Title

Candidate Number: 24669

Department of Physics, University of Bath, Bath BA2 7AY, United Kingdom

March 22, 2022

Abstract

Diffusion limited aggregation (DLA) clusters also known as Brownian trees are branching fractal objects that arise from the aggregation of random-walking particles. A fractal is an object with non-integer dimension, its uniform mass density grows by a non-integer power of its linear size. Because of their self-similar properties and scale invariance DLA systems are used to model a variety of physical and biological processes, as such the effects of the conditions in a DLA simulation on the resulting DLA cluster are well studied. This report examines the fractal dimension of DLA clusters generated from an initial 'seed' particle with varying probability of particle adherence. By employing a least squares fit of the expected relationship between particle number N_c and linear size r_{max} on data from simulated clusters we determine the fractal dimension d_f of a DLA cluster to be 1.696±0.044. Further, we compare the efficacy of our method with an alternate method of deriving dimension and show a negative correlation between the probability of particle adhesion and fractal dimension.

Introduction

The primary mechanism in Diffusion limited aggregation (DLA) systems is Brownian motion, named for early contributor Robert Brown [1] and later studied by Einstein [2], whereby fluctuations within a medium impart forces on diffuse particles that result in seemingly random motion. *Ab-initio* simulation of all of the constituent particles of a diffuse substance in a medium is computationally expensive, however it has been shown by Donsker's theorem that Brownian motion is well approximated by the stochastic Weiner process. The Weiner process can be modelled as a discrete random walk on a lattice in the limit as the lattice constant tends to zero (the scaling limit), thus for practical computation a discrete random walk is a sufficient approximation of Brownian motion. Additionally some physical systems have characteristic length scales, further justifying their approximation by a discrete random walk.

DLA models produce DLA clusters also known as Brownian trees, fractal objects that grow by adhering random-walking particles to one another. A common initial condition in the study of natural systems is a single initial particle onto which the diffuse particles attach after random walking towards the structure from some starting radius. The single particle initial state model was first proposed by Eden in 1961 [3] to describe the growth of bacterial colonies from an initial particle as a stochastic process. Further developments by Witten and Sander in 1981 [4] on metal-particle aggregation introduced clustering by Brownian motion to the Eden model, often referred to as 'classical' DLA, they also show DLA clusters to be critical phenomena arising from irreversible growth processes rather than an equilibrium ensemble. Other studies have extended the model to additional spatial dimensions [?], or varying geometries such as curved [?] and spherical [5] surfaces. Further, Laplacian growth theory has succeeded in describing varying DLA systems along with other fractal generating systems under a single universality class [6] [7].

DLA systems and the corresponding fractal geometry of DLA clusters are used to describe many natural phenomena because of their scale independence and self-similar nature. Some well studied examples of DLA arise in electrodeposition [source], dielectric breakdown [?] [8], dendrite formation[source], bacterial colony growth [3], biomedical imaging[source], and plant growth [?] [?]. The physical interpretation of fractal dimension relating the growth and density of fractal and self-similar phenomena is valuable in many applications.

Modifications to specific aspects the classical DLA model can result in significant changes to the generated DLA clusters. Example modifications include: changing the initial conditions of the system, including initial particle placement; altering the location that new particles are introduced before they undergo Brownian motion [9]; introducing different species of particles with differing adherence rules to the adhesion mechanism by altering the probability that a particle will successfully adhere to the cluster on contact [9]. The changes in the DLA cluster induced by these modifications can effect the direction of growth (e.g. producing spiral clusters) and significantly, the dimension of the cluster. Ranguelov *et al* [9] found that lowering the probability of particles adhering on contact in-

creases the dimension of the DLA cluster. This report aims to verify some of these findings by determining the fractal dimension of a classical DLA cluster, then comparing with the dimension of clusters generated with an alternate scheme that varies the probability of adhesion.

