A Model of Diffusion-Limited Aggregation

A. M. L. H.
Department of Physics, University of Bath, Bath BA2 7AY, United Kingdom

Abstract

Computational models of diffusion-limited aggregation (DLA) present a way to simulate diffusion-based systems of interest, with control over various parameters of the simulation. This investigation involved a simplified variant of the DLA model to analyse the fractal-like characteristic of the aggregate clusters, dependent on diagonal/non-diagonal attachment and non-finite probability of attachment. The fractal dimension for a 4000-particle cluster was found to be 1.72±0.13 without diagonal attachment and 1.61±0.03 with.

1. Introduction

Particles in fluid suspension are understood to undergo Brownian motion [1], moving in 'random walks' as they collide with the molecules of the fluid [2]; more generally referred to as diffusion. As the particles diffuse in the fluid, they begin to attach to one another, forming clusters – or aggregates – of particles that grow over time. These clusters are referred to as Brownian trees, stemming from the process of their formation, and the dendritic structures that arise as they coalesce. These tree-like structures are, in fact, a type of fractal – an object with repeating patterns across different length scales [3].

One computational method for modelling the growth of such clusters was proposed by Witten and Sander in their 1981 paper "Diffusion-limited aggregation, a kinetic critical phenomenon" [4]. Here, 'diffusion-limited' simply means that the concentration of particles in the fluid must be low enough that the particles do not interact with each other when random walking – this is the 'rate-limiting step'.

DLA has subsequently found application in modelling a multitude of systems, all with predominantly diffusion-based transport. Several examples of these include dielectric breakdown, neuron growth, polymer branching and even coral formation [5]; the structures formed in these cases all demonstrate characteristic fractal-like natures. While other models have existed prior, such as percolating clusters, self-avoiding walks and the Eden growth model (of which the DLA model is a variant) for the study of growth processes, the work done by Witten and Sander on DLA renewed interest in their study, and marked the beginning of significant research into aggregation processes. Indeed, the freedom to manipulate initial conditions and parameters in computational simulations of DLA can aid in furthering understanding of the variables that influence aggregation, as well as the fractal properties that appear in their formation.

This investigation sought to implement a simple DLA simulation using C++ to study the effects of various parameter choices, such as diagonal/non-diagonal attachment and probabilistic attachment on the shape and growth of a 2-dimensional aggregate. The fractal dimension and its associated uncertainty was also analysed in relation the parameters, and compared to values in other published literature.

2. Theory and method

2.1 The DLA model

At its basis, diffusion-limited aggregation (DLA) is a deceptively simple model. The simulation described by Witten and Sander involved an arbitrary lattice coordinate system, with a single seed particle fixed at the origin. Another particle is generated some random distance from the seed, and allowed to random walk until it bumps into, and attaches to the seed. Particles that stray outside the bounds of the lattice are discarded, and another one introduced in its place. The process is then repeated over and over to grow the familiar dendritic cluster pattern [4]. While an effective method, a long time is needed to grow clusters as the random walks can take particles in circuitous routes before they finally arrive at the main cluster – potentially becoming computationally expensive.

For the purposes of this study, a modified version of the model above was used [3], producing much the same result in a shorter timeframe. This new method made use of three circular regions – the 'killing circle', 'starting circle' and 'cluster circle' – centred about the initial seed particle as in the original, figure 1 below. The radius of the cluster circle, $r_{max,}$ gives an estimate of the cluster size. The radius of the starting circle, r_{start}, is the radius within which new particles are introduced into the system.

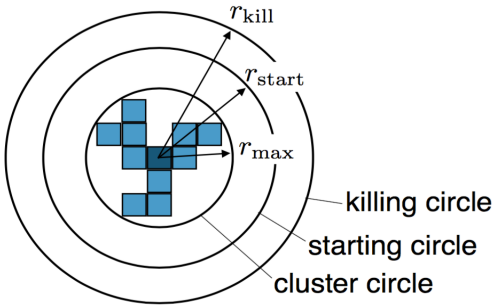

Figure 1: Illustration of the modified DLA model, showing the placement of the cluster and circles [3]. By default, the size of r_{start} was set to 1.2 times of r_{max}; r_{kill} was set 1.7 times larger.

The introduction of a 'killing circle' served to impose a limit on how far the particles would be permitted to random walk before removal; a new particle would then be generated in its place within the radius of the 'starting circle'. In this way, particles take shorter random walks before eventually sticking to the cluster, being prevented from moving too far towards the lattice bounds and thus reducing simulation time. Having particle generation only in the 'starting circle' further served to improve program efficiency, as particles further out would be stuck in an infinite formation-removal loop as they encountered the killing circle.

