Investigation of fractal dimension in modified diffusion-limited aggregation algorithms

Candidate Number: 24669

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

April 30, 2022

Abstract

Following observation of unconventional superconductive effects in twisted bilayer graphene, the novel field of 'twistronics' has seen a rapid progression in theory and experiment. When a Van der Vaals heterostructure formed from a bilayer of 2D lattices is twisted at a so-called 'magic-angle' perturbation effects and interlayer tunneling lead to the formation of a flat electronic band at fermi level in some materials. These exotic electronic properties have the potential to be exploited as high temperature superconductors or in novel electronic devices. Using a multiorbital tight binding model (TBM) we model a bilayer of transition metal dichalcogenide 2H-$NbSe_2$ with interlayer tunneling at various twist angles. We observe ***** in the electronic band structure at *** points. Our results show potential for further modelling of the $NbSe_2$.

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, resulting 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 (e.g. electrodeposition [3]), 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 [4] to describe the growth of bacterial colonies from as a stochastic process. Further developments by Witten and Sander in 1981 [5] 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 [6], or varying geometries such as curved [7] and spherical [8] surfaces. Furthermore, Laplacian growth theory has succeeded in describing varying DLA systems along with other fractal generating systems under a single universality class [9, 10].

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 applications of DLA include: electrodeposition [3], dendrite formation [11], dielectric breakdown [12, 13], bacterial colony growth [4], biomedical imaging[7], and plant growth [14, 15]. The physical interpretation of fractal dimension as relating the growth and density of fractal and self-similar phenomena is valuable to many applications.

Modifications to specific aspects of 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 placement and quantity of initial particles [7]; altering the locations where new particles are introduced before they undergo Brownian motion [16]; introducing different species of particles with differing adherence rules [16]; or to the adhesion mechanism by altering the probability that a particle will successfully adhere to the cluster on contact [16]. The changes in the DLA cluster induced by these modifications can affect the direction of growth (e.g. producing spiral clusters) and significantly, the dimension of the cluster. Ranguelov *et al* [16] found that lowering the probability of particles adhering on contact increases the dimension of the DLA cluster. This

report aims to verify this finding by determining the fractal dimension of a classical DLA cluster, then comparing it 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 [5]. A variation on the Eden model [4], an intitial particle is placed at the origin of a two dimensional square grid of predetermined size. Three radii from the origin as seen in FIG 1 are defined: r_{max}, r_{start}, and r_{kill}. The distance of the furthest particle in the cluster from the origin r_{max} 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 grid origin. 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} and 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 the function *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_{start} 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 2×10^5 particles, hence the selected grid size is sufficient.

Some approximations are made in this algorithm to improve computational efficiency. The original model described by Witten and Sander [5] removes particles when they reach the edge of the grid. However, 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 a point on the circle 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.