Method and simulation

Classical DLA algorithm

Our DLA algorithm is similar to that described by Witten and Sander [4]. A variation on the Eden model [3], an intitial particle is placed at the origin of a two dimensional square grid of predetermined size. Three radii from the origin are defined r_{max}, r_{start}, and r_{kill}. r_{max} is the radius of the furthest particle in the cluster from the origin, and is used to estimate the linear size of the cluster. The distances $r_{start} = 1.2 \cdot r_{max}$ and $r_{kill} = 1.7 \cdot r_{start}$ determine the radii of creation and deletion of active random-walking particles in the system. The DLA algorithm is as follows

1. Initialise the system with one particle at the origin the grid. r_{start}, and r_{kill} are given suitable initial values.
2. Create a new particle at a random location on the circle defined by r_{start} set it as *active*.
3. **while** there is an *active* particle:
   - **if** the position of the particle has distance from the origin greater or equal to r_{kill} **then** the particle is removed.
   - **else if** the position of the particle is adjacent to a particle on the cluster **and** *checkStick()* **then** the particle is added to the cluster and becomes *inactive*.
   - **else** the *active* particle performs a discrete random walk.
4. **if** the cluster is the desired size **stop**, **else goto** 2.

there are several things to note about the algorithm and its implementation In this initial implementation of the DLA system we define *checkStick()* to always return *true*, so a particle will always adhere on contact with the cluster. The initial start radius $r_{start} = 5$, is chosen such that the cluster can be generated for several starting particles (otherwise r_{max} as defined would start at 0). The domain of the simulation given by the grid size is predetermined, our simulations were performed in 3200×3200 grids, these are large enough to generate clusters with 9×10^5 particles. Once r_{kill} reaches half the grid size the simulation stops, our results do not use clusters of more than 5×10^5 particles, hence the selected grid size is sufficient.

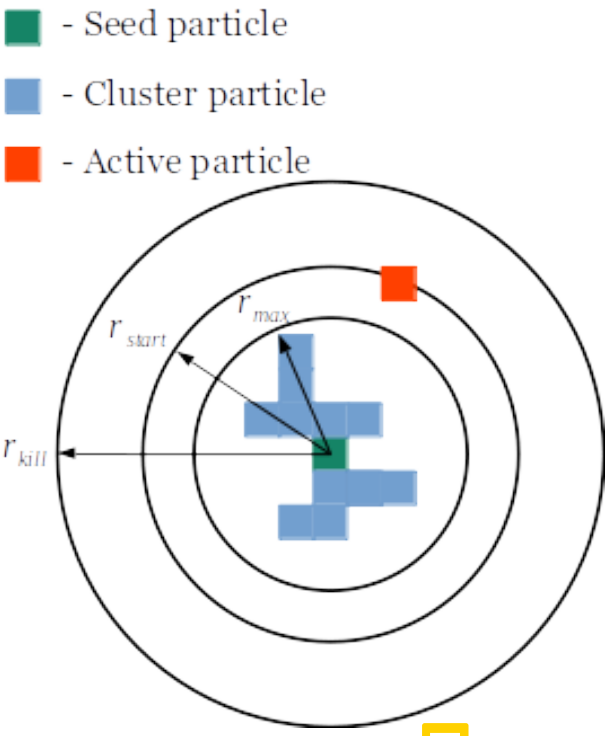

Figure 1: Diagram of the diffusion limited aggregation (DLA) system. A particle is created at a random point on r_{start} and then undergoes a discrete random walk. If its position exceeds the distance r_{kill} it is removed, if it makes contact with the cluster it is made inactive and joins the cluster. The DLA cluster grows from an initial seed particle located at the origin, its size can be estimated from r_{max}.

Some approximations are made in this algorithm to improve computational efficiency. The original model by Witten and Sander [4] removes particles when they reach the edge of the grid this can result in significant computation spent on random walking a particle in the domain only for it to walk to the edge and be removed. Further, if it does random walk back to radius r_{start} its previous motion away from the cluster will not effect its subsequent motion. We therefore restrict the random walking to within r_{kill} which still allows for some motion away from the cluster while improving overall efficiency. This restriction will have an effect on the resulting DLA cluster however we assume the effect is negligible for the reasons outlined.

