NTNU – Trondheim
Norwegian University of
Science and Technology

DLA based agglomeration of aerosols

Simen Berntsen

November 9, 2015

Specialization project applied physics
Department of Physics
Norwegian University of Science and Technology (NTNU).

Supervisor: Alex Hansen.

Contents

1 Introduction

This is the section where the introduction to the project will be written. For now, please just read the notes:

This project has been on taking a Diffusion-Limited Aggregation (DLA) approach to the clumping of aerosols in the lower atmosphere in relation to emission from fossil fuelled based combustion (etc.). Diffusion-Limited Aggregation is an algorithmic approach to growth of structures where limited supply of mass is the bottle-neck. A number of different growth models have been proposed, however the first one was proposed by T. A. Witten Jr. and L. M. Sanders [4]. This model builds on the Eden model, first proposed by Eden [1], but with a random walker determining the aggregation. Write something about Diffusion Limited Cluster Aggregation here!

2 Theory

2.1 Fractal dimension

To compare the DLA simulations to experimentally observed clusters, one studied and compared the density correlations (and fractal dimensions). The general expression for correlations are

$$C(\mathbf{r}) = \langle \rho(\mathbf{r'})\rho(\mathbf{r'}+\mathbf{r})\rangle. \tag{1}$$

In equation (1), $C(\mathbf{r})$ is the correlation function, ρ is the density and $\mathbf{r}$ and $\mathbf{r'}$ are points in space. For particle clusters such as those generated by DLA, one defines this density to be $\rho(\mathbf{r}) = 1$ for occupied points, and 0 otherwise. An approximation of the ensemble average of the density correlation function which is valid for $r_{cluster} \gg r_{particle}$ is given as (from [4])

$$C(\mathbf{r}) = \frac{1}{N}\sum_{\mathbf{r'}} \rho(\mathbf{r'})\rho(\mathbf{r'}+\mathbf{r}). \tag{2}$$

In equation (2), N is the number of particles in the cluster. All other quantities are as in (1). It can then be shown that $C(r)$ has the scaling relation Some papers just state that the general scaling relation is assumed to be of the form... should look into this more closely

$$C(r) \sim r^{\alpha}. \tag{3}$$

Moreover, the mass of the cluster scales like

$$M(r) \sim r^{d_f}, \tag{4}$$

so that

$$C(r) = \frac{dM(r)}{2\pi r dr} \sim \frac{r^{d_f-1}}{2\pi r dr} \sim r^{d_f-2}. \tag{5}$$

from equation (5), one may generalize the equation by replacing the 2 in the last exponent with d (the euclidean dimension), so that comparison of exponents yields

$$d_f = d - \alpha. \tag{6}$$

??? should include how to calculate the fractal dimension from density correlation function!

An alternative way of calculating the fractal dimension is through the radius of gyration R_g. This stems from the scaling relation

$$N = k_0 \left(\frac{R_g}{a} \right)^{d_f}, \tag{7}$$

where N is the number of particles in the cluster, k_0 is the fractal prefactor (also called structural coefficient), a is the diameter of the particles, and d_f is the fractal dimension. In equation (7), R_g is defined as

$$R_g = \frac{1}{N} \sum_{i=1}^{N} r_i, \tag{8}$$

where r_i is the distance from particle number i to the centre of mass R_{CM}. Mention the substitution $M \leftrightarrow N$ because of $m_i = 1$.

2.2 Family-Vicsek scaling

A scaling relation for the cluster size distribution $n_s(t)$ has been found by F. Family and T. Vicsek [3]. ??? Write about scaling relation, and how this is used to analyze moving clusters.

3 Method

The fundamental algorithm of DLA, as described by Witten & Sanders [4] is as follows; a seed particle (the particle we follow) is placed at the centre. This particle will be the basis of the growing cluster. Another particle is then introduced at some distance from this seeding particle. The starting position of this particle can in principle be infinitely far away from the particle, but for practical matters, it is often placed close to the seeding particle. This particle will perform a random walk in the space around the seeding particle, with the intention that at some point in time, the walking particle will hit (i.e. be sufficiently close to) the seeding particle, and the two will clump together.

All of the following algorithms have been implemented using C++ 14. ??? Don't know about the following part, but: The code was run on OS X Yosemite 2,7 GHz Intel Core i5 8 GB 1867 MHz DDR3 system.

