敏感, 对于单位圆上的标记点, 初始点的位置设置为 $(0,0)$ 和 $(10^6, 10^6)$, 两次实验生成的图案相同。

(3). 操作 30 万次, 取第二十万到三十万次的结果, 与另一次独立实验的第 1 到第十万次的结果对比, 两次实验生成的图案相同。

(4). 将五十万次操作分解五十次从头开始的一万次操作。两次实验得到的图案相同。

1.3 基于引理的唯象研究

基于上述实验现象 (3)(4), 我们提出一个引理²: 混沌游戏的图案对于初始点的位置不敏感, 所有初始位置生成的图案是一样的 (实验操作次数足够多) 我们研究一个由三个母点组成的阵列, 基于这个引理, 我们提出阵列假说: 假设有无穷个这样的点阵, 每个点阵的三个母点最初分布在正三角形的三个顶点即设置的三个目标点上。这无穷个点阵之间的行为是无关的, 但是假设每个点阵内部所有点的行为都一致。

我们用矢量对移动的母点和三个标记点进行定位, 用单位圆周上的矢量 \vec{A}_i 定位标记点, 用 \vec{r}_n 定位第 n 次操作后的母点, 如图-4。

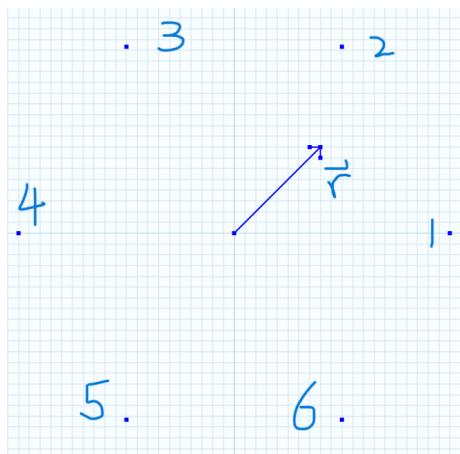


图 4: 矢量定位示意图

执行一次操作, 由游戏规则, 不妨假设随机到的数为 m , 新的母点的位置矢量为: $\vec{r}' = (1-\beta)\vec{r} + \beta\vec{A}_m$, 不难发现, 这个阵列的形状, 朝向和各个点之间的相对位置关系都没有改变。新的母点阵列, 其尺度变为原来的 $(1-\beta)$ 倍, 而整体产生一个 $\beta\vec{A}_m$ 的位移。因为母点阵列在操作后, 形状不变, 只是大

² 这在第二部分会给出证明

小各向同性变化并产生位移的性质，我们可以用点阵中心点的位置矢量来表征阵列的移动，并用一个数来表征阵列相对最开始时的大小。定义初始时，操作次数为 0，特征位置矢量为 $\vec{r}_0 = \vec{0}$ ；设第 n 次操作所得随机数为 i_n 由移动关系可知

$$\vec{r}_{n+1} = (1 - \beta) \vec{r}_n + \beta \vec{A}_{i_n} \quad (1)$$

再操作一次，不妨设此次随机到的数为 i_{n+1} ，可以得到：

$$\vec{r}_{n+2} = (1 - \beta) \vec{r}_{n+1} + \beta \vec{A}_{i_{n+1}} = (1 - \beta)^2 \vec{r}_n + (1 - \beta)\beta \vec{A}_{i_n} + \beta \vec{A}_{i_{n+1}} \quad (2)$$

由此看出，阵列经过 n 次操作后，大小变为原来的 $(1 - \beta)^n$ 。若这 n 次操作得到的随机数序列为：

$$(i_1, i_2, \dots, i_n) \quad (3)$$

那么 n 次操作后，阵列的位置矢量为：

$$\vec{r}_n = \beta [(1 - \beta)^{n-1} \vec{A}_{i_1} + (1 - \beta)^{n-2} \vec{A}_{i_2} + \dots + \vec{A}_{i_n}] \quad (4)$$

仔细分析 (4) 式，将它变形为：

$$\begin{aligned} \vec{r}_n &= \beta [\vec{A}_{i_n} + (1 - \beta) \vec{A}_{i_{n-1}} + (1 - \beta)^2 \vec{A}_{i_{n-2}} + \dots + (1 - \beta)^{n-1} \vec{A}_{i_1}] \\ \vec{r}_n &= \beta \vec{A}_{i_n} + \beta (1 - \beta) \vec{A}_{i_{n-1}} + \beta (1 - \beta)^2 \vec{A}_{i_{n-2}} + \dots + \beta (1 - \beta)^{n-1} \vec{A}_{i_1} \end{aligned} \quad (5)$$

上式中，后一项的模是前一项的 $(1 - \beta)$ 倍。于是得到：阵列尺度相对于初始时变换的比例 $\frac{L_n}{L_0}$ ，与阵列位移矢量增量的模相对初始时的比例 $\frac{|\vec{r}_{n+1} - \vec{r}_n|}{|\vec{r}_1 - \vec{r}_0|}$ 恰好相等，均为 $(1 - \beta)^n$ 。我们不妨将式 (3) 写开来看：

$$\begin{aligned} \vec{f}_{n,0} &= \vec{0} \\ \vec{f}_{n,1} &= \beta \vec{A}_{i_n} \\ \vec{f}_{n,2} &= \beta \vec{A}_{i_n} + (1 - \beta) \beta \vec{A}_{i_{n-1}} \\ &\vdots \\ \vec{f}_{n,j} &= \vec{f}_{j-1} + (1 - \beta)^{j-1} \beta \vec{A}_{i_{n+1-j}} \\ &\vdots \\ \vec{f}_{n,n} &= \beta \vec{A}_{i_n} + \beta (1 - \beta) \vec{A}_{i_{n-1}} + \dots + \beta (1 - \beta)^{n-1} \vec{A}_{i_1} \end{aligned} \quad (6)$$

由 (6) 可以看出， $\forall j \in \{1, 2, 3, \dots, n\}$ ， $\vec{f}_{n,j}$ 在 $\vec{f}_{n,j-1}$ 的基础上，移动了与 $(j-1)$ 级分形尺度成比例的移动距离。

$$\begin{aligned} \vec{f}_{n,j-1} &= \beta \vec{A}_{i_n} + \beta (1 - \beta) \vec{A}_{i_{n-1}} + \dots + \beta (1 - \beta)^{j-2} \vec{A}_{i_{n+2-j}} \\ \vec{f}_{n,j} &= \beta \vec{A}_{i_n} + \beta (1 - \beta) \vec{A}_{i_{n-1}} + \dots + \beta (1 - \beta)^{j-2} \vec{A}_{i_{n+2-j}} + \beta (1 - \beta)^{j-1} \vec{A}_{i_{n+1-j}} \end{aligned}$$