2.2 Fractals and fractal dimension

As mentioned in section 1, the intricate, branching patterns formed in DLA are examples of fractal behaviour; objects that demonstrate such behaviour are fractal objects. Magnification of a single branch yields a shape like that of the overall cluster – self-similarity across different length scales. One method to describe fractal objects is by determination of their fractal dimension. In this investigation, the number of particles N_c that form a cluster with radius r_{max} was used to calculate the fractal dimension, d_f. The general equation of this relation [3] is given as

$$N_c(r_{max}) = (\alpha r_{max})^{d_f} + \beta \tag{1}$$

where α, β are some unknown constants. Taking the natural logarithm of both sides and then differentiating with respect to lnr_{max} gives the new form of (1) as

$$\frac{d(lnN_c)}{d(lnr_{max})} = \frac{d_f}{1 + \frac{\beta}{(\alpha r_{max})^{d_f}}} \tag{2}$$

In the limit where r_{max} becomes large, the both α and β become negligible, and the left-hand side of (2) approaches d_f, allowing the fractal dimension to be estimated by plotting lnN_c against lnr_{max} and measuring the gradient of the straight line.

2.3 Estimation of errors

Error analysis is a crucial part of standard scientific rigour, allowing for proper justification of the experimental results. Here, the standard deviation and standard error of the results were examined, given by equations (3) and (4), respectively [6]

$$\sigma_X = \left[\overline{X(t)^2} - \overline{X(t)}^2\right]^{\frac{1}{2}} \tag{3}$$

$$\sigma_{\bar{X}} = \frac{1}{\sqrt{n-1}}\left[\overline{X(t)^2} - \overline{X(t)}^2\right]^{\frac{1}{2}} \tag{4}$$

$\overline{X(t)}$ is the average of $X_i(t)$, the measurements of some variable over a series of runs $i = 1 \ldots n$ in a time t,

$$\overline{X(t)} = \frac{1}{n}\sum_i X_i(t) \tag{5}$$

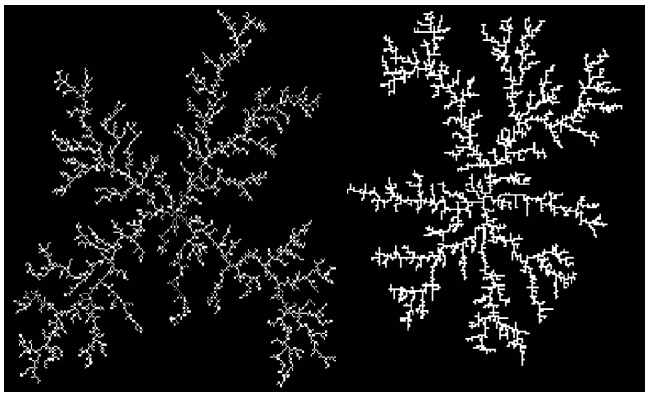

Figure 2: 4000-particle clusters generated with (a) and without (b) diagonal attachment enabled, illustrating their characteristics.

3. Investigative results

3.1 Diagonal sticking of particles

The first modification to the DLA model was the introduction of diagonal particle attachment. This was facilitated by declaring a new integer variable `attachDiag`. When set to 1, `attachDiag` would allow particles to occupy grid sites directly on the diagonals of the cluster. This created 4 additional positions for attachment, aside from along the horizontal/vertical axes, for a total of 8 possible sites. The value of `attachDiag` could be also toggled between 0 and 1 by pressing the 'j'-key while the program window is active, bypassing the need to re-compile the program to change attachment. A code snippet of this implementation can be found in appendix A.

Several DLA clusters were then generated, both with and without diagonal attachment. The effect of the attachment rule was observed for particle numbers N_c = 1000, 2000, 3000, 4000 over several runs of the code. Figure 2 shows two examples generated. Most notably, enabling diagonal attachment created clusters with significantly larger radii compared to purely horizontal/vertical attachment. The branches in figure 2(a) are seen to be thinner and sparser, diverging out further. In comparison, figure 2(b) shows thicker, more well-defined branches in the cluster, with denser grouping around the central seed.

These observations suggest that the additional attachment sites created by allowing diagonal attachment facilitate particles sticking to branches further out from the centre, as the particles undergo fewer steps in their random walk before encountering some part of the cluster. This also contributes to the comparative sparseness, as particles are less likely to attach to the innermost branches. The opposite is true without diagonal attachment, as the decrease in potential attachment locations lets the particles walk deeper into the structure before sticking, encouraging the branches to 'grow' thicker, rather than longer.

It was also noted that enabling diagonal attachment increased the time required to generate clusters. On average, generation of a 4000-particle cluster without diagonal sticking was completed within 4 minutes. However, the creating the same cluster but with the diagonal attachment rule caused the time to more than double, taking around 9 minutes. With the number of particles unchanged, an explanation for this two-fold increase is likely to be due to the increase in grid size with the cluster radius. Referring to the theory behind the model, a larger grid would increase the random-walk path length for particles; hence lengthening the time before they coalesce and requiring more computational resources to simulate.

