# **单中心DLA模型模拟**

“程序”文件夹中：

3.c、3\_8.c、3\_6.c分别为4、8、6配位的计算程序的源码，linux下编译，对应可执行文件为 \*.out形式。

其中\*.jl实际上为json格式，可与\*.out放于同一文件夹下，运行\*.out后，用jupyter或IPython打开\*.jl并运行。

主要算法：

在c源文件中，定义1024×1024的矩阵来表示对应位置有无粒子。

模拟时，首先随机生成粒子初始整数坐标（0，y）, y∈[0,1024)与（x，0）, x∈[0,1024).

之后使用walk()函数使其向四周随机行走一步，

用done()函数判断其是否与已经生长的粒子相邻，

若相邻则停止行走，把现在位置对应的矩阵元改为1，表示粒子附着于此；

若四周无粒子则重复以上的行走、判断。

sandbox法中，每次计算矩阵中心的r×r的小矩阵中非零元的个数，之后增加r；

在盒计数法中，根据边长的约数把矩阵分块，计算每个分割出的小矩阵中是否有非零元，如果有，计数加一，直接计算下个小矩阵。每次输出计数，之后缩小分割矩阵的大小，增大分割数量，重复计算至每个分割只包含单个矩阵元。

在十字型4配位的条件下，模拟出的粒子如图1-a，

8配位的条件下，模拟出的粒子如图2-a，

六边形网格6配位的情况下，模拟图如图3-a，