下面我们完成基于阵列假说的证明：

因为阵列同步操作不改变阵列的形状，只产生尺度变换和平移，所以 $\forall n \in \mathbb{N}$ 次操作后阵列形状和朝向不变， $\vec{f}_{n,j}$ 始终指向阵列中心。注意到操作的“压缩”性质，即每次操作阵列的大小和相对位移矢量都变为上一次的 $(1 - \beta)$ 倍 $[0 < (1 - \beta) < 1]$ 。

因为 $\overrightarrow{f_{n,j}}$ 在 $\overrightarrow{f_{n,j-1}}$ 的基础上, 移动了与 $(j-1)$ 级“分形”尺度成比例的移动距离。由(6)式, 可以做一次坐标变换: 把 $\overrightarrow{f_{n,j}}$ 指向的位置视为新的坐标原点, 并将坐标轴刻度放大 $\frac{1}{(1-\beta)^j}$ 倍, 那么在变换后的坐标系下观察 $\overrightarrow{f_{n,j}}$ 之后的矢量的移动, 相比于矢量 $\overrightarrow{f_{n,0}}$ 到 $\overrightarrow{f_{n,j}}$ 的移动, 除了随机数可能不同之外, 其他行为都是一样的。如果简单修改游戏规则: 每次操作每个母点都分裂成 n 个分别完成朝向每个目标点的操作而不是根据随机数来对某一个目标点操作, 那么尺度变换后会看到完全一样的行为。所以这个行为是随机自相似的。

假设 $\overrightarrow{f_{n,j}}$ 指向 j 级“分形”的中心, 那么由上述结论, $\overrightarrow{f_{n,j+1}}$ 指向 $(j+1)$ 级“分形”的中心, 相邻等级“分形”的相对位置由 β 和 \vec{A}_i 确定, 因为 $\overrightarrow{f_{n,0}} = \overrightarrow{0}$ 即指向零级“分形”的中心, 那么由数学归纳法可知, $\forall j \in \{1, 2, 3, \dots, n\}$ $\overrightarrow{f_{n,j}}$ 指向 j 级“分形”的中心, 所以 $\forall n \in \mathbb{N}$ $\overrightarrow{f_{n,n}}$ 指向 n 级“分形”的中心, 即 \vec{r}_n 指向 n 级“分形”的中心。在操作次数 $N \rightarrow +\infty$ 时, 产生无限的随机自相似性。

由操作的压缩性质, 我们将 $\vec{r}_n(\overrightarrow{f_{n,n}})$ 中的每一项取模相加, 可以得到一个等比数列, 不难验证其绝对收敛性。所以 \vec{r}_n 指向平面上有限远处确定的一个点, 那么它到底指向平面上的哪个点? 我们再看(3)式, 并对它进行变换:

$$\begin{aligned} & \left(i_1, i_2, \dots, i_{j-1}, i_j \right) \\ & \rightarrow \left([i_1, i_2, \dots, i_{n-1}], i_n \right) \\ & \rightarrow \left(i_n, [i_{n-1}, \dots, i_2, i_1] \right)^{-1} \end{aligned} \quad (7)$$

与(6)式对比, 发现这 n 次操作生成的随机序列(3)的逆序列: 就是操作 n 次时这个阵列的“分形”定位数组。

$$\begin{aligned} [\alpha_1, \alpha_2, \dots, \alpha_n] &= (i_n, i_{n-1}, \dots, i_2, i_1) \\ \alpha_i &= i_{n+1-i} \end{aligned} \quad (8)$$

\vec{r}_n 指向这个 n 级“分形”的中心。由压缩性质, 在 $n \rightarrow +\infty$ 阵列与这个“分形”中心重合。

由上文可知每个阵列最终都会收敛到一个确定的“分形” $[\alpha_1, \alpha_2, \dots, \alpha_n]$ 如果承认阵列假说的成立, 即有 $M \rightarrow +\infty$ 个初始的阵列各自独立地进行无限次操作, 那么根据“决定定理”, 高级子分形 H 在较低级子分形 L 的子块中的位置由分形定位数组 $[\alpha_1, \alpha_2, \dots, \alpha_n]$ 中第 L 位的 α_L 决定。由游戏规则, 数组中的每一个 α 按照预定概率取值, 所以在有无限多阵列的时候, 这些阵列的操作结果将会按照设定密度分布, 并且遍历整体“分形”的每一个“子分形”, 并且使得 $N \rightarrow +\infty$ 时的无限随机自相似性退化为无限的设定概率分布的自相似性。至此我们完成了基于阵列假说的证明。

阵列假说除了可以解释实验现象(1)还可以得到更多结论:

1. 由式(1)和游戏规则可以得到, 操作的标度变换系数为 $(1-\beta)$, 而自身相似图形的数目为 n , 由豪斯道夫维数的定义, 我们不难知道, 进行一次比例为 $\frac{1}{1-\beta}$ 的放大, 需要 n 个与自己一样的图形, 所以混沌游戏生成图案的分形维数为:

$$D = \frac{\ln n}{\ln \frac{1}{1-\beta}} \quad (9)$$

2. 实验发现, 固定一个 n 时, 只有取某些特定的 β 生成的图像才最好看, 而某些 β 取值使得式(9)得出大于等于 2 或者小于等于 1 的结果, 我们不去讨论这些情况。表 1 给出了圆周分布下使得图案最好看标记点数目 n 以及 β 的取值:

n	β	对应分形
3	$\frac{1}{2}$	谢尔宾斯基三角
4	> 0.5	康托尔四分集
5	$\frac{\sqrt{5}-1}{2}$	五角星（康托尔五分集）
6	$\frac{2}{3}$	科赫雪花

表 1: 一些图案的 n 和 β 取值

使用阵列假说可以得到使得生成的分形图最大最清晰的 n 和 β 定量的关系式，并且在已知分形维数 D 大于 1 小于 2 时，阵列假说可以预测给定标记点（ n 和具体位置）和 β 时，生成的图案形状。如图 5-8 给出了最合适的关系下阵列假说预测和真实输出的图片：