3.2 Analysis of fractal dimension

The code was further modified to add a file-writing procedure, outputting the radius of the cluster to an external text file at 10, 100, 200, 300, 400..., N_c particles as the simulation ran. As runtime was noted earlier to be an issue, the file-writing syntax was implemented using the `'\n'` to end the call rather than `endl` as the default behaviour of `endl` is to continuously flush the write buffer with every iteration, slowing down the printing to file. Using `'\n'` instead reduced simulation and write times for a 4000-particle cluster to 7 minutes, from the 18 minutes needed previously – a substantial improvement

With file-writing in place, several runs were done to gather a sufficient data set for analysis. For each maximum number of particles, the code was run 5 times, yielding 5 sets of data for estimating the fractal dimension. As seen in figure 3, there is a wide variation in d_f below N_c =2000, with the values becoming more consistent with increasing N_c. The fractal dimension was found to be 1.62±0.03 at a maximum of N_c =4000. The fractal dimension obtained by Witten and Sander [4] was quoted as 1.66, larger than the upper bound the value from this DLA simulation.

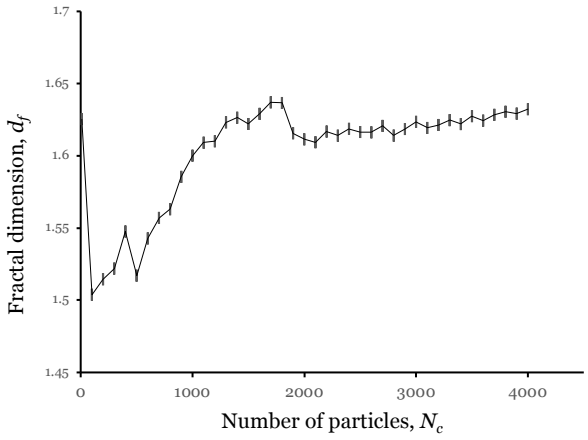

Figure 3: Visualisation of variation in fractal dimension with increase in cluster size, represented by number of particles. Standard deviation error bars used.

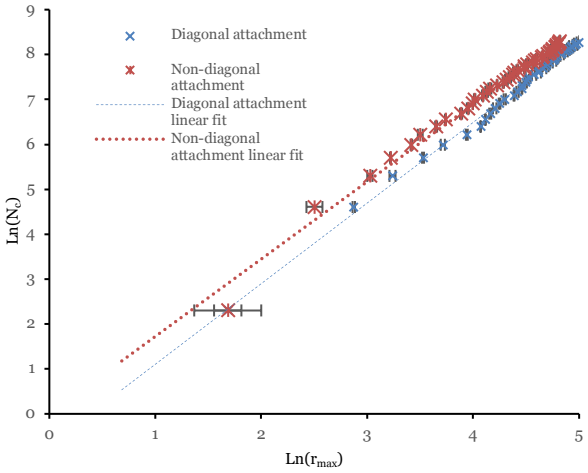

Figure 4: Comparison of plots for 4000 particles, with and without diagonal attachment enabled. Error bars calculated using the standard deviation

Next, the effect on diagonal attachment on fractal dimension was investigated. Figure 4 shows how enabling diagonal attachment causes the plot to shift to the right along the x-axis. As noted in section 3.1, the diagonal attachment allows greater growth of cluster radius for a constant particle number. For non-diagonal attachment, the fractal dimension with 4000 particles was determined to be 1.72±0.13; with diagonal attachment, the fractal dimension was instead 1.61±0.03.

3.3 Bouncing probability

One of usual assumptions in DLA models is that particles should always adhere with 100% probability upon contact with the cluster. However, real particles will have some amount of kinetic energy, allowing them to occasionally rebound off the cluster [3]. In computational DLA systems, this is implemented as having the particle 'bounce off' the cluster, and continue to random walk, either to the edge of the lattice or to another site where it might finally adhere. In this model, this 'bouncing off' of particles was handled by setting some finite attachment probability, p. The checkStick function was modified to take an additional function call, dependent on whether particles could 'bounce off' the cluster. The function bounceCondition then determines if the particle should attach to the cluster, based on whether the variable stickProbability was less than some randomly generated number.

The simulation was then tested for various values of p = 0.05, 0.5, 1 (which was used as a control value) for 4000 particles without diagonal attachment, shown in figure 5. At p=0.5, the branches were marginally less sparse than at p=1, despite the probability decreasing by half. A more significant result was obtained for p=0.05, with the structure being significantly denser, giving it a 'hairy' appearance. It would be inferred that further dropping the value of p to, say, 0.02 might yield a very densely-packed, almost solid cluster.