The random number sampling used for the random walk and the initial positions of particles generated on r_{start} uses pseudo-random number generation to efficiently generate large quantities of random numbers. To ensure that the system does not generate the same cluster every time, the *seed* of the pseudo-random number generater is generated as a 'true' random number, allowing for the efficient sampling of many psuedo-random numbers that are independent between simu-

lations, the resulting DLA clusters are consequently independent from one another.

Modified DLA - random adhesion

Previously we had defined *checkStick()* to always be *true*, however in the modified algorithm described by Ranguelov *et al* [9] we redefine *checkStick()* to check against a preset probability of particle adherence P_{stick}, retrieving the unmodified algorithm for $P_{stick} = 1$. If the particle fails to adhere to the cluster it will do nothing until the next update where it will perform another random walk. An additional modification is made to the random walk procedure, such that if a particle tries to move into an occupied grid position it will not move and instead *checkStick()* again. This prevents multiple particles from occupying a single grid space and allows particles to have multiple 'chances' to adhere to the cluster at the same site.

Fractal dimension

A definition of dimension suitable to numerical computations of fractal dimension is the Minkowski-Bouligand or 'box-count' dimension where, in a Euclidian space, the number of boxes N (or 'density') with side length a in a grid that cover a fractal of one dimensional size R are related by

$$\begin{aligned} N &= (R/a)^{d_f} \\ \implies d_f &= \frac{\ln(N)}{\ln(R/a)}, \end{aligned} \tag{1}$$

where d_f is the fractal dimension. With this definition we can retrieve known dimensions such as $d_{line} = 1, d_{square} = 2$ etc.

To determine the fractal dimension of the generated DLA clusters we first set $a = 1$ as our DLA system is discrete in the grid, then $N = N_c$ is the number of particles in the cluster. We then assume that r_{max}, the distance of the furthest point in the cluster from the origin is representative of the one-dimensional size of the cluster with some correction terms α, β. So our general relation is

$$\begin{aligned} N_c(r_{max}) &= (\alpha r_{max})^{d_f} + \beta \\ \implies \frac{d(\ln N_c)}{d(\ln r_{max})} &= \frac{d_f}{1 + \beta/(\alpha r_{max})^{d_f}} \\ \implies d_f &= \lim_{r_{max}\to\infty} \frac{d(\ln N_c)}{d(\ln r_{max})}, \end{aligned} \tag{2}$$

hence for sufficiently large r_{max} relative to the length scale $a = 1$ we can estimate the fractal dimension d_f of a DLA cluster.

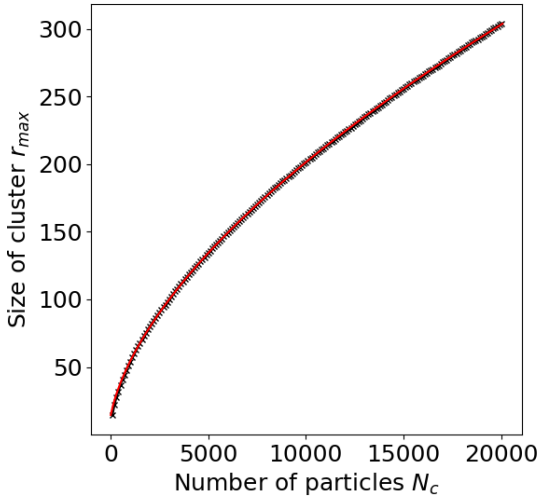

Figure 2: Mean cluster size r_{max} against particle number N_c over a sample of 100 generated DLA clusters. Error bars representing the 95% confidence interval on the mean of r_{max} are comparable to symbol sizes and omitted for clarity. A least squares fit of the general relationship described in 2 is overlaid in red. The mean fractal dimension d_f of the 100 cluster sample is found to be 1.696±0.044.

Implementation

The DLA data discussed in this report was generated by an implementation of the described DLA algorithm in C++. Standard C++ libraries are used for sampling of 'true' and psuedo-random numbers used in the random walk and particle generation. Utilities were written to record multiple simulations with varying parameters and to export recorded data from simulations to a csv format. Data analysis and derivation of fractal dimension was performed in Python using the numpy and pandas libraries. All computations were performed on a 64-bit desktop processor running linux mint 20.1 'Ulyssa'.