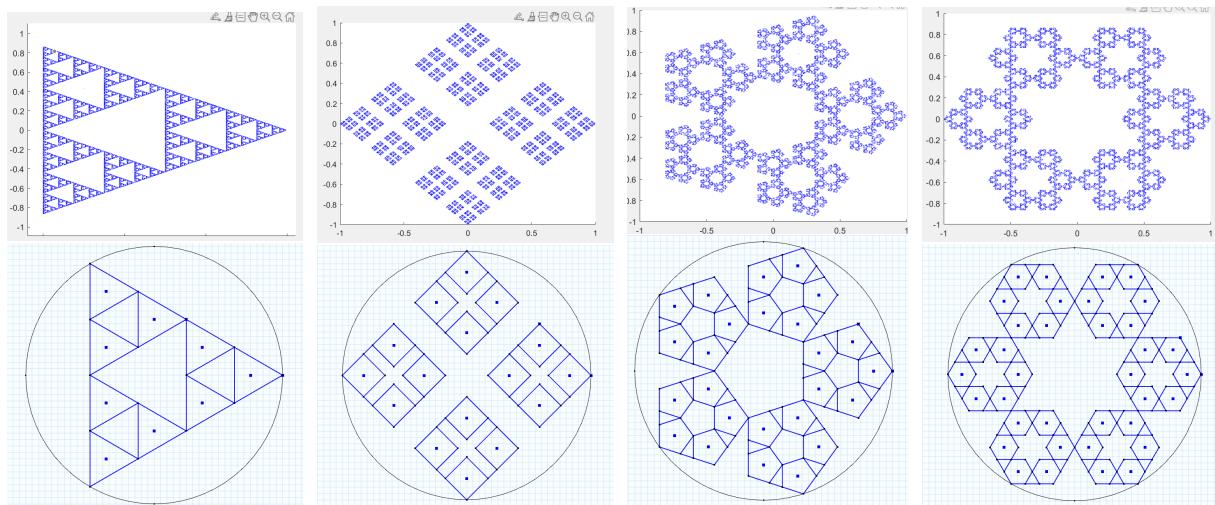


图 5

图 6

图 7

图 8

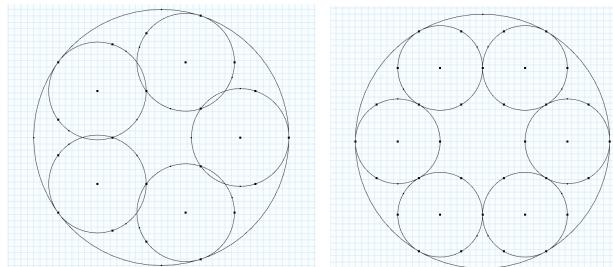


图 9

图 10

上图给出 n 为 5 和 6 的情况，并且不难证明：当 n 为奇数时，相邻两个子分形所在的小圆相交，而 n 为偶数时则相切。由几何关系可以得到：

$$\begin{aligned} \beta &= \frac{1}{1 + \sin \frac{1}{n}} && \text{如果 } n \text{ 为偶数} \\ \beta &= \frac{1}{1 + 2 \sin \frac{1}{2n}} && \text{如果 } n \text{ 为奇数} \end{aligned} \quad (10)$$

3. 在原先标记点的基础上，将圆形也设定为标记点，这样得到的图形会更“充实”，不过依然可以用阵列假说预测。如图 11-14

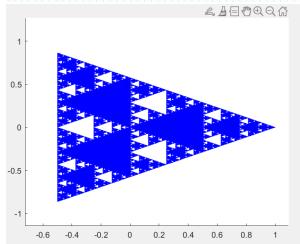
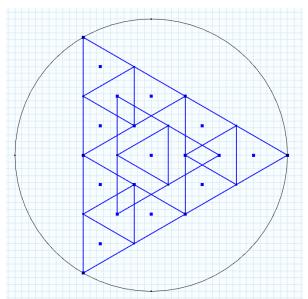


图 11

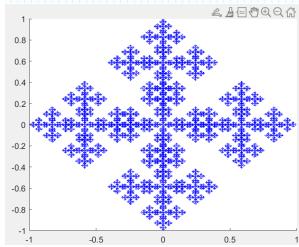
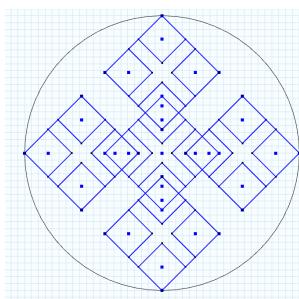


图 12

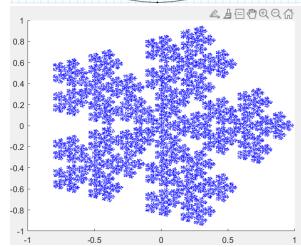
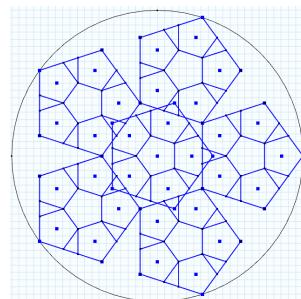


图 13

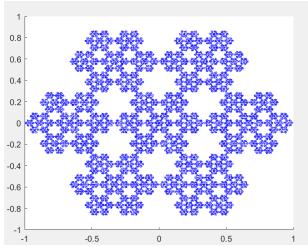
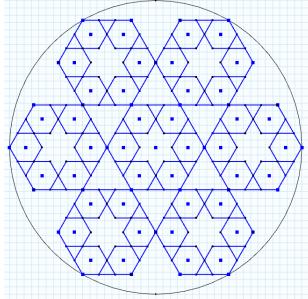


图 14

4. 我们发现大部分具有三次及以上对称轴的分形，例如谢尔宾斯基地毯，都可以用混沌游戏进行构造。但是此时标记点则不能简单地分布在单位圆上，不过上述阵列操作的结论依然成立。

假设要构造的分形维数为 $D = \frac{\ln a}{\ln b}$ ，通过式 (9) 可以确定这个分形一些最基本的信息：

$$n = a \quad \beta = 1 - \frac{1}{b}$$

标记点分布可以用上文的阵列假说以及对称性确定。这些标记点满足的条件是：恰好分布在所有标记点上母点组成的阵列，经过 2 次相同行为的操作后得到的新阵列 1 和新阵列 2，均分布在相应标记点的对应位置，并且结构与初始完全相似。