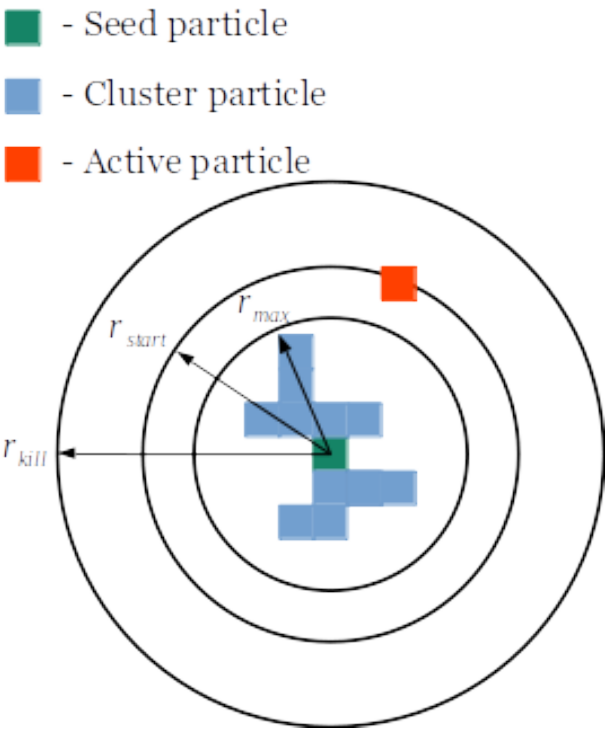


Figure 1: Diagram of the diffusion-limited aggregation (DLA) system. A particle is created at a random point on r_{start} and then undergoes a discrete random walk. If its position exceeds the distance r_{kill} it is removed. If it makes contact with the cluster it is made inactive and joins the cluster.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

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, a 'true' random number is used as the *seed* of the pseudo-random number generater for each new cluster. This allows for the efficient sampling of many psuedo-random numbers that are indepen-

dent between simulations, the resulting DLA clusters are consequently independent from one another.

Modified DLA - random adhesion

Previously we define *checkStick()* to always be *true*. However, in the modified algorithm as described by Ranguelov *et al* [16] 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: in a Euclidian space, the number of boxes N (or 'density') with side length a 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 well 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.

Two methods are employed in the derivation of d_f for individual DLA clusters: firstly a least squares fit method of the relationship described by (2) is applied to the recorded values of N_c and r_{max} of the cluster, the corresponding d_f is inferred from the fit with error; 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 for some N_c. d_f is then found at the largest N_c and corresponding r_{max} that the cluster attains. The efficacy of these methods when applied is discussed with the results of this report.

Implementation

Please see the appendix for practical details about the specific implementation of the DLA algorithm used for this report.

Results and discussion

We make the assumtion that the distribution of r_{max} and corresponding fractal dimension d_f between many randomly sampled clusters is normal (by the central limit theorem [17]) - allowing for the statistical treatment of error. Unless otherwise stated any uncertainties given will be the 95% confidence interval on the mean. Please see the appendix for examples of clusters generated.

DLA cluster fractal dimension

Using the DLA system as described we can generate 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 of N_c and r_{max} as they grow. For this dataset and the analysis in this subsection we set $P_{stick} = 1$.

Using the least squares fit method the relationship described in (2) is fit to each DLA cluster generated and the d_f are inferred. We find the mean $\overline{d_f}$ to be 1.696±0.044, where the uncertainty includes a correction for fitting error. The mean d_f from the ratio of logarithms method taken at $N_c = 20000$ on each cluster finds $\overline{d_f} = 1.733 \pm 0.004$. The validity of the ratio of logarithms result is discussed in the following section. Both measurements agree with other literature which suggest a fractal dimension of $\sim$ 1.7 in simulations [5, 7, 16] and in real physical systems such as dielectric breakdown [12, 13]. A plot of the mean $r_{max} = \overline{r_{max}}$ against N_c is shown in FIG 2, overlaid is a plot of $N_c = \overline{r_{max}}^{\overline{d_f}}$ with the $\overline{d_f}$ determined by the ratio of logarithms method.

The least squares fit method has significant drawbacks that are important to consider when interpreting these