Please see the appendix for further practical details about the specific DLA program used for this report.

Results

DLA cluster fractal dimension

Using the DLA system as described we have generated datasets from several DLA clusters of varying sizes. In particular a dataset of 100 clusters of $N_c = 20000$ particles is generated with recorded values for N_c and r_{max} as they grow. We make the assumtion that the distribution of r_{max} and corresponding fractal dimension d_f between randomly sampled clusters is normal - allow-

ing for the statistical treatment of error. Unless otherwise stated any uncertainties given will be the 95% confidence interval on the mean.

To derive the fractal dimension d_f a least squares fit of the relationship described in (2) is applied to each DLA cluster generated. A fit on the mean of $r_{max} = \overline{r_{max}}$ is shown in FIG 2 to illustrate the fitting method, it should be noted that in the final derivation of d_f the mean is taken on the 100 derived d_f and not on r_{max}. The mean of the values of d_f of each cluster is 1.696±0.044, where the uncertainty includes a correction for fitting error. This measurement agrees with other literature which suggest a fractal dimension of 1.7[source]. Comparing this with physical systems we find that NUMBER[source]

It should be noted that in the derivation of fractal dimension several approximations are made, primarily that r_{max} is is representative of the one dimensional size of the DLA cluster and that the relationship between N_c, r_{max} and d_f given by (2) is accurate at all scales. These assumtions do not fully address the discrete nature of the simulation as our simulations have a characteristic length scale representated by the grid spacing (set as 1). A key property of fractals are their scale invariance, which no longer holds in our model for clusters of small particle number N_c. This can however be useful in modelling systems which do have characteristic length scales and discrete particles such as electrodeposition. Other models may also generate DLA clusters from circular particles[source] or on non-square grids[source], for smaller sized clusters this can better emulate Brownian motion due to the increased degrees of freedom, however in the limit as the cluster size increases these should be equivalent to the Weiner process[source].

Effect of random adhesion on fractal dimension

To examine the effects of random adhesion on the fractal dimension of DLA clusters a dataset of 100 clusters per sticking probability $P_{stick} \in [0.1, 1]$ with interval 0.1 (1000 clusters total) of size $N_c = 5000$ was generated. The mean dimension is determined as in the $P_{stick} = 1$ case, that is d_f are found for each cluster and the mean $\overline{d_f}$ is taken over the 100 samples at each P_{stick}. The result is 10 mean values of d_f with statistical uncertainty (95% confidence interval on the mean) for each P_{stick}.

Two methods are employed in the derivation of d_f for individual clusters: firstly the least squares fit method as described in the previous section, where a fit of the relationship described by (2) is applied to the values of N_c and r_{max} of a cluster and the corresponding d_f with fitting error is inferred; the second method approximates

$$d_f \approx \frac{\ln N_c}{\ln r_{max}}, \quad r_{max} \approx R >> a \tag{3}$$

by assuming that r_{max} is sufficiently large when taking d_f for $N_c = 5000$ of the fully grown cluster. FIG is a plot of $\overline{d_f}$ against P_{stick} that compares the results both methods of deriving d_f

Both of the methods outlined have significant drawbacks that are important to consider when interpreting the results they yield. For the fitting method the relationship described in (2) may not hold for smaller clusters: the dummy variables α and β are included to allow for corrections, however the least squares procedure will give equal weight to the data at small N_c which is not as representative of the fractal properties of the clusters that appear at cluster sizes much larger than the grid spacing. There are ways of mitigating this however, with a sufficiently large cluster the fit can be applied to data that omits N_c smaller than a cutoff value, better representing the fractal growth of the cluster for $r_{max} >> a = 1$. Larger clusters require significantly more computation which is why the dataset we have used only generates clusters to $N_c = 5000$, the consequences of this can be seen in $\overline{d_f}$ when $P_{stick} = 1$ which fails to reproduce to within error the $\overline{d_f}$ determined in the previous section for $N_c = 20000$. A smaller dataset of larger clusters was created however the statistical uncertainties in the $\overline{d_f}$ were too large (20%). Even when taking the larger number of samples the statistical uncertainty in $\overline{d_f}$ is significant and can be seen in FIG 4. Fits on the $\overline{r_{max}}$ are shown in FIG 3 to illustrate the fitting method - these fits do not describe $\overline{d_f}$. In fact when overlaid, fits that use the derived $\overline{d_f}$ poorly align with $\overline{r_{max}}$ and N_c, suggesting that $\overline{d_f}$ and $\overline{r_{max}}$ as defined may not be well related by (2), which could present an issue in application to datasets concerning $\overline{r_{max}}$.