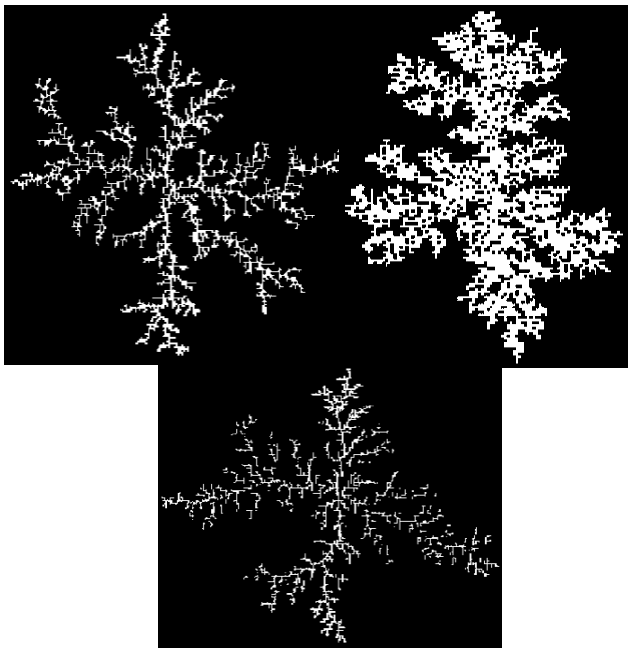

Figure 5: Clockwise, from top left - comparison of results for p =0.5, p= 0.05 and p=1. Note the increase in branch thickness with decreasing probability of attachment.

Running the simulation with a greater number of particles might have yielded more interesting results, however due to computational time restraints this was not possible. The cluster at p=0.01 was generated over 20 minutes, much longer than any other simulation thus far.

4. Conclusion

Allowing for diagonal attachment was seen to result in an increase in overall cluster radius, due to there being greater locations for particle attachment, as seen in figure 2. This however came with an increased computational cost, with a 4000-particle simulation taking 9 minutes on average.

A greater number of particles used in the simulation was seen to improve the estimate for fractal dimension, with the smallest standard deviations seen at numbers above 2000 particles, figure 3. The fractal dimension for non-diagonal attachment with 4000 particles was found to be 1.72±0.13, and 1.61±0.03 for diagonal attachment. The value obtained by Witten and Sander was 1.66, of which the values from this model failed to match. This suggests that the simplifications introduced in this model may have caused it to be an inadequate estimate, despite the relatively small standard deviation obtained.

Introduction of a finite attachment probability was seen produce interesting changes in the cluster formation, however the long times needed to generate even a small cluster made further study prohibitively time-consuming.

References:

[1] Brown, R., 1828. XXVII. A brief account of microscopical observations made in the months of June, July and August 1827, on the particles contained in the pollen of plants; and on the general existence of active molecules in organic and inorganic bodies. *The Philosophical Magazine*, *4*(21), pp.161-173.

[2] Einstein, A., 1956. *Investigations on the Theory of the Brownian Movement*. Courier Corporation.

[3] Rimpilainen, V. J. T., 2018. PH30056 Computational Physics B, DLA Coursework Assignment. University of Bath.

[4] T. A. Witten and L. M. Sander, Diffusion-Limited Aggregation, a Kinetic Critical Phenomenon, *Phys. Rev. Lett.* 47 (1981) 1400-1403.

[5] Tokuyama, M. and Kawasaki, K., 1984. Fractal dimensions for diffusion-limited aggregation. *Physics Letters A*, *100*(7), pp.337-340.

[6] Rimpilainen, V. J. T., 2018. PH30056 Computational Physics B, Lecture 4. University of Bath.

Appendix

A) Code snippet for toggle of `attachDiag`

```
 case 'j':
            sys->attachDiag = (sys->attachDiag == 0 ? 1 : 0); // ternary operator
used to switch value of attachDiag
            cout << "diag: Diagonal attachment - " << (sys->attachDiag == 0 ? "OFF" :
"ON") << endl;
            break;
```

B) File printing procedure

```
 if ( checkStick() ) {
            //cout << "stick" << endl;
            setParticleInactive();  // make the particle inactive (stuck)
            updateClusterRadius( lastP->pos );  // update the cluster radius, addCircle,
etc.

            if ( logfile.is_open() ) {
                if ((numParticles == 10) || (numParticles % 100 == 0)) {
                    logfile << numParticles << " " << clusterRadius << '\n';
                }
            }

        }
```

C) Setting sticking probability

```
bool DLASystem::bounceCondition() {
    if (!bounceEnabled || stickProbability == 1.0) {
        return false;
    }

    int m = 1000;
    int x = rand() % (m + 1);
    double ratio = x/m;

    bool shouldStick = (ratio < stickProbability);
    bool bounceCondition = !shouldStick;
    return bounceCondition;

}
```