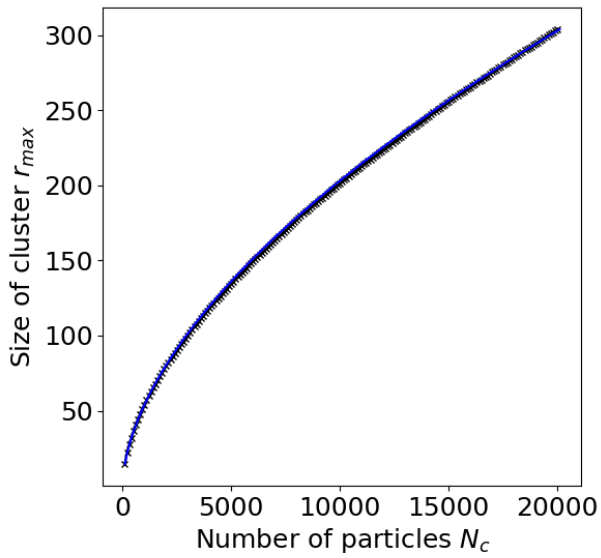


Figure 2: Mean cluster size $\overline{r_{max}}$ against particle number N_c over a sample of 100 randomly generated DLA clusters. Error bars representing the 95% confidence interval on the mean of r_{max} are comparable to symbol sizes and omitted for clarity. The mean fractal dimension $\overline{d_f}$ of the 100 cluster sample is found to be 1.733±0.004. Overlaid in blue is a plot of $N_c = \overline{r_{max}}^{\overline{d_f}}$.

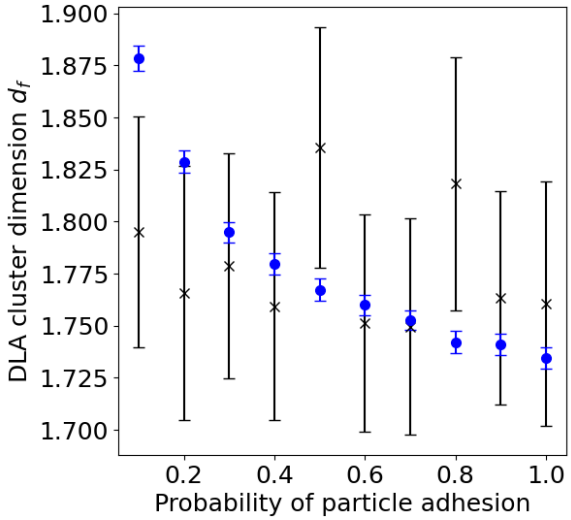


Figure 3: Derived mean fractal dimension $\overline{d_f}$ against sticking probability P_{stick}. Two derivations of $\overline{d_f}$ are shown: in blue $\overline{d_f}$ is determined by the ratio of logarithms as in (3) by taking the mean d_f at $N_c = 5000$ and corresponding r_{max} of each cluster; in black $\overline{d_f}$ is determined by fitting the relationship in (2) to each cluster and taking the mean. In both error bars represent a 95% confidence interval on the mean.

results. We make the assumtion that the relationship described in (2) holds for all scales: the dummy variables α and β are included to allow for corrections, however the least squares procedure will give weight to the data at small N_c, which is not as representative of the fractal properties that only appear at larger cluster sizes as $r_{max} >> a = 1$. There are ways of mitigating this however, e.g. for a sufficiently large cluster the fit can be applied to data that omits N_c smaller than some cutoff value. Larger clusters require significantly more computation which is why we limit our dataset to just 100 generated clusters of $N_c = 20000$. This computational cost becomes a greater issue in the following section.

A key property of fractals is their scale invariance, which our DLA model cannot reproduce for clusters of very small particle number N_c. This can however be useful in modelling physical systems which do have characteristic length scales and discrete particles such as electrodeposition[3]. Other models may also generate DLA clusters from circular particles or on non-square grids, 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 all be equivalent to the Weiner process / Brownian motion.

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 $N_c = 5000$ is generated. The mean dimension is determined as in the previous section, 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 values of $\overline{d_f}$ corresponding to each P_{stick}. Smaller clusters than in the previous dataset are generated due to the computational cost of generating larger clusters. A smaller dataset of larger clusters was created however the statistical uncertainties in the resulting $\overline{d_f}$ were too large ($\sim 20\%$) when using the least squares fit method.

Both the least squares fit method and ratio of logarithms method of deriving d_f are applied to this data, a comparison between the two can be seen in FIG 3. We can observe some striking differences in the derived $\overline{d_f}$ between the two methods: for $P_{stick} = 1$ both methods are able to reproduce within error the ratio of logarithms $\overline{d_f}$ determined in the previous section. On this dataset at $P_{stick} = 1$ the ratio of logarithms finds $\overline{d_f} = 1.735 \pm 0.005$ and the least squares fit finds $\overline{d_f} = 1.761 \pm 0.059$. Note that the least squares fit method cannot reproduce its own results within error, this harms its

validity.