3.1 On-lattice DLA

For the on-lattice case, the algortihm is pretty straight forward, as is described in the beginning of section 3. Thus it will only briefly be discussed here, with the main focus being on the off-lattice version. Special for the on-lattice case is that the particle can only move in discrete steps and directions. That is, it may only move along the axis (either north, south, east or west), but not diagonally. More specific for the simulation run in this project is that the walking particles would always start at a random (uniformly distributed) point on the perimeter of the grid. If it was to take a step causing it to move outside of the grid, it would restart somewhere else on the perimeter. Although this cannot be done without some loss of generality, it is still done to increase the simulation speed. ??? should look into this more?

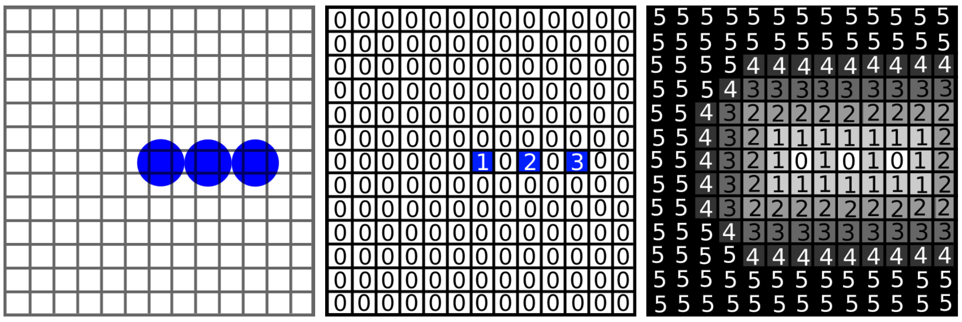
Since the on-lattice simulation is of less interest compared to the off-lattice case, only a crude on-lattice simulation was performed. The results of the on-lattice simulations can be found in ??? Link to section with results for on-lattice DLA simulations.

3.2 Off-lattice DLA

In off-lattice DLA simulations, the particle is no longer restricted to move along any axis. This means it can move freely in an approximately continous space, and any direction. This makes the algorithm is more advanced than the on-lattice one. The algorithm used for the off-lattice DLA simulation is based on the one suggested by Kuijpers et. al. [2]. In this algorithm, one uses an on-lattice approximation for storing the positions of the particles in the cluster, thus being more RAM dependant than other algorithms, but with the benefit of reducing the calculation of sites viable for hits. Like in section 3.1, the particle starts along a circle of radius R_{start} from the cluster, and will be killed if the wander too far from the cluster. This limit is set to $R_{kill} = 5R_{cluster}$, where $R_{cluster}$ is the radius of the smallest circle spanning the cluster.

The concept is as follows: The program uses three arrays (illustrated in figure 3), which together keeps track of all information of interest. The first one, $\mathbf{A}$, is an $N \times 2$ array, which only keeps track of the exact coordinates of the particle. These coordinates are then mapped to correspond to integer values, so that they may correspond to a cell number in $\mathbf{B}$, an $n \times n$ (n is lattice size) array in which all the elements are either 0 or the label of the particle. See figure 1a and 1b for an illustration of this. In this context, label refers to the number given to the particle when it attaches to the cluster, starting from the original seed particle. The final array, $\mathbf{C}$, stores the distance from each cell to a particle, as shown in figure 1c. This is done in order to keep track of the distance from a walking particle to a point on the cluster. The great benefit of this algorithm lies in being able to take steps of varying length, depending on the distance from the cluster, which is only possible without loss of generality if the particle is far away from the cluster. For simplicity, there is a set a maximum distance D_{max} to be the maximum distance cap. This reduces the amount of number of calculations required per iteration.

Throughout the process, the walking particle will take steps depending on its distance from the cluster. The following three criterias determines the different regimes the walker may be in with respect to step size. It is worth noting that L_{min} is the step size taken by the particle corresponding to Brownian motion when the particle is close to the cluster. This value is set before the simulation starts. What regime the particle is in depends on wheter the approximated distance between the walker and the cluster, d_{wc}, is so small that there is a chance of hitting the cluster if the particle takes a step of


(a) Illustration of the real situation, where the particles for simplicity allign nicely along the horizontal axis.

(b) Illustration of how the labelling of the cluster was done.

(c) Illustration of the rounded off distances at each point in space.