图 15-17 给出了三种不同于前文圆周上混沌游戏分形图形：谢尔宾斯基地毯，box 分形和一种 cantor 分形的标记点分布，预测和真实结构。

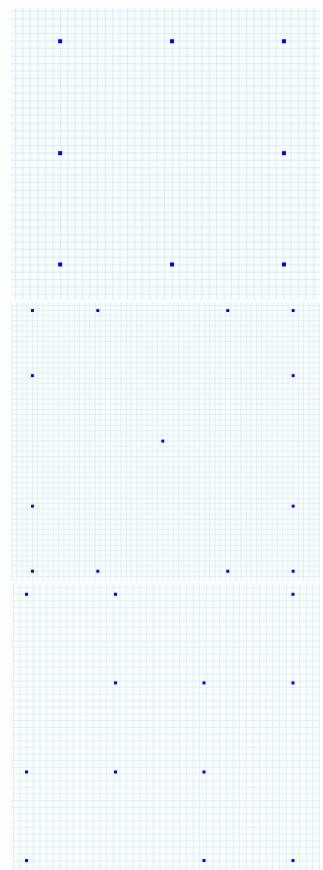


图 15: 生成点阵

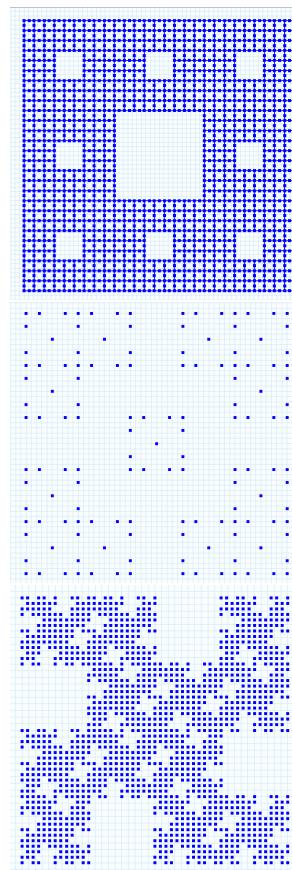


图 16: 阵列假说的预测

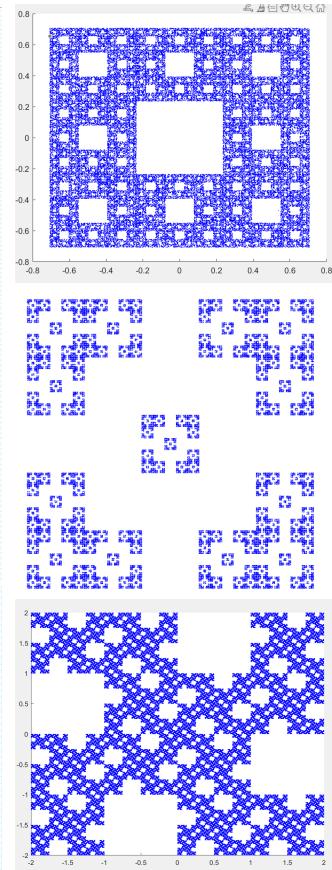


图 17: 最终实验结果

上述阵列假说虽然可以解释一部分实验现象，并且可以预测分形图形的构造，但是有一个致命的问题：它无法解释分形位置与初始选点无关的情况，因为在阵列假说中，阵列的初始位置是生成的图案的中心位置，而非标记点的中心位置。为了解释完整的实验现象，我们需要从最基本的公式：式(1)重新开始。

2 探究混沌游戏的本质

2.1 前文中引理的证明

现在我们不加任何假设，完整地研究混沌游戏。对于初始位置在任意 \vec{r}_0 处的初始母点，在整个游戏中依然有

$$\vec{r}_{n+1} = (1-\beta)\vec{r}_n + \beta\vec{A}_{i_n} \quad (11)$$

成立，其中 i_n 是第 n 次操作中得到的随机数，由此自然可得

$$\begin{aligned} \vec{r}_n &= \beta[(1-\beta)^{n-1}\vec{A}_{i_1} + (1-\beta)^{n-2}\vec{A}_{i_2} + \cdots + \vec{A}_{i_n}] + (1-\beta)^n\vec{r}_0 \\ \vec{r}_{n+1} &= \beta[(1-\beta)^n\vec{A}_{i_1} + (1-\beta)^{n-1}\vec{A}_{i_2} + \cdots + (1-\beta)\vec{A}_{i_n} + \vec{A}_{i_{n+1}}] + (1-\beta)^{n+1}\vec{r}_0 \end{aligned} \quad (12)$$

发现标度变换关系依然成立， \vec{r}_{n+1} 依然是将 \vec{r}_n 进行大小变换并位移。

令 $\vec{f}_n = \beta[(1-\beta)^{n-1}\vec{A}_{i_1} + (1-\beta)^{n-2}\vec{A}_{i_2} + \cdots + \vec{A}_{i_n}]$ 由前文结论可知， \vec{f}_n 指向由(4)给出的 n 级分形的中心位置

$$\beta[(1-\beta)^n\vec{A}_{i_1} + (1-\beta)^{n-1}\vec{A}_{i_2} + \cdots + (1-\beta)\vec{A}_{i_n} + \vec{A}_{i_{n+1}}]$$

而有相对偏移量

$$(1 - \beta)^n \vec{r}_0$$

注意到 \vec{r}_n 指向的分形等级为 n , 这个 n 级分形相对零级分形缩小的尺度, 与 \vec{r}_0 的偏移量相对 \vec{r}_0 的偏移量缩小尺度相同, 均为 $(1 - \beta)^n$ 由此可知: 操作次数越多, \vec{r}_n 描绘的分形级次越高, 细节越多。因为每次都带有与所描绘的分形等级成固定比例的偏移量, 因此自相似性依然不变。再由分形定位式(6)可知, 因为数组 $[\alpha_1, \alpha_2, \dots, \alpha_n]$ 中的数 α 取值按照预设的概率分布, 所以这个母点将会在所有等级的分形中按照设定概率, 在设定标记点附近分布。由此可完成证明: 混沌游戏给出的图案确实是相对每个标记点, 按照预设概率进行密度分布的分形图形。而能够产生无限自相似结构的根本原因, 在于变换是“压缩”的!

2.2 IFS 迭代函数系统

不难发现, 混沌游戏的规则可以这样进行推广。用列向量 $(x_n, y_n)^T$ 描述操作 n 次后母点的位置, 而移动规则变为:

$$(x_{n+1}, y_{n+1})^T = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}_{i_n} (x_n, y_n)^T + (b_1, b_2)_{i_n}^T \quad (13)$$

或者写作

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = F_{i_n} \begin{pmatrix} x_n \\ y_n \end{pmatrix} + f_{i_n} \quad (14)$$

其中 F_{i_n} 是 2×2 矩阵, f_{i_n} 是 2×1 列向量。这样的操作称为仿射变换, 变换规则数目和变换具体形式由外界给定。不难得到, 前文中混沌游戏对应的仿射变换为:

$$\begin{aligned} F_{i_n} &= \begin{pmatrix} 1 - \beta & 0 \\ 0 & 1 - \beta \end{pmatrix} \\ f_{i_n} &= \beta \vec{A}_{i_n} \\ &= \begin{pmatrix} X_{i_n} \\ Y_{i_n} \end{pmatrix} \end{aligned} \quad (15)$$

其中 (X_{i_n}, Y_{i_n}) 是标记点 A_{i_n} 的坐标。

这样的一组变换成为迭代函数系统 (IFS), 通过与上文类似的方法, 可以证明 IFS 变换下的图形是自相似的, 同样是与初始点位置无关的分形结构。它与混沌游戏有相似的要求: 一个阵列中所有点在经过相同的操作后, 这个阵列的改变只能是尺度的缩小, 平面内的旋转和平面内的平移这三种变换的线性组合。而 IFS 生成的图形, 则可以用上文阵列假说推广得到的“拼贴定理”进行预测。IFS 可以给出更加多样化的图案, 图-18 是用 IFS 绘制的巴恩斯利蕨的图案。