From the least squares fit method (shown in black in FIG 3) we see measurements of $\overline{d_f}$ with no clear correlation, outlier values that deviate from the expected trend, and significant uncertainty. No reasonable conclusion about correlation between d_f and P_{stick} can be drawn from these results. On the other hand, the ratio of logarithms method shows a clear negative correlation between d_f and P_{stick} in line with the results of Ranguelov *et al* [16]. The validity of this method should not be taken for granted however, as it relies on the assumtion that measurements taken are in the limit $r_{max} >> a = 1$. The small clusters in this dataset have $r_{max} \sim 10^2 \times a$, which seems to be sufficient for this approximation.

An intuitive interpretation of this negative correlation is that: when a particle fails to adhere in a denser region of the cluster, its subsequent random motion is more likely to collide with another nearby cluster particle. This means denser 'branches' will grow faster than less dense regions in the cluster. This also allows particles to diffuse towards the centre of the cluster and fill in gaps, where they otherwise would have been 'captured' nearer to the starting radius r_{start} [16]. These effects overall increase the density of the generated cluster, and thus its fractal dimension.

Conclusion

By comparing the two methods of deriving d_f over varying P_{stick} and critically evaluating the results they produce, we conclude that the ratio of logarithms method is superior to the least squares fit method. The least squares fit method is unsuitable for determining d_f under the conditions we have applied it because of its large statistical uncertainty and inability to reproduce its own results between similar datasets. As such, we will disregard the results from the least squares fit method in our following conclusions. In particular we will disregard its determined fractal dimension for $P_{stick} = 1$ and the lack of correlation between $\overline{d_f}$ and P_{stick} shown in FIG 3.

Using the ratio of logarithms method we determin the fractal dimension of DLA clusters with sticking probability $P_{stick} = 1$ to be $\overline{d_f} = 1.733 \pm 0.004$. Additionally we find a negative correlation between the fractal dimension d_f and sticking probability P_{stick} in concurrence with the literature [16, 12]. This shows that simple changes to the DLA algorithm can have significant effects on the properties of the resulting clusters.

An example application of the modified probability of adhesion algorithm to a physical system is Pietronero and Wiesmann's model for dielectric breakdown [12]. In the model, they give the probability of adhesion as a function of the local electric field by some relation coefficient. Their findings also show a negative correlation between fractal dimension and adhesion probability.

Ranguelov *et al* also make modifications to the radius at which the particles are generated $r_{start}(\theta)$ by setting it as a function of 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. Their findings provide examples of further modifications that can be made to a DLA system to produce new fractals or better adapt the DLA algorithm to model natural phenomena.

Further improvements could be made to our results by sampling many more clusters, and clusters of larger sizes to reduce our statistical error. This would also justify further improvements to our implementation of the algorithm that increase computational efficiency. With improving understanding of fractals and DLA systems we can hope to better model the natural processes in which they are seen.

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] Y. H. Shaikh, J. Pathan, A. Jallel, and A. R. Khan, "Nickel dendritic pattern in electrodeposition at different temperatures," *ES Materials & Manufacturing*, vol. 15, pp. 78–84, 2022.

[4] 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.

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

[6] R. C. Ball, M. Nauenberg, and T. A. Witten, "Diffusion-controlled aggregation in the contin-

uum approximation," *Physical Review A*, vol. 29, pp. 2017–2020, 1984.

[7] J. Choi, D. Crowdy, and M. Z. Bazant, "Diffusion-limited aggregation on curved surfaces," *EPL (Europhysics Letters)*, vol. 91, p. 46005, aug 2010.

[8] 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.

[9] 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.

[10] 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.