Figure 1: Illustration of a dummy configuration of particles, and how each array work. The seed particle is in all cases placed in a 13×13 array, at position (7,7). For simplicity, the other particles are not random walkers in this dummy case, but move in from right to left, along the horizontal axis.

length L_{min}. The distance d_{wc} can be found by looking in the cell labeled with the rounded coordinates of the walker in **C**. In addition, one must take into account that what is stored is the coordinates of the centre of the particles, meaning that there must be a distance of $2r_p$ between them at least. Since one looks at the rounded off coordinates in when finding d_{wc}, one adds a 1 to make sure that the region includes all possibilities of hits.

$d_{wc} \leq 2r_p + L_{min} + 1$: The particle is within hitting range of the cluster. That may be a single particle or lots of particles, and there has to be done an exact calculation for determining if the particle will hit for all of the particles within the range of d_{wc}. This is done by looking at all particles stored in **B** within a square of size $2(2r_p + L_{min} + 1) + 1$ centered around the cell with indices given by the rounded coordinates of the walking particle. All non-zero elements within this square gives the location of the coordinates in **A**, so they can be extracted. Then the calculation of is the particle hitting is performed for all particles within hitting range, and the one yielding the lowest positive step required will be the one the particle latches on to.

$2r_p + L_{min} + 1 \leq d_{wc} \leq D_{max}$: The particle is in an intermediate position, meaning it can only move a distance of $d_{wc} - (2r_p + L_{min} + 1)$ without having to check for a possibility of a hit.

$d_{wc} = D_{max}$: The particle is far away from the cluster, allowing it to move a distance of $\max(d_{w0} - R_{max} - (2r_p + L_{min} + 1), D_{max} - (2r_p + L_{min} + 1))$.

The particle will attach to the cluster iff the step performed is shorter than L_{min}. Then **A**,**B** and **C** are updated, and a new particle starts walking. A new walker is launched from the starting distance R_{start}, and this process is repeated until the cluster contains the desired amount of particles.

From the aggregated cluster, the fractal dimension d_f was calculated using a fit of the data to equation (7).

3.3 Diffusion limited cluster aggregation

The algortihm for Diffusion limited cluster aggregation has been developed for this project. It is inspired by the concepts of Kuijpers et. al. [2]. However, the speed up in the off-lattice DLA algorithm has NOT YET been implemented, as it is not as straight forward to keep track of distances between clusters and the orientation of clusters/particles. This is also due to the fact that all clusters will perform a random walk throughout the simulation.

So to begin with, one starts out with a density of interest and distributes this over a domain. These are distributed randomly (uniformly at the moment) within a given domain size. For simplicity, one uses periodical boundary conditions. The positions of the particles are stored in a dynamical array *clusters*, as seen in figure 2 and 3c. The labels of the clusters correspond to the column number of *clusters*, and are stored in the enumerated grid *num_grid* (see figure 3b).

This storage system allows for easy checks to see if a particle has any neighbours (at the moment only nearest neighbours), which may allow for a hit. In the event that a hit happens, the two clusters will join together and form a new cluster. In this case both arrays has to be updated, making sure the labelling of the new cluster is correct. For conventional purposes, the new cluster will always have the label as the lowest cluster comming in. This means if cluster 2 collides with cluster 6, the new cluster will keep label 2 while 6 is erased. All clusters with label higher than 6 will then have their label reduced by 1. After some while, one might have a situation similar to that shown in figure 3a.

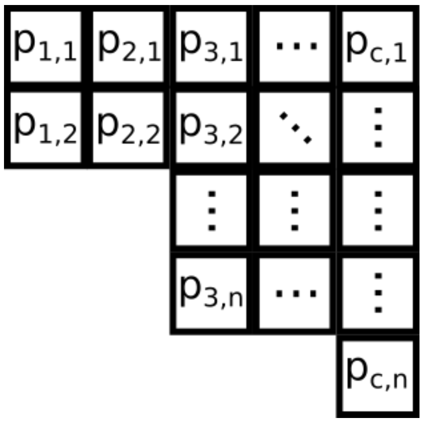
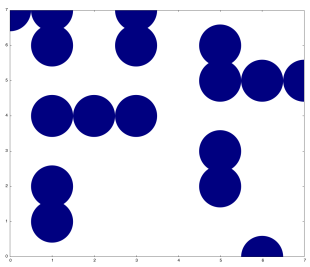


Figure 2: Schematic of the structure of the clusters array. Note that the dimension is not $N \times N$, but is rather dynamic int the sense that neither axis are static. The subscript c indicates the label for the cluster, while n denotes the particle number within a cluster c.