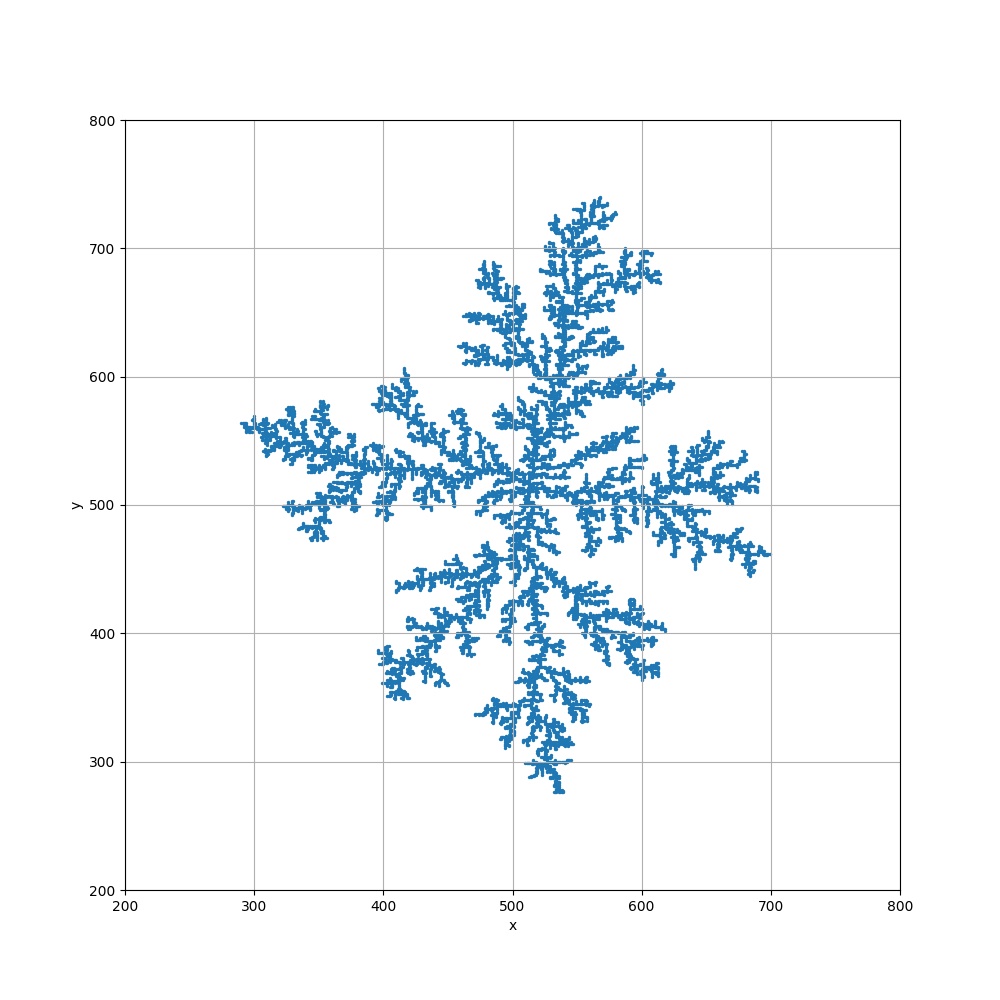


图1-a

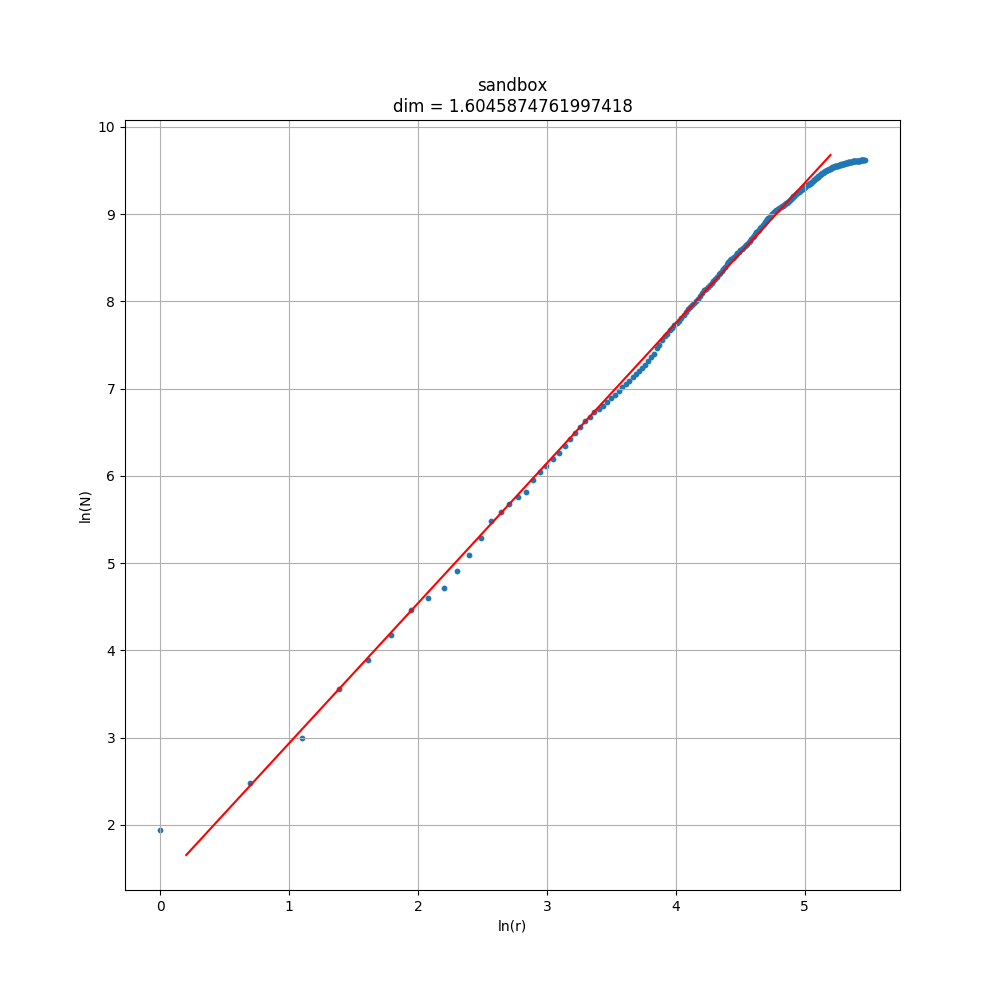


图1-b

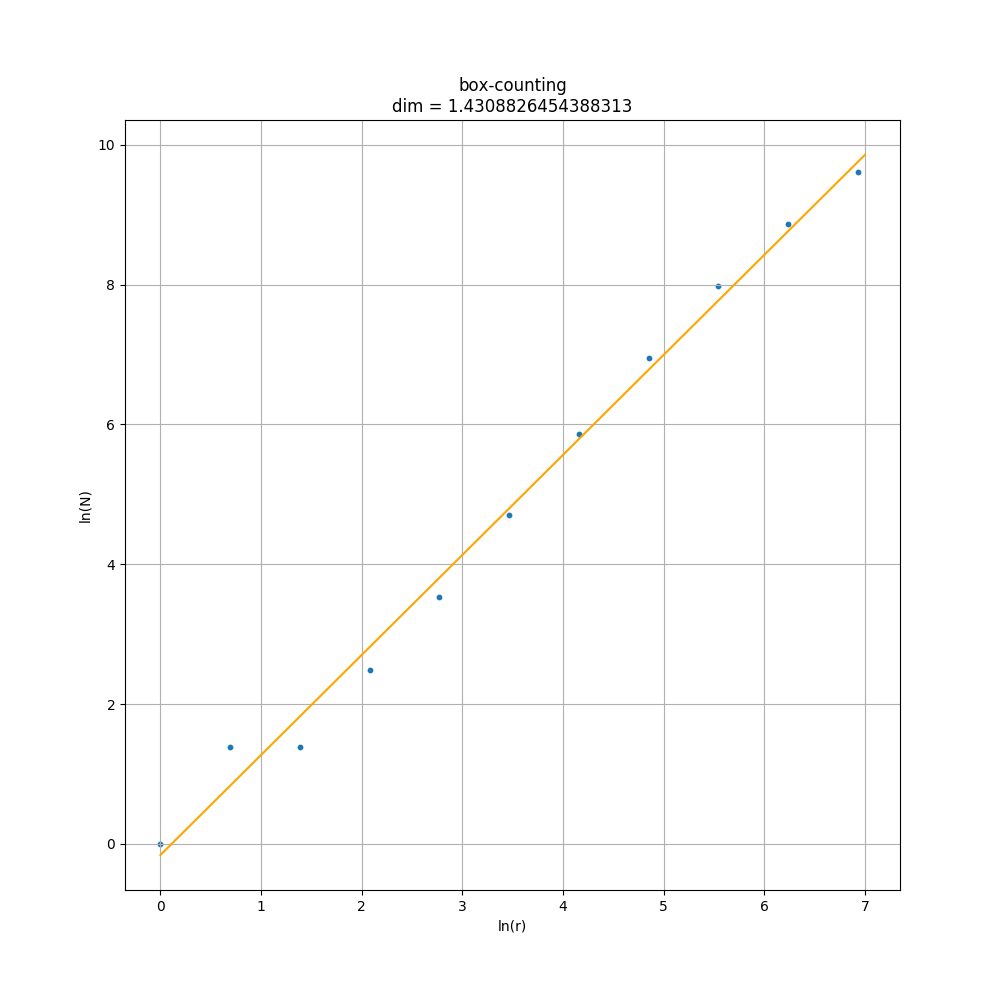


图1-c

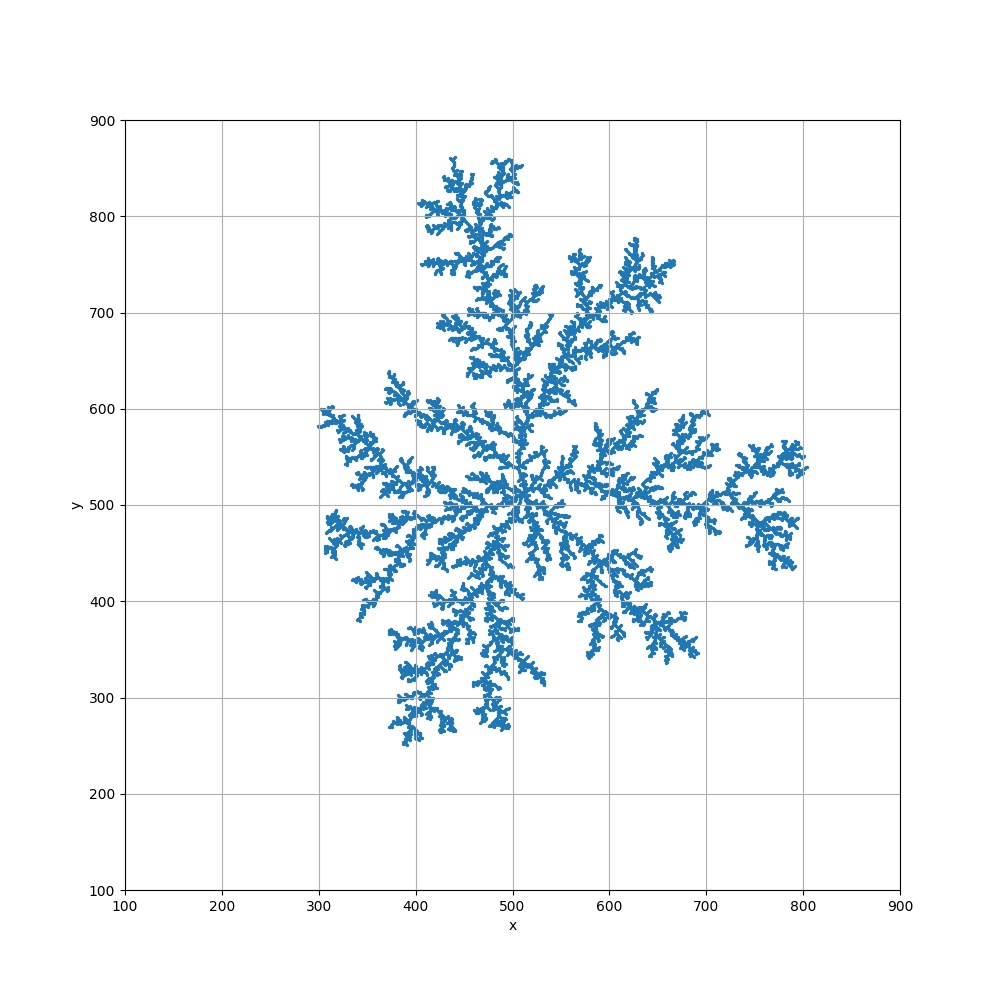


图2-a

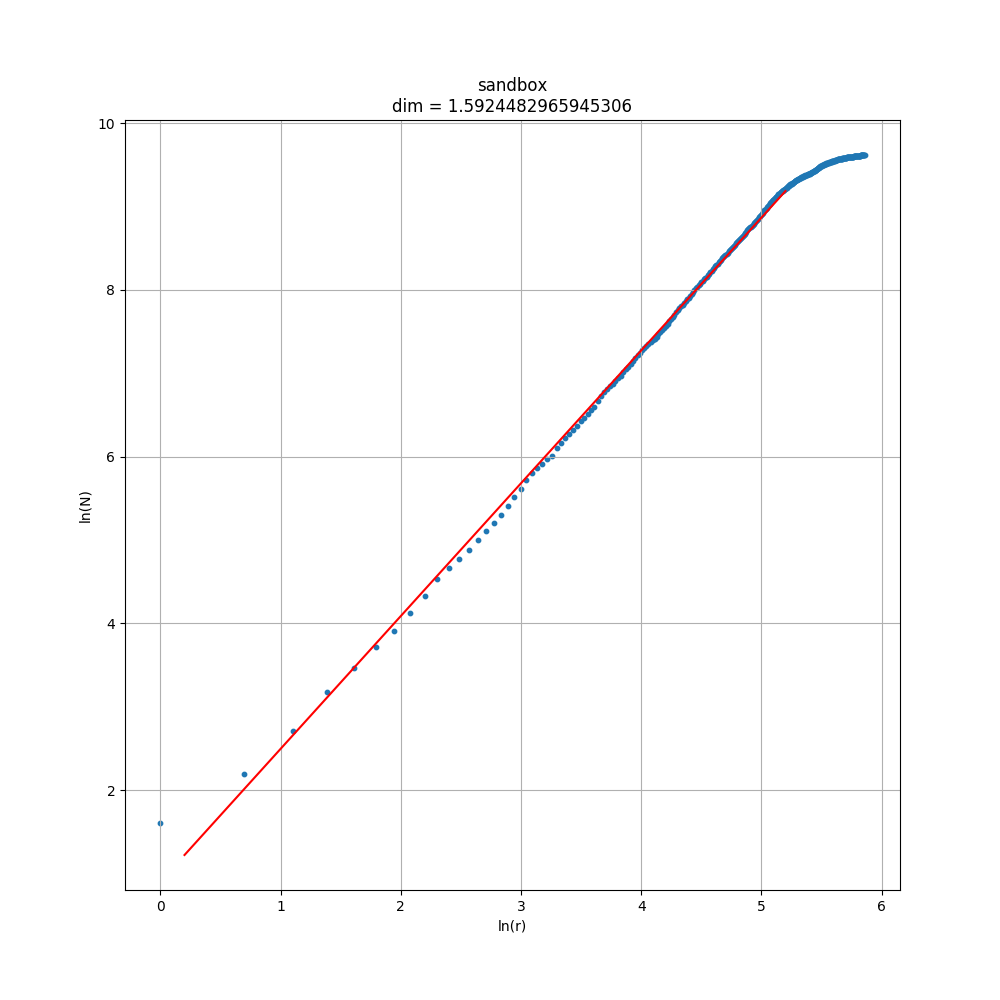