[11] D. Li, Z. Zheng, Z. Shui, M. Long, J. Yu, K. W. Wong, L. Yang, □, L. Zhang, and W.-M. Lau, "Using elemental se and ag to grow pure ag2se dendrites/dendritic-films of highly oriented (001) nanocrystals," *Journal of Physical Chemistry C*, vol. 112, pp. 2845–2850, 2008.

[12] L. Pietronero and H. Wiesmann, "Stochastic model for dielectric breakdown.," *J Stat Phys*, vol. 36, p. 909–916, 1984.

[13] 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.

[14] B. B. Mandelbrot, *The Fractal Geometry of Nature*. Benoit B. Mandelbrot, 1 ed., 1982.

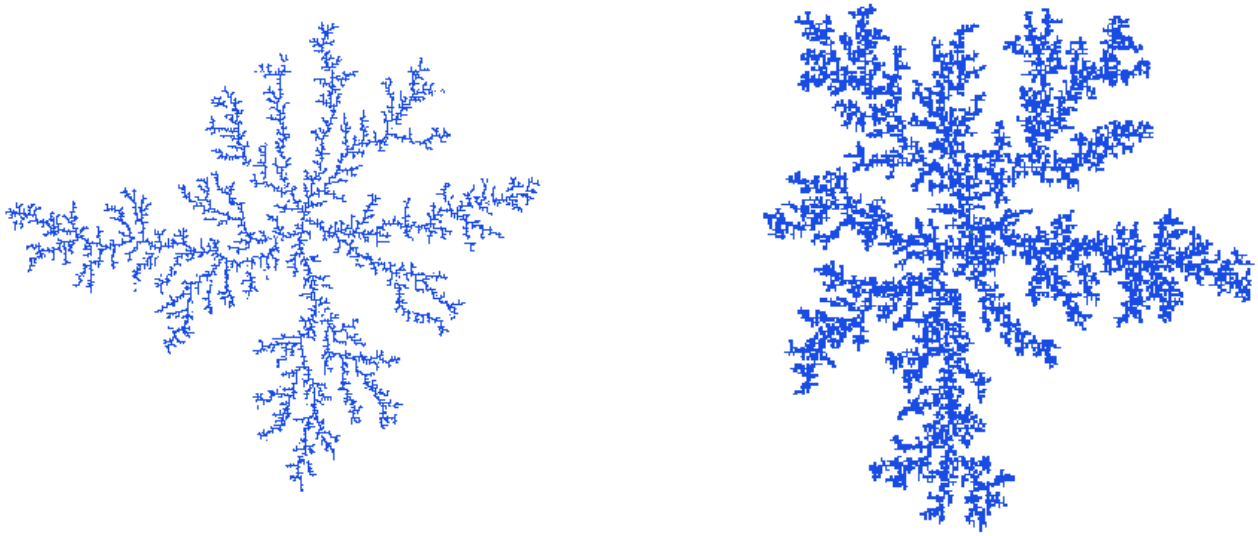
[15] A. Zhang, Dongsheng; Samal and J. R. Brandle, "A method for estimating fractal dimension of tree crowns from digital images," *CSE Journal Articles*, vol. 97, 2007.

[16] 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.

[17] E. V. Sadovnik, "A central limit theorem for fluctuations of internal diffusion-limited aggregation with multiple sources urop + final paper 2016," 2016.

Appendix

Example DLA clusters



(a) $P_{stick} = 1$ (b) $P_{stick} = 0.1$

Figure 4: Example DLA clusters generated to $N_c = 10000$.

Confidence intervals

For the mean quantities in this report error is presented as the 95% confidence interval on the mean given by

$$a = 1.96 \frac{1}{\sqrt{n}} \sigma_x \tag{4}$$

For some mean quantity $\bar{x}$ where σ_x is the standard deviation on the mean and n is the number of samples. The confidence interval is then $[\bar{x} - a, \bar{x} + 1]$. This can be interpreted as the interval such that there is a 95% chance the interval contains the true population mean of $\bar{x}$.

Implementation of algorithm

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'.