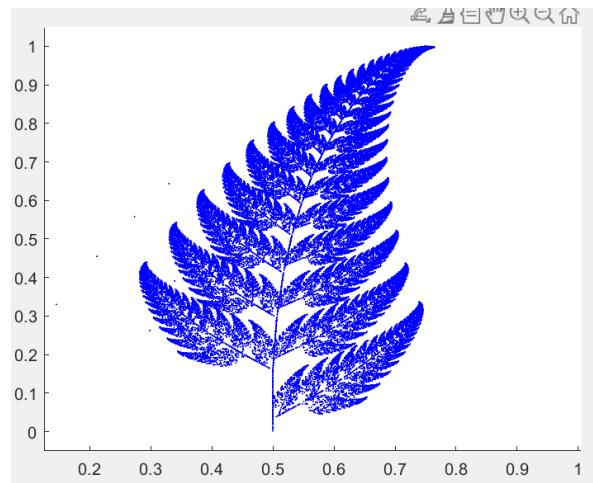


图 18: IFS 巴恩斯利蕨

通过用恒等变换替换这个程序中的某一个或者某几个仿射操作，可以发现，四个不同的仿射变换分别绘制是：定位，绘制茎秆，并绘制所有茎秆左侧叶子，绘制所有茎秆右侧叶子。通过搭建完成独立任务的仿射变换并“拼贴”在一起，即可构造更丰富的分形图。

2.3 进一步推广到动力系统

如果再进一步推广映射范围，就可以得到动力系统。绘制出整个复平面上使得映射函数：

$$Z_{n+1} = F(Z_n)$$

满足不发散条件的所有点，可以得到比 IFS 具有更加多样性的分形结构。图 19-21 是我们用 matlab 绘制的牛顿分形，芒德布罗集，和 logistic map，这些动力系统分别对应的映射是：

$Z_{n+1} = Z_n - \frac{f(z)}{f'(z)}$	牛顿分形， $f(z)$ 是多项式函数
$Z_{n+1} = Z_n^2 + C$	芒德布罗集
$Z_{n+1} = \mu Z_n(1 - Z_n)$	logistic map

(16)

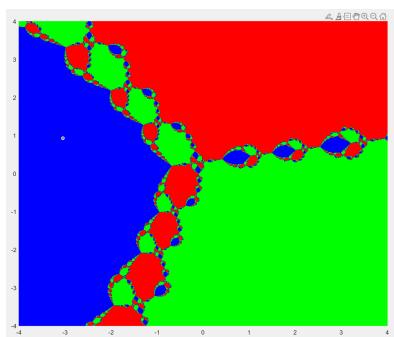


图 19: 三次牛顿分形

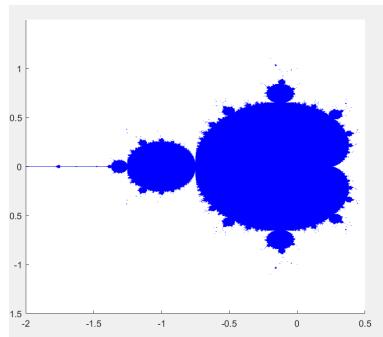


图 20: 芒德布罗集

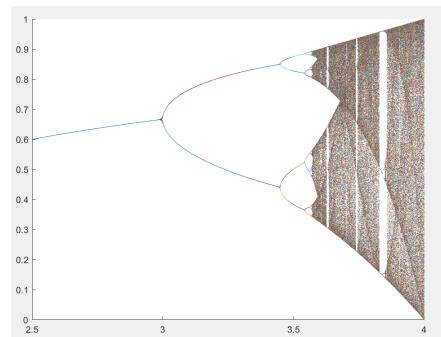


图 21: logistic map

3 分形的应用

这里我们简略介绍分形在物理学中的应用价值:

3.1 分形与优化设计

2018 年 comsol 用户年会上的一个案例: 3D 打印散热器设计的仿真与优化, 从定义的罚热流函数极小值的方向出发, 得到的最终结果竟然也是一个分形: 随着迭代次数的增加, 最终的结果类似于 DLA 模型生长的样子, 如图-22.

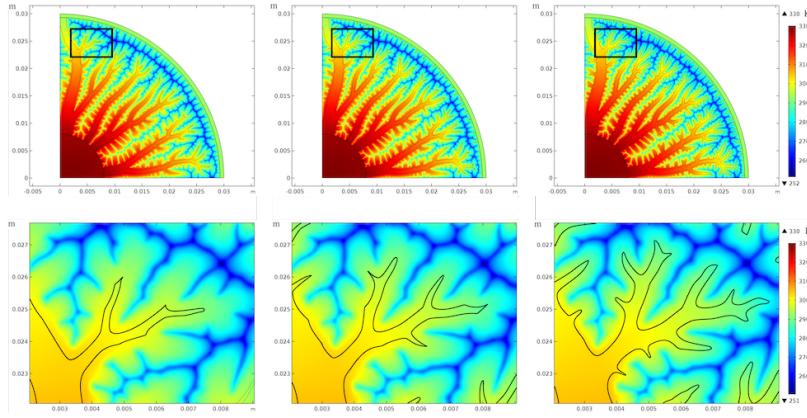


图 22: 拓扑优化函数竟然出现类似分形结构

这其实不难理解, 上例优化的最终结果是使得结构传热效率最大, 那么不难发现: 最佳拓扑设计结果中, 每个点处的热流密度一定是相等的。因为如果不相等, 总是可以在低热流密度处的点处再次优化, 直到整体函数值达到最小。此时这个图形就具有了一定程度的自相似性。而类似的构型在树木的分岔结构, 粘性指进等现象中常见。

另一个分形与设计相关的例子是分形天线, 由 maxwell 方程组:

$$\begin{aligned} \nabla \cdot \mathbf{E} &= \frac{\rho}{\epsilon_0} \\ \nabla \cdot \mathbf{B} &= 0 \\ \nabla \times \mathbf{E} &= -\frac{\partial \mathbf{B}}{\partial t} \\ \nabla \times \mathbf{B} &= \mu_0 \mathbf{j} + \mu_0 \epsilon_0 \frac{\partial \mathbf{E}}{\partial t} \end{aligned} \tag{17}$$

可以得到: $\forall \lambda \in \mathbb{R}^*$ 方程组经过这个变换不变:

$$\begin{aligned} r &\rightarrow \lambda r \\ \omega &\rightarrow \lambda \omega \end{aligned} \tag{18}$$

与只能用于单频谐振的半波偶极天线不同, 使用混沌游戏构造的分形天线不需要额外的匹配网络即可将天线输入阻抗调整为 50 欧姆的参考特性阻抗, 从而获得高阶谐振并且由于自相似性, 分形天线具有一系列功率峰值的工作频域。与混沌游戏构造出的规则分形不同, 使用随机分形构造的分形天线则适合连续频段, 具体参考文献 [3] 和 [4]。

下图给出了分形天线与普通天线的对比:

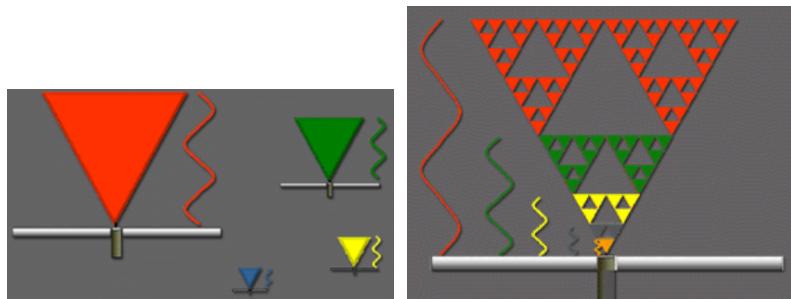


图 23: 普通天线

图 24: 分形天线

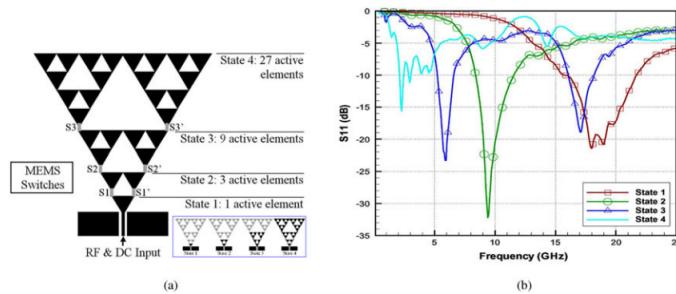


图 25: 分形天线的性能

3.2 分形与物理现象

物理现象中经典的分形案例有电解时的分形沉积，晶体生长，闪电的分形形状，粘性指进等，如图是我们用 Comsol 制作的雪花生长以及粘性指进的仿真结果：

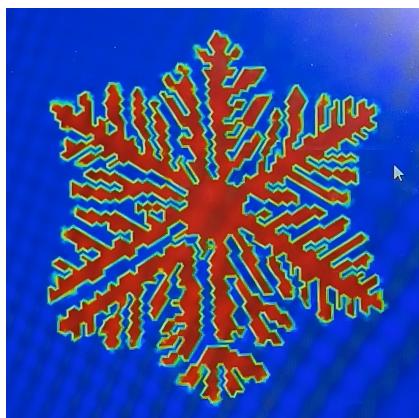


图 26: 雪花生长

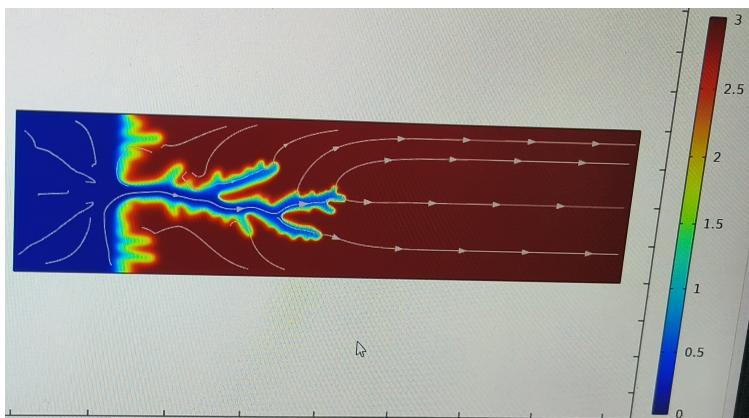


图 27: 粘性指进

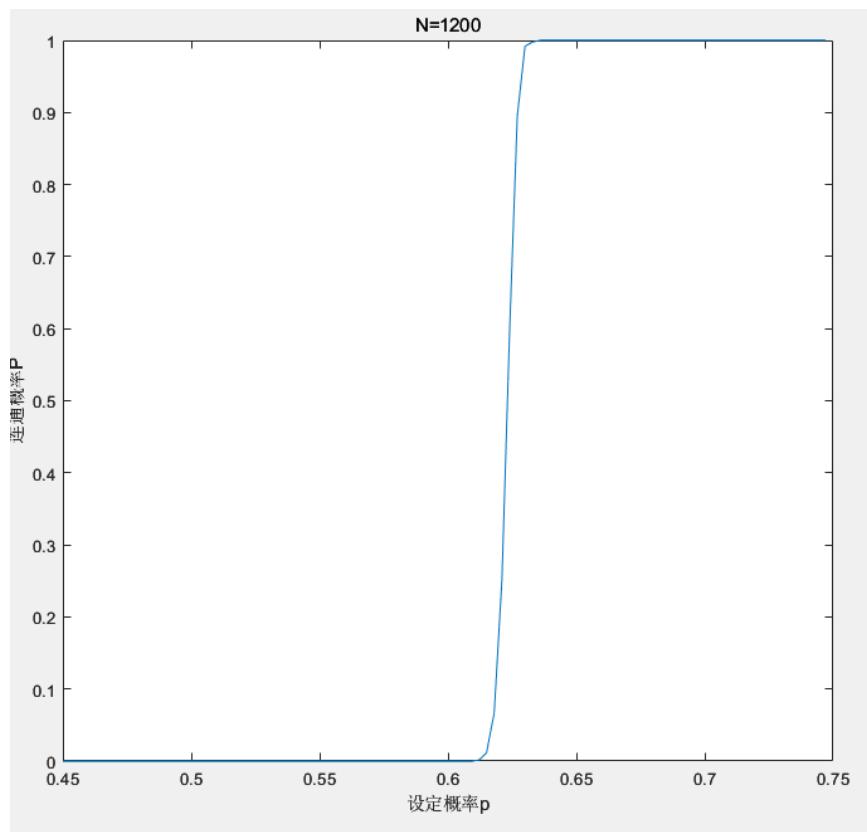
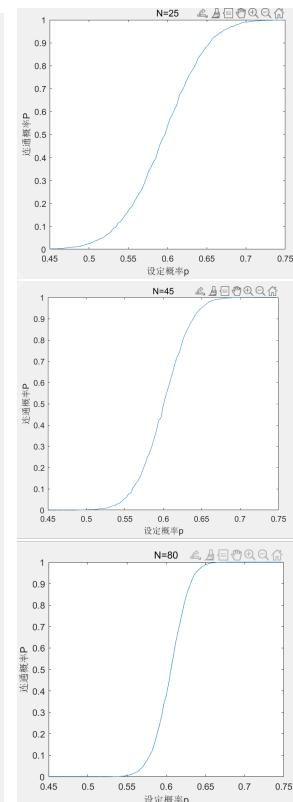
这些案例共同的特点是生长率与该处场的梯度的幂律成比例，使得生长过程中存在可以被放大的不稳定性，他们都可以用 DLA 模型来进行研究：相关内容可以参考文献 [5]。

3.3 分形与临界现象

另一个在物理学中有深刻意义的是渝渗模型，它在多孔介质流动，凝聚态物理，以及临界现象研究等领域应用广泛。

让我们假设一个 $N \times N$ 的网格，其中顶面与底面有电势差。网格的每个单元都有连通（1）和断开（0）

两种状态，每个格点有 p 的概率处于状态 1。如果一个格点与它的一个相邻格点都处于状态 1，那么他们之间是连通的。我们的问题是网络整体连通性如何，即顶面是否与底面连通？连通概率 P 与设定的 p 有什么关系？如图为我们编写的 matlab 程序运行的结果；我们发现，在 $p=pc$ 的临界位置，网络连通的概率发生了突变，从几乎无法连通突变为几乎全部连通。在 N 逐渐增大的过程中，连通概率的突变性越来越明显，数学上可以证明在网格大小趋于无限时，概率突变是瞬间完成的