图2-b

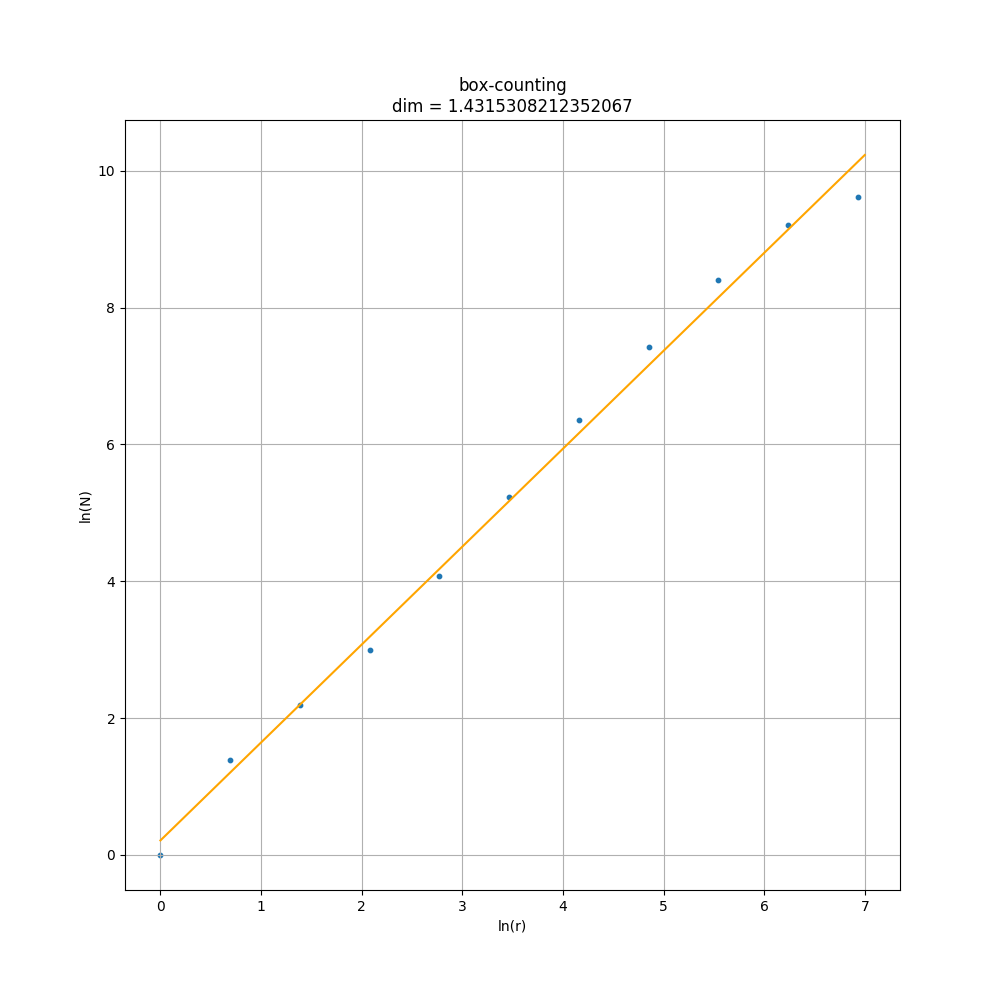


图2-c

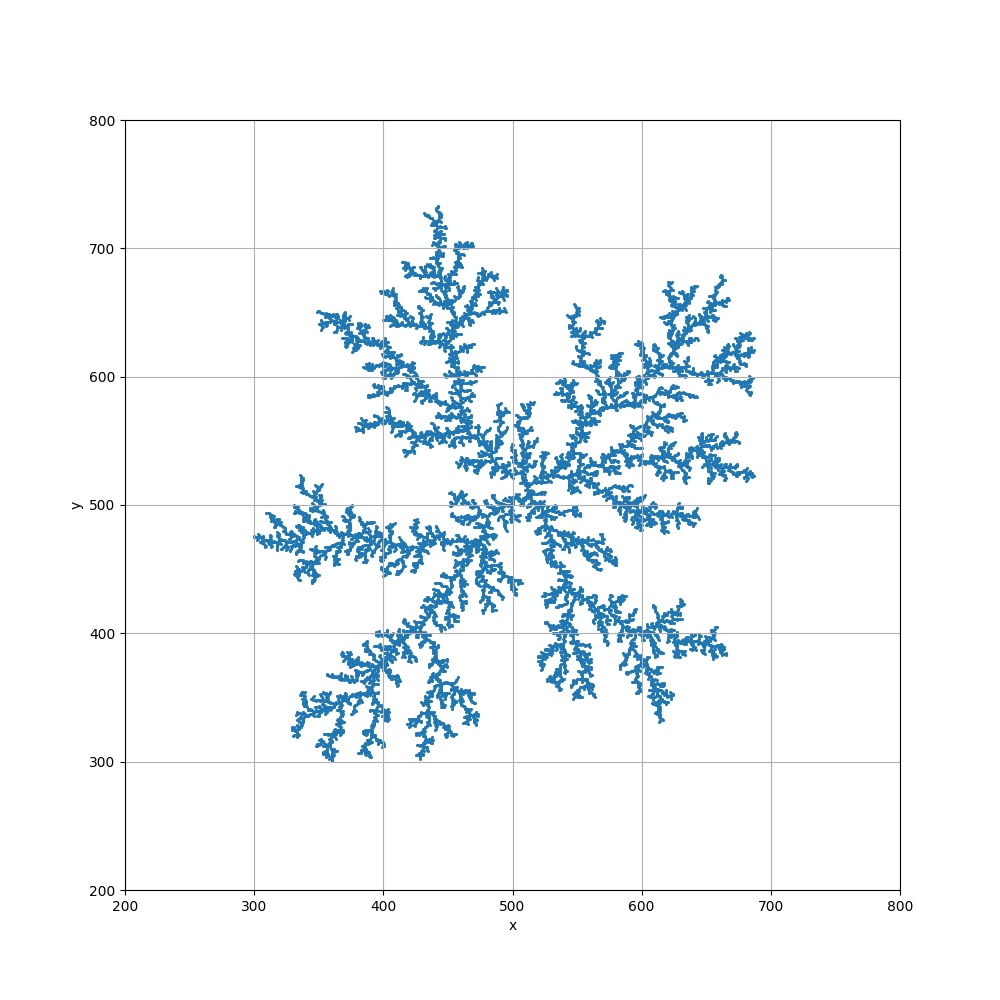


图3-a

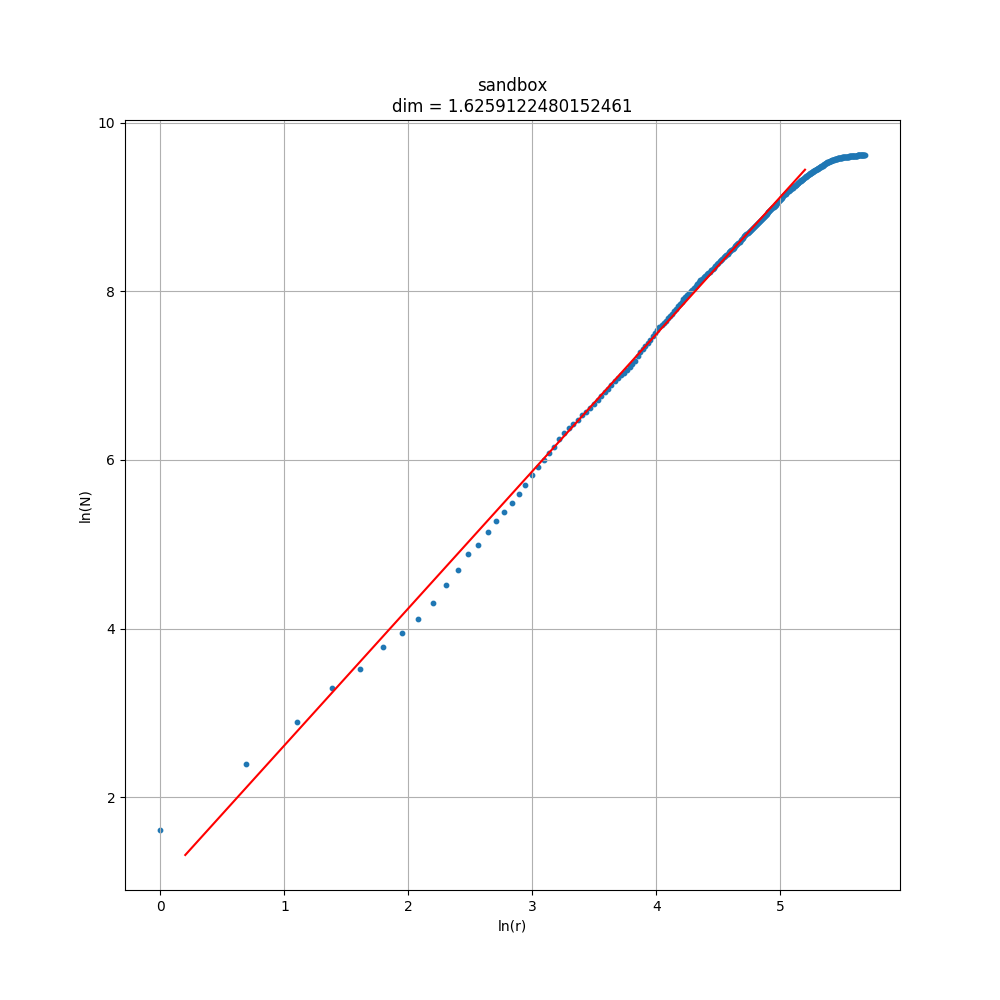


图3-b

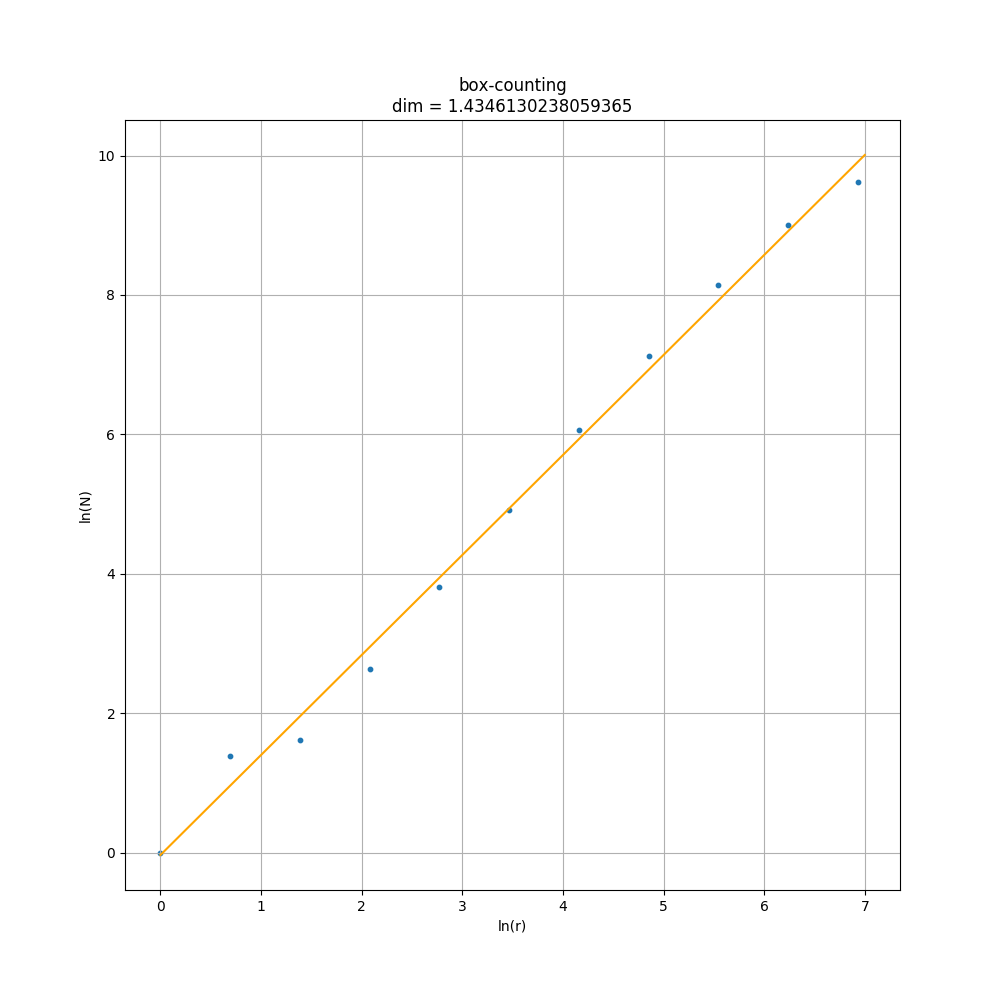