The second method that relies on the approximation in (3)

Conclusion

begin conclusion

Ranguelov *et al* also make modifications to the radius at which the particles are generated $r_{start}(\theta)$ by setting it the radius of the cluster at angle θ + a constant, they also introduce multiple species of particles which have different probabilities of attaching to each other. The resulting clusters can vary drastically in shape and dimension. We do not implement these modifications in our simulation, but they provide good examples of modifications that can be made to a DLA system.

poincare results

mars results

End conclusion

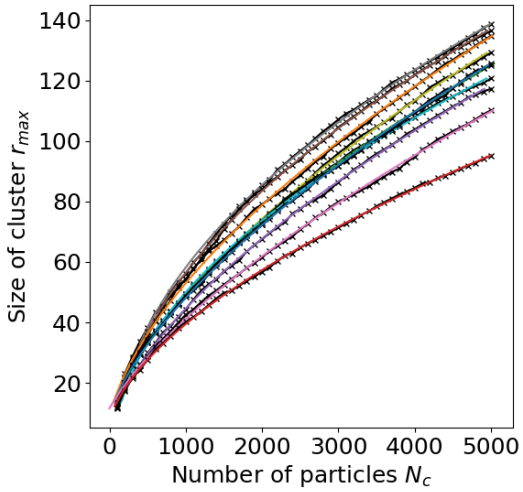

Figure 3: Mean cluster size r_{max} against particle number N_c over a sample of 100 clusters for each sticking probability $P_{stick} \in [0.1, 1]$ with interval 0.1 (1000 samples in total). Error bars representing the 95% confidence interval on the mean of r_{max} are comparable to symbol sizes and are omitted. Fits of the relationship described by 2 for each P_{stick} are overlaid. Note this graph is only included to demonstrate the fitting procedure and is not related to the d_f shown in FIG 4.

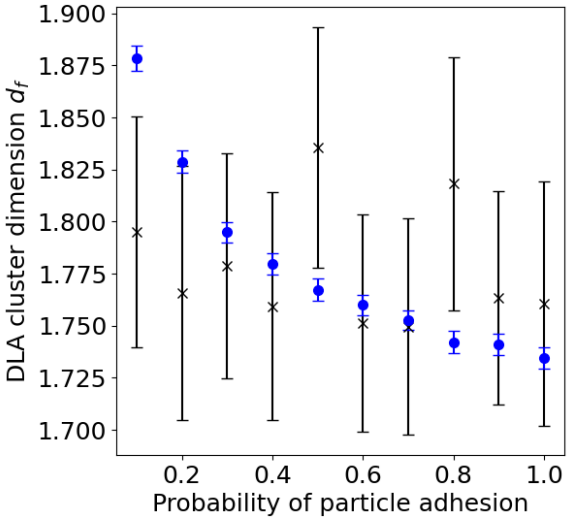

Figure 4: Derived mean fractal dimension d_f against sticking probability P_{stick}. Two derivations of d_f are shown: in blue d_f is defined assuming the limit as in 1 by taking $N_c = 5000$ and corresponding r_{max}; in black d_f is determined by fitting the relationship in 2 to each cluster and taking a mean. In both error bars represent a 95% confidence interval on the mean.

Acknowledgements

We would like to acknowledge the contributions of Dr A. Souslov and Dr D. Tsang for their code and resources on which the unmodified DLA model is based.

References

[1] P. Pearle, B. Collett, K. Bart, D. Bilderback, D. Newman, and S. Samuels, "What brown saw and you can too," *Am. J. Phys.*, vol. 78, p. 1278–1289, 2010.