图 28: 渗透模型, $N=1200$ 图 29: 从上到下 N 分别为: 25, 45, 80

数学上可以使用“重整化群”的方法研究这种自相似的行为。将四个小个格子合并成一个较大的格子，合并规则如图：

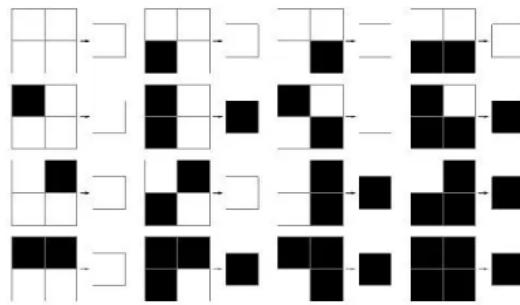


图 30: 重整化群合并规则

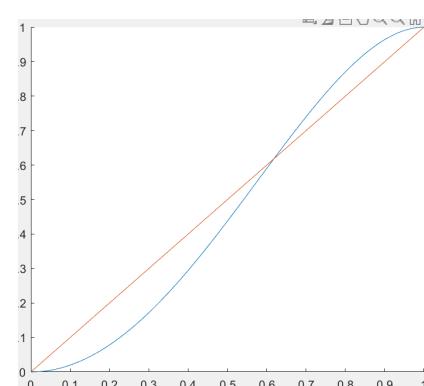


图 31: 求出不动点

那么我们看到，如果小格子处于状态 (1) 的概率为 p ，那么合并得到的大格子处于状态 (1) 的概

率为:

$$\begin{aligned} F(p) &= p^4 + 4p^3(1-p) + 2p^2(1-p)^2 \\ &= 2p^2 - p^4 \end{aligned} \quad (19)$$

由于分形的性质，在相变临界位置应该表现出自相似性，求出上式的不动点：程序运行的结果显示的确在 $p = p_c$ 处系统的性质发生突变。

$$2p^2 - p^4 = p \Rightarrow p_c \approx 0.618$$

如果换一种可视化方法，使用不同的颜色给按照上文定义的连通区域染色，那么出现的结构将会是分形结构。这实际上是一种二级相变，类似的结构在 ising 模型，微观导电模型，相变等领域中常见且应用广泛。下图给出了处于相变临界处的染色渝渗模型和 ising 模型的对比图：

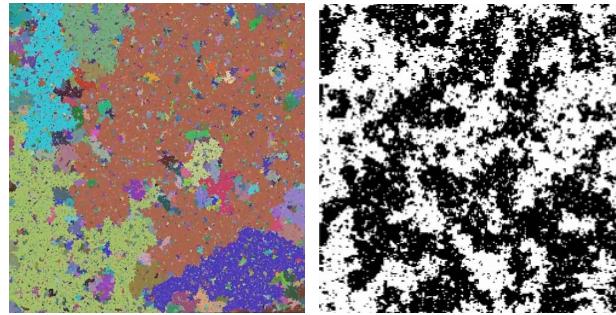


图 32: 染色渝渗模型

图 33: ising 模型

3.4 更多关于重整化群

暂且不提重整化群在 ising 模型，导电渝渗，范德瓦尔斯相变等中近似求解临界点的应用，重整化群对于分形的处理还可以进一步推广：上文提到的不动点，广义上说是一阶不动点，即：

$$F(Z) = Z$$

那么自然引出高阶不动点的定义：

$$F^n(Z) = Z \quad (20)$$

如果映射 $F(x)$ 是多项式函数，例如

$$F(Z) = Z^2 + C$$

那么根据代数基本定理， $F^n(Z) = Z$ 在复平面上一定是有解的，而这一组方程在复平面上所有的解，自然形成茱莉亚集。下图给出了我们用这种方法求而非传统方法求出的茱莉亚集：

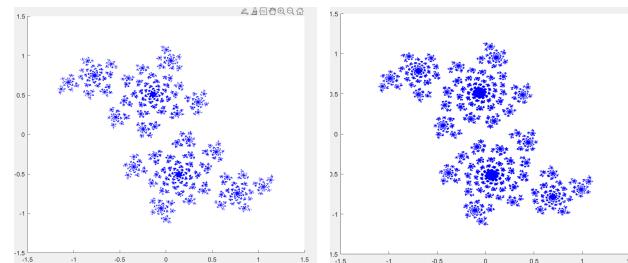


图 34: $C=0.1+0.65i$

图 35: $C=0.2+0.58i$

类似的高阶不动点的结构在芒德布罗集和 logistic 混沌映射中表现得特别明显。

4 结语

对于看似充满随机性的混沌游戏，以及它生成的一系列神秘的图形，我们使用数学工具证明它的分形性质，并给出预测和构造混沌游戏分形的理论方法。最后将其推广到 IFS 系统并发现内部最本质的结构：压缩映射。这种生成图形的方式在计算机图形学，尤其是图像压缩算法以及极限编程中应用广泛。

在物理学的世界中，混沌的行为才是最普遍的，但是其中不乏奇妙的原理。混沌是时间的分形，分形是空间的混沌。倍周期分叉这一通向混沌最普遍的方式，如 logistic 映射，与 ising 模型，渝渗导电模型竟然都联系着芒德布罗集，茱莉亚集，和重整化群等数学。我们正看到，分形从数学家们的玩具，正在逐步变成物理学家和工程师手中的工具。

5 代码附录

下面是我们编写的混沌游戏，IFS 巴恩斯利蕨，基于混沌游戏构造 box 分形，牛顿分形以及渝渗模型的 matlab 代码：

Listing 1: 圆周上的混沌游戏

```
function fractal()
N=3; %%%%玩家设置标记点数目;
if (1 == mod(N,2))
    beta = 1/(1+2*sin(pi/(2*N)));
else
    beta = 1/(1+sin(pi/N));
end
Num=2000; %操作次数
hold on;
A=zeros(2,N);
B=zeros(2,Num);
r=[0;0]; %初始位置(x;y);
for s=1:N
    A(:,s) = [cos(2*pi*s/N);sin((2*pi*s/N))];
end

for k=1:Num
    p = randi([1,N]);
    r = (1-beta)*r + beta*A(:,p);
    B(:,k)=r;
end
scatter(B(1,:),B(2,:),10,".", "blue");
```