图3-c

在双对数图中，我们可以看到，使用sandbox法得到的分形维数在1.6附近，但是使用盒计数法得到的只有1.4左右。

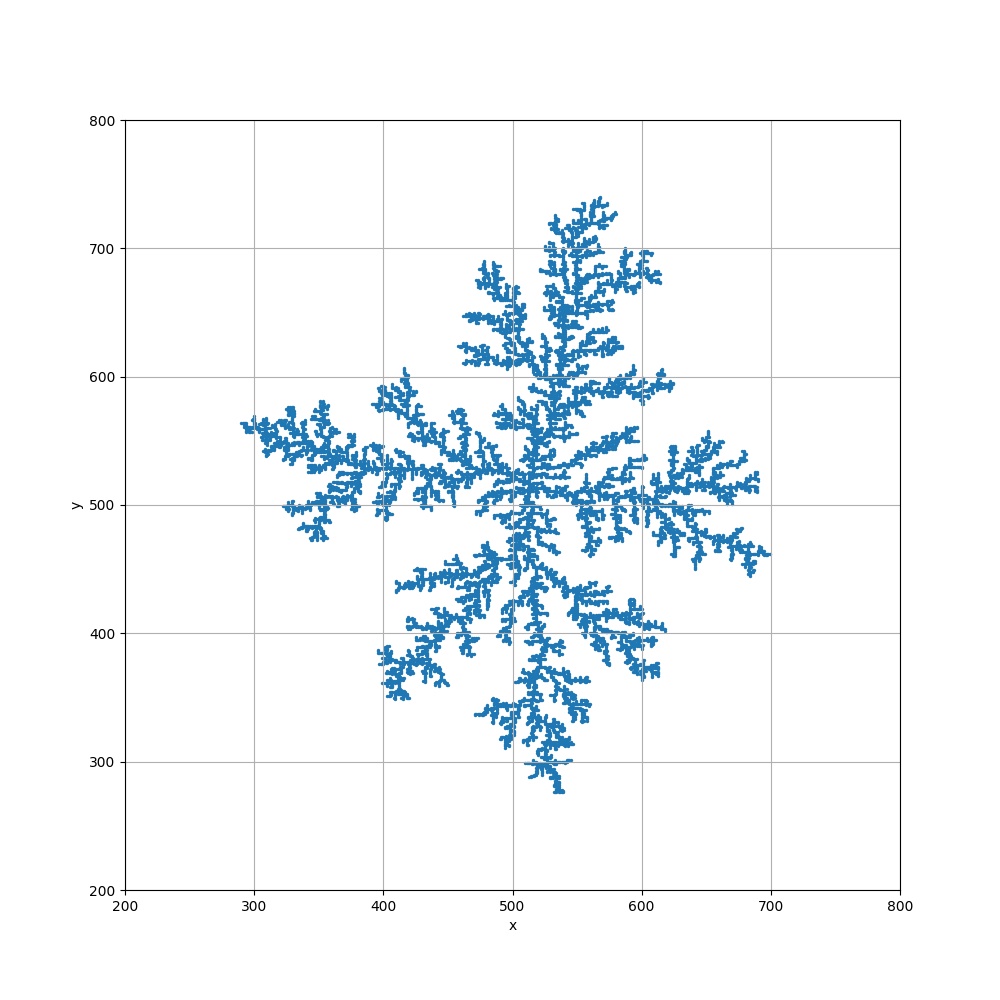
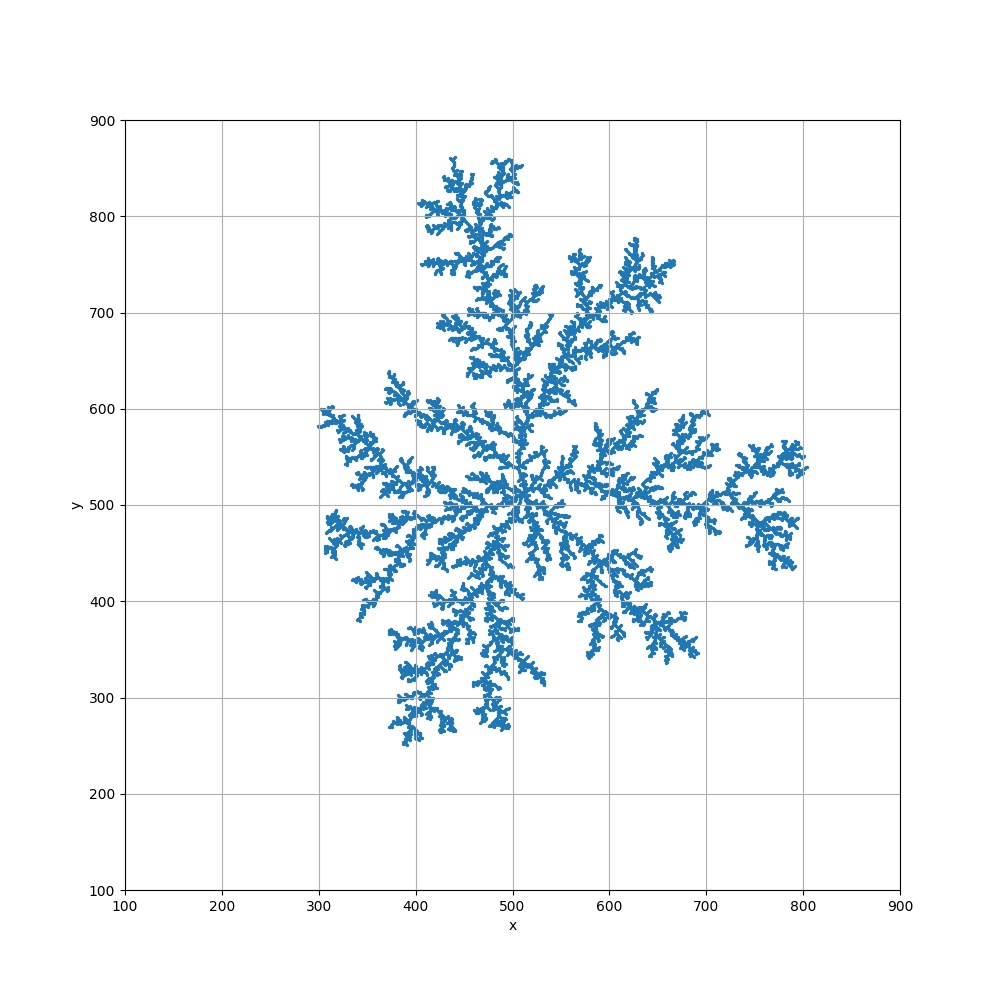
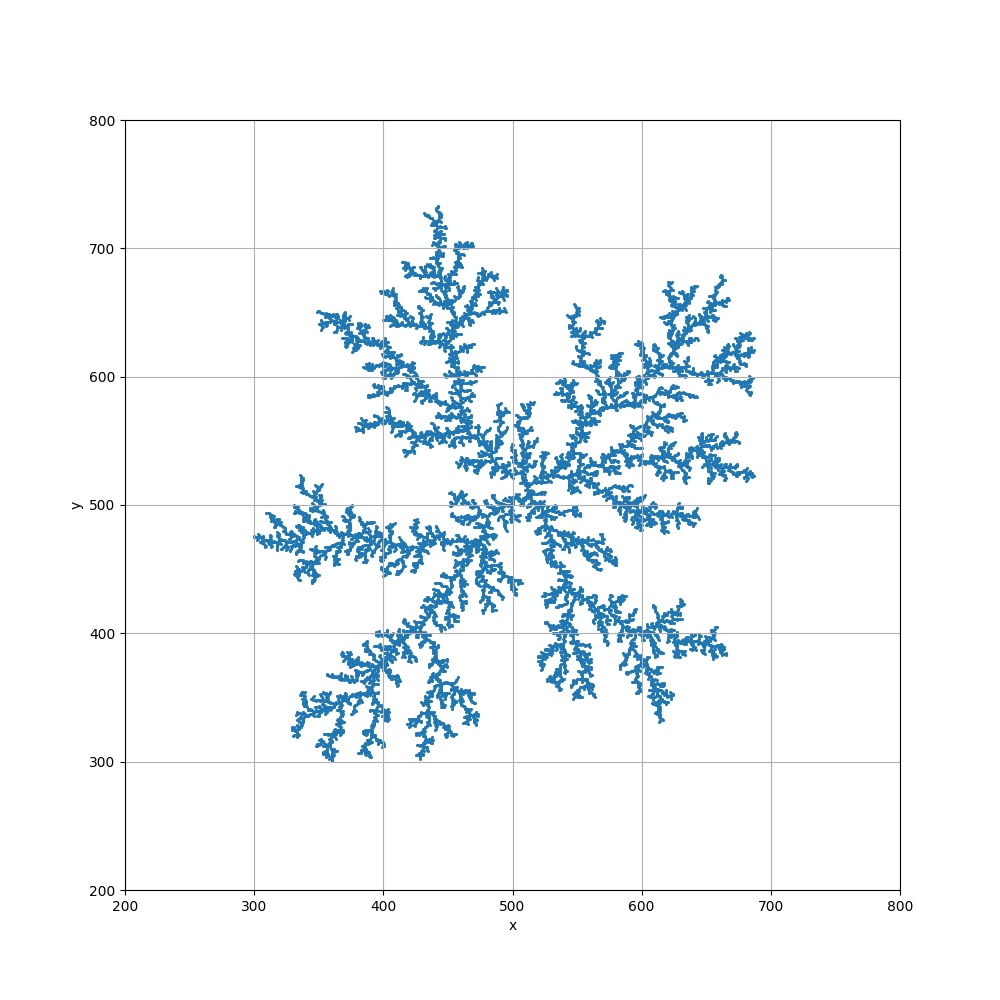
对于其中的差别产生的原因，我目前还没有定论，猜想原因可能是靠近图像尖端的区域正在快速生长，没有完全成为分形图形，而靠近中心区域远离尖端，外来粒子随机行走至此处概率小，所以生长较慢，形状几乎不变，可以认为已具有分形特性

在sandbox法中，我们可以只使用靠近中心的线性程度高的部分，忽略靠近尖端的还没有完全成长为分形图形的部分，所以结果较为准确；

在盒计数法中，我们的统计范围始终是整个图形，其中包括了正在生长的、形状尚未完全收敛的尖端区域，结果可能不准确；

可以稍加验证：在图b中，如果我们把线性区域右边的非线性区域也一起进行线性拟合，拟合结果中直线的斜率势必会减小，也就数说当sandbox法计入图片尖端区域时，会出现与盒计数法相同的问题，即计算出的维数下降。

比较1、2、3的a图，可以发现关于4、8、6配位得到的结果差别不大，唯一的区别可能是4配位模型图形的枝干更倾向于沿着x或y轴的方向生长

补充：4、8、6配位示意图