[2] A. Einstein, "Über die von der molekularkinetischen theorie der wärme geforderte bewegung von in ruhenden flüssigkeiten suspendierten teilchen," *Annalen der Physik*, vol. 322, no. 8, pp. 549–560, 1905.

[3] M. Eden, "A two-dimensional growth process," in *Proceedings of Fourth Berkeley Symposium on Mathematics*, vol. 4, p. 223–239, University of California Press., 1961.

[4] T. A. Witten and L. M. Sander, "Diffusion-limited aggregation, a kinetic critical phenomenon," *Phys. Rev. Lett.*, vol. 47, pp. 1400–1403, Nov 1981.

[5] J. M. Tenti, S. N. Hernández Guiance, and I. M. Irurzun, "Fractal dimension of diffusion-limited aggregation clusters grown on spherical surfaces," *Phys. Rev. E*, vol. 103, p. 012138, Jan 2021.

[6] J. Mathiesen, I. Procaccia, H. L. Swinney, and M. Thrasher, "The universality class of diffusion-limited aggregation and viscous fingering," *EPL*, vol. 76, pp. 257–263, 2005.

[7] J. R. Nicol'as-Carlock, J. L. E. Carrillo-Estrada, and V. Dossetti, "Universality of fractal to non-fractal morphological transitions in stochastic growth processes," *arXiv: Statistical Mechanics*, 2016.

[8] I. Irurzun, P. Bergero, V. Mola, M. Cordero, J. Vicente, and E. Mola, "Dielectric breakdown in solids modeled by dbm and dla," *Chaos, Solitons & Fractals*, vol. 13, no. 6, pp. 1333–1343, 2002.

[9] B. Ranguelov, D. Goranova, V. Tonchev, and R. Yakimova, "Diffusion limited aggregation with modified local rules," *Comptes Rendus de L'Academie Bulgare des Sciences*, vol. 65, 05 2011.

Appendix

The numerical... was done using maple

Try to make this single width

Eqns, ICs

```
Eqns := {diff(vxh(t), t) = -G*M*xh(t)/Rh(t)^3,
         diff(vyh(t), t) = -G*M*yh(t)/Rh(t)^3,
         diff(xh(t), t) = vxh(t),
         diff(yh(t), t) = vyh(t)};

ICs := {vxh(0) = 0,
        vyh(0) = sqrt(G*M*(1 + e)/(a*(1 - e))),
        xh(0) = a*(1 - e),
        yh(0) = 0};
```

with rotation TIDY THIS

```
Eqns_rotation := Eqns union {diff(omega(t), t) = -G*M_sat*beta^2*(xh(t)*sin(theta(t)) - yh(
diff(theta(t), t) = omega(t)};
```

Code developed for use in taking Poincaré sections: The *poincare()* procedure, written in maple, takes inputs for *thetao, omegao*| the initial conditions for rotation and *n* the number of orbits to generate data for. *thetao* and *omegao* should be given as lists of values, for a single intitial condition lists of one item can be passed to the procedure.*poincare()* will then find each state $(\theta(t) \mathrm{mod} 2\pi, \omega(t)/\omega_H)$ where $t = nT_H$ for n from 0 to n at every combination of initial conditions $\theta(0) =$ thetao and $\omega(0) =$ omegao it recieves. The resulting data is output as an array of touples.

```
poincare := proc(thetao::list, omegao::list, n::nonnegint)
  local data, local_ICs, soln, omeg, thet, i, j, k;
  global wh, Eqns_rotation, ICs, ICs_rotation;
  data := [];
  for i in thetao do
    for j in omegao do
      local_ICs := ICs union {omega(0) = j, theta(0) = i};
      soln := dsolve({Eqns_rotation[], local_ICs[]}, numeric, maxfun = -1, range = 0 .. (n
      for k from 0 to n do
        omeg := rhs(soln(k*Th)[2])/wh;
        thet := frem(rhs(soln(k . Th)[3]), evalf(2*Pi));
        data := [op(data), [thet, omeg]];
      end do;
    end do;
  end do;
  return data;
end proc:
```