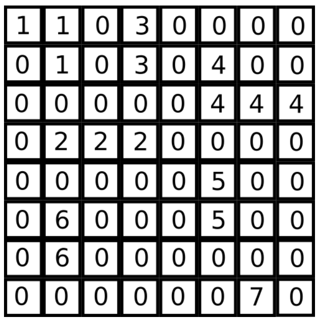
4 Results

The results for the on-lattice simulations can be found in figure 4. ??? The fractal dimension was calculated straight forward in this case, only looking at the relation between R and N, not using equation (7). Should probalby also update the figure with a figure of more particles, so just get it done some weekend. Should probably also fix the calculation of d_f. It is worth noting that the fractal dimension oobtained from 4 is in accordance with what was found in [4], although the method of calculation was slightly different so an exact accordance is not to be expected.

The results for the off-lattice simulations can be found in figure 5, with corresponding regression plot in figure 6 (see equation (7)). These results fits with the results obtained by Kuijpers [2].

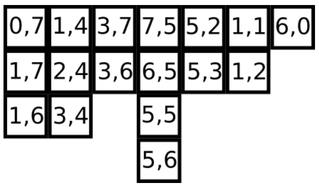


(a) Example of how a configuration would look.



1	1	0	3	0	0	0	0
0	1	0	3	0	4	0	0
0	0	0	0	0	4	4	4
0	2	2	2	0	0	0	0
0	0	0	0	0	5	0	0
0	6	0	0	0	5	0	0
0	6	0	0	0	0	0	0
0	0	0	0	0	0	7	0

(b) Example of how the enumerated grid would look.



0,7	1,4	3,7	7,5	5,2	1,1	6,0
1,7	2,4	3,6	6,5	5,3	1,2	
1,6	3,4		5,5			
			5,6			

(c) Example of how the clusters array would look.

Figure 3: Example of how the data structures would look.

A References

[1] Eden, M. A two-dimensional growth process. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 4: Contributions to Biology and Problems of Medicine* (Berkeley, Calif., 1961), University of California Press, pp. 223–239.

[2] Kuijpers, K. R., de Martín, L., and van Ommen, J. R. Optimizing off-lattice diffusion-limited aggregation. *Computer Physics Communications 185*, 3 (2014), 841 – 846.

[3] Vicsek, T., and Family, F. Dynamic scaling for aggregation of clusters. *Phys. Rev. Lett. 52* (May 1984), 1669–1672.

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

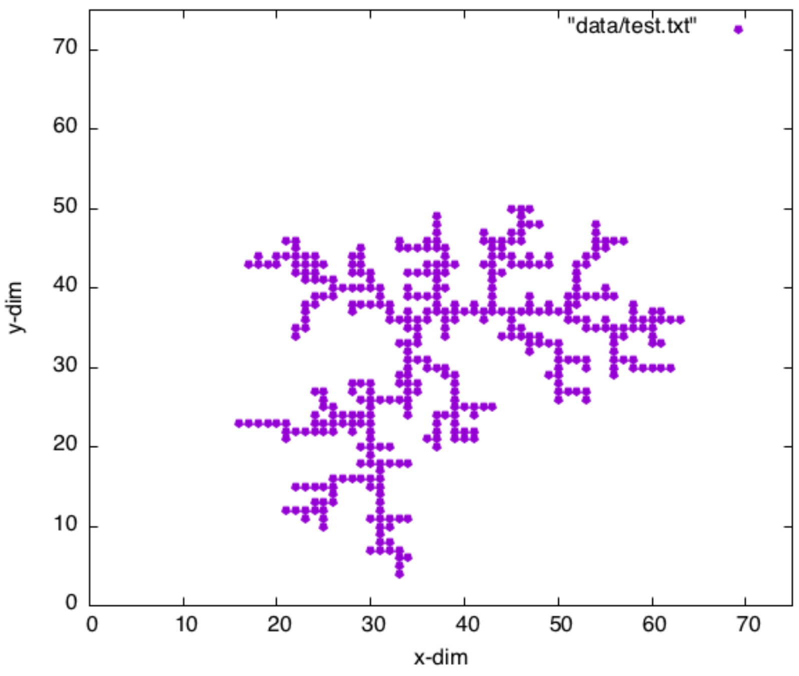


Figure 4: ??? I'm sure there could be some improvements to this. On-lattice DLA simulation of 400 particles on a 75 by 75 grid. This particular cluster has a fractal dimension of $d_f = 1.70999$.