Note: when compiling and running the C++ code provided, the makefile in 'source' has been edited. To compile the code place the .cpp and .h files into a directory called 'source' and add in the missing/unedited files (Particle.h, Window.h, Window.cpp). Run 'make -B -C ./source/' from the directory above 'source'. The compiled binary and the data it produces are in this directory, execute the binary using './run'. The python data analysis file 'DLA.py' should also be run from this same directory

The changes made to the C++ code given by Dr A. Souslov and Dr D. Tsang are spread throughout the source code provided (i.e: CSVWrite.h, DLASystem.cpp, DLASystem.h, mainDLA.cpp, rnd.h, Makefile). Most of the changes made are used when pressing 't' in the simulation window to set parameters for data recording and P_{stick} etc. Some new functions have been added to the DLASystem class, and a new class CSVWrite is written for data recording. Some select functions are presented here but you would do best to look at the source code with comments.

```
// check if the last particle should stick (to a neighbour)
int DLASystem::checkStick() {
        Particle *lastP = particleList[numParticles - 1];
        int result = 0;
        // loop over neighbours
        for (int i = 0; i < 4; i++) {
                double checkpos[2];
                setPosNeighbour(checkpos, lastP->pos, i);
                // if the neighbour is occupied...
                if (readGrid(checkpos) == 1)
      if(randomStick){
              double rr = static_cast<double>(rgen.randomInt(1000));
        if (static_cast<double>(rr/1000.0) < stickChance){result = 1;}
        else {result = 0;}
      }
      else {result = 1;}
        }
        return result;
}

void DLASystem::recordData(int numOfSims, int nInterval_, pair<int, int> nRange_) {
  if(running != 0 && !recording){cout << "please pause simulation before recording data" <<
  else{
    if (numOfSims >= 2){
      cout << "Starting DLA multiple recording" << endl;
      simNum = numOfSims;
      numSims = numOfSims;
    }else cout << "Starting DLA recording" << endl;

    recording = true;

    nInterval = nInterval_;
    nRange = nRange_;
  }
}

//record data on update
void DLASystem::updateRecording() {
  if(numParticles == nRange.second){
    //cout << "recording data at N = " << numParticles << endl;
    numData->push_back(static_cast<double>(numParticles));
    radiusData->push_back(clusterRadius);

    if (simNum == numSims) {dataSet->push_back(*numData);} //only need one column of this

    //when finished in range write data to dataSet \& reset sim
    if (simNum >= 1){
      dataSet->push_back(*radiusData);
      cout << "Sucessfully recorded sim number " << simNum << " data in range (" << nRange.

      //reset initial conditions
      Reset(); //will reset numData and radiusData
      setRandomSeed();
      setFast();

      if (simStickNum != 0 && simNum \% simStickNum == 1 && simNum != numSims){
        if (stickDiff != 0){
          stickChance -= stickDiff;
```

```
            if(stickChance<=0){
              cout << "sticking chance is now 0, finishing recording" << endl;
              writeDataCSV();
              recording = false;
              pauseRunning();
            }
            else cout << "new sticking chance set to " << stickChance << endl;
          }
        }

        setRunning();

        if (simNum == 1) {writeDataCSV();}

        simNum--;
        //cout << "conditions reset for simulation number " << simNum << endl;
      }

      if (simNum <= 0) {recording = false; pauseRunning();}
    }
    else if(numParticles > nRange.first
            && numParticles % nInterval == 0
            && numParticles > numData->size() * nInterval + nRange.first){
      //cout << "recording data at N = " << numParticles << endl;
      numData->push_back(static_cast<double>(numParticles)); //dont really need to do this...
      radiusData->push_back(clusterRadius);
    }
}

//write recorded data to a CSV file
void DLASystem::writeDataCSV(){
  auto csv = new CSVWrite("./data.csv");

  csv->WriteVector(dataSet);
  csv->CSVClose();

  //clearDataSet();

  recording = false; //should be false anyways
}
```