Listing 2: IFS 巴恩斯利蕨

```
function ifs_4()
%巴恩斯利蕨
p1=0.85;
p2=0.86;
p3=0.93;
%p4=1;
r=[0;0];
A1=[0.85,0.04;-0.04,0.85];a1=[0.075;0.180];
A2=[0.0,0.0;0.0,0.16];a2=[0.5;0.0];
```

```

A3=[0.2,-0.26;0.23,0.22];a3=[0.4;0.045];
A4=[-0.15,0.28;0.24,0.24];a4=[0.575;-0.086];
Num = 10000;
X=zeros(2,Num);
for i=1:Num
    s=rand;
    if (s<p1)
        r=A1*s+a1;
    elseif(s<p2)
        r=A2*s+a2;
    elseif(s<p3)
        r=A3*s+a3;
    else
        r=A4*s+a4;
    end
    X(:,i)=r;
end
scatter(X(1,:),X(2,:),0.4,'.', 'blue')

```

Listing 3: box 分形

```

function fractal_13()

%%%%box分形
beta=4/5;
Num=70000;
hold on;
B=zeros(2,Num);
r=[0;0]; %初始位置(x;y);

A(:,1)=[2.5,2.5];
A(:,2)=[-2.5,2.5];
A(:,3)=[-2.5,-2.5];
A(:,4)=[2.5,-2.5];
A(:,5)=[2.5;1.25];
A(:,6)=[2.5;-1.25];
A(:,7)=[-2.5;1.25];
A(:,8)=[-2.5;-1.25];
A(:,9)=[1.25;2.5];
A(:,10)=[-1.25;2.5];
A(:,11)=[-1.25;-2.5];
A(:,12)=[1.25;-2.5];
A(:,13)=[0;0];

for k=1:Num
    p = randi([1,13]);
    r = (1-beta)*r + beta*A(:,p);
    B(:,k)=r;
end
scatter(B(1,:),B(2,:),0.5,".", "blue");

```

Listing 4: 牛顿分形-f(z) 是三次多项式

```
function Nfractal_3(z1,z2,z3)
```

```
% z^3 - (z1+z2+z3)*z^2 + (z1*z2+z1*z3+z2*z3)*z - z1*z2*z3 = 0;
% z^3 + P*z^2 + Q*z + R = 0;
% Zn+1 = Zn -f(z)/f'(z) ;
% M = 10000; %每个点迭代次数;

delta = 0.4*1e-2; %精细程度
a=-4:delta:4;
b=-4:delta:4;
nx=8/delta +1 ; %
ny=8/delta +1 ; %

s_r=0;
s_g=0;
s_b=0;
s_d=0;

for rx = 1:nx
    for ry = 1:ny
        q = calculate_N3(a(1,rx),b(1,ry),z1,z2,z3);
        switch q
            case 1
                s_r = s_r +1;
                Arx(1,s_r)=a(1,rx);
                Ary(1,s_r)=b(1,ry);
            case 2
                s_g = s_g +1;
                Agx(1,s_g)=a(1,rx);
                Agy(1,s_g)=b(1,ry);
            case 3
                s_b = s_b +1;
                Abx(1,s_b)=a(1,rx);
                Aby(1,s_b)=b(1,ry);
            case 0
                s_d = s_d +1;
                Adax(1,s_d)=a(1,rx);
                Aday(1,s_d)=b(1,ry);
        end
    end
end

hold on;
scatter(Arx,Ary,0.01,".", "red");
scatter(Agx,Agy,0.01,".", "green");
scatter(Abx,Aby,0.01,".", "blue");
scatter(Adax,Aday,0.01,".", "black");

function q = calculate_N3(a,b,z1,z2,z3)
% z^3 - (z1+z2+z3)*z^2 + (z1*z2+z1*z3+z2*z3)*z - z1*z2*z3 = 0;
% z^3 + P*z^2 + Q*z + R = 0;
% Zn+1 = Zn -f(z)/f'(z) ;

z=a+1i*b;
```

```

P=-(z1+z2+z3);
Q=z1*z2+z1*z3+z2*z3;
R=-z1*z2*z3;

for n0=1:72
    L=[abs(z-z1),abs(z-z2),abs(z-z3)];
    r = min(L);
    if (r<0.0001)
        if (r==abs(z-z1))
            q=1;
            return
        end
        if (r==abs(z-z2))
            q=2;
            return
        end
        if (r==abs(z-z3))
            q=3;
            return
        end
    end
    for n1 = 1:2
        z = z - (z^3 + P*z^2 + Q*z + R)/(3*z^2 + 2*P*z + Q);
    end
end
q=0;

```

Listing 5: 渗透模型

```

function ys()

r = 1200;
c = 1200;      %%N*N meshgrid
Allnum = 12000; %%average
%%% range p
delta_p = 0.003;
p0=0.45;%%start p
p = p0;
N = 100;
%%%%%

```

```

list = zeros(1,N);
for kk = 1:N
    num = 0;
    for k=1:Allnum
        A = start(p,r,c);
        B = A(:,1);
        for j=1:c
            B = judge(B,A(:,j));
        end
        if sum(B)>0
            num = num +1;
        end
    end

```

```

    end
    list(1, kk)=(num/Allnum);
    p = p + delta_p;
end
plot(p0:delta_p:(p-delta_p),list);

function Re = judge(B,A)
[r,~]=size(B);
Re = B;
for i = 1:r
    if (A(i,1)==0)
        Re(i,1)=0;
    end
end

for i = 1:r
    if (Re(i,1))
        for k1=i-1:(-1):1
            if (Re(k1+1,1) && A(k1,1))
                Re(k1,1)=1;

            end
        end
        for k2=i+1:r
            if (Re(k2-1,1) && A(k2,1))
                Re(k2,1)=1;
            end
        end
    end
end

function A = start(p,r,c)
A = zeros(r,c);
for i = 1:r
    for j =1:c
        A(i,j) = (rand()<p);
    end
end

```

参考文献

- [1] Kenneth Falconer. 分形几何. 北京: 人民邮电出版社. 2007
- [2] .F.Lange,C. Heinl,G. Li,C. Emmelmann Numerical optimization of active heat sinks considering restrictions of selective laser melting .Proceedings of the 2018 COMSOL Conference in Lausanne.
- [3] Comsol MultiPhysics. Sierpinski 分形单极天线
- [4] Anirban Karmakar . Fractal antennas and arrays: a review and recent developments
- [5] 李梓丰. 从电磁学课后习题到计算物理

Chaos Games and Fractals

Li Zifeng

2022, School of Physics, Nanjing University, Nanjing 210046

Abstract: As we all know, nonlinear dynamical systems such as the Newton-Raphson iteration and $Z_{n+1} = Z_n^2 + C$ can construct various fractal structures, the most famous of which are the Mandelbrot set and the Julia set. We are already familiar with the chaos and fractal structures caused by nonlinear phenomena. There are almost no examples in nature of linear transformations generating fractal structures, as their behavior is always predictable. However, if linear transformations are combined with randomness, a different result can be obtained. The Chaos Games is the most classic example of randomness generating fractals. This paper analyzes the fundamental cause of the ordered fractal structure caused by random behavior in the chaotic game and derives a theory that can explain and predict the behavior of the chaotic game. Finally, the application of fractal structures in physics is analyzed.

Key words: Chaos Games;fractal