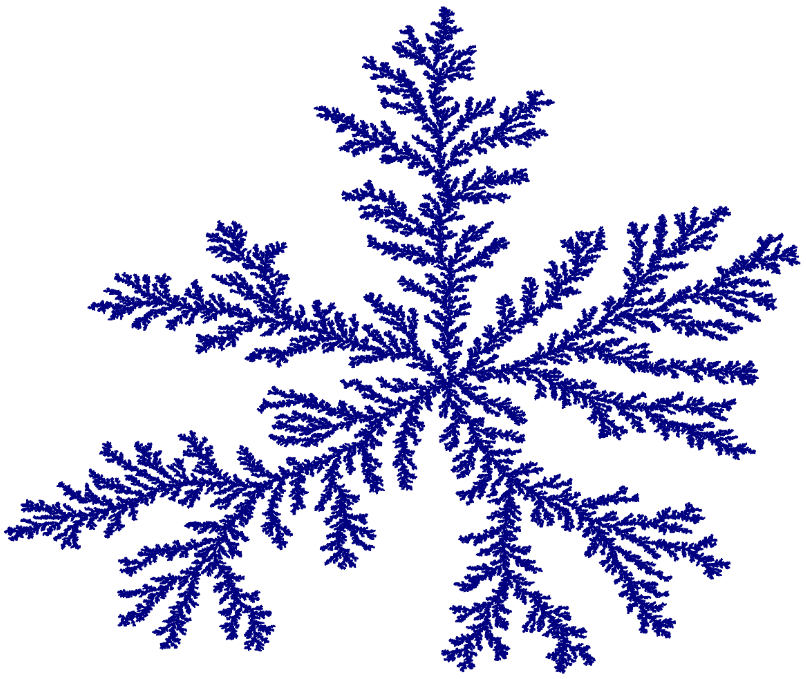


Figure 5: ??? Replace with a borderless version, so that one doesn't have to have x-ticks and x-labels etc... Off-lattice DLA simulation of 10^6 particles. This particular cluster has a fractal dimension of $d_f = 1.7248 \pm 0.006033$.

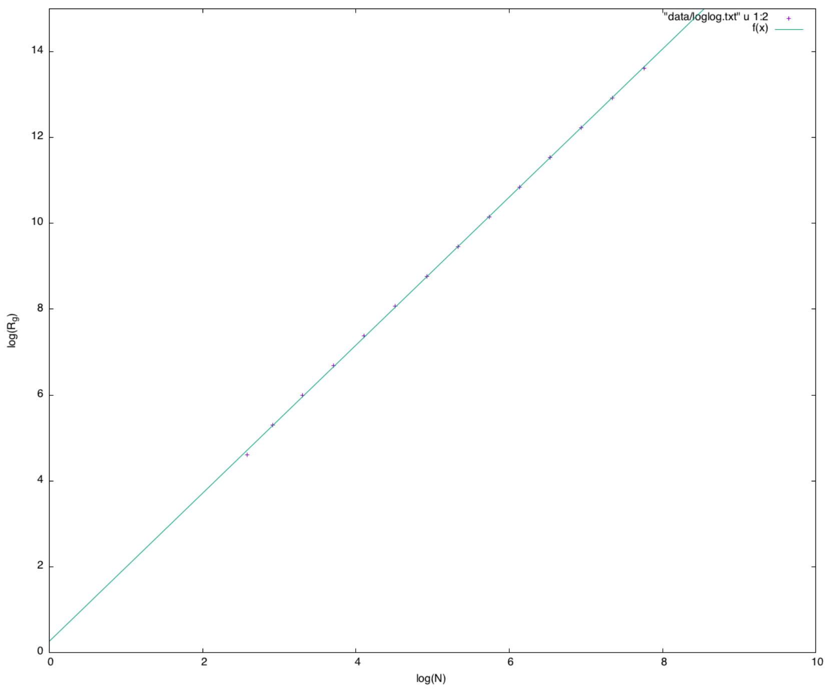


Figure 6: The log-log plot of N and R_g. From this, the fractal dimension was calculated using linear regression. ??? replace with a plot including the slope/linear approximation.