Fast simulation of diffusion-limited aggregation (DLA), and the shielding effect

ilangkay[*]

National Institute of Physics, University of the Philippines Diliman, Philippines

[*]Corresponding author: amsurname@university.edu

Abstract

Although the exact forces and mechanisms may differ, low-density systems whose particle transport is driven by diffusion generally appear similar such as frost patterns and river networks. The qualitative aspects of such systems can be modeled with diffusion-limited aggregation (DLA) which produces fractal structures from randomly walking particles attaching to a cluster. In this paper, I write programs to simulate a randomly walking particle, and DLA using a method that reduces simulation times. It was found that particles are most likely to attach onto the outer tips of a cluster, which then shields its inner portions and thus producing the characteristic tips, branches and valleys. This shielding effect is responsible for the similarities in appearance of the diffusion-dominated systems.

Keywords: random walk, diffusion-limited aggregation

1 Introduction

Though governed by different forces and mechanisms, frost patterns on glass, electrodeposition patterns, mineral veins, and river networks generally look similar. As one might expect, these systems belong to a broad class in which diffusion primarily drives the transport of particles [1]. Under low-density conditions, the qualitative aspects of such systems can be modeled with diffusion-limited aggregation (DLA).

In 1981, Witten and Sandler proposed DLA as a process describing the irreversible aggregation of tiny solid particles [2]. In this model, they considered a seed fixed within a lattice and introduced particles one at a time to randomly walk inside the lattice until they attach to the growing cluster formed by the seed and deposited particles. The result is an fascinating fractal structure described as having multiple outward tips that shield inner surfaces from more particles attaching onto them. For interested readers, Halsey (2000) further explores the mathematical theory behind this structure and other related advances [1].

In this paper, I write programs to simulate a randomly walking particle, and diffusion-limited aggregation using an method that reduces simulation times. I also discuss the shielding effect that produces the characteristic results of DLA and is likely at work in physical systems.

2 Methodology

In both programs, I represented a square lattice of side length n by an $n \times n$ numpy array as this data structure matches the grid structure of lattices and offers convenient indexing and storage. Lattice sites visited by a randomly walking particle or currently occupied by a deposited particle are signified by nonzero entries in this array. When a simulation has finished, the resulting array is visualized using matplotlib's `imshow` function.

A randomly walking particle is simulated in a step-wise fashion within a loop until it has walked a given number of steps—Figure 1 shows a flowchart of the program logic. A particle is first put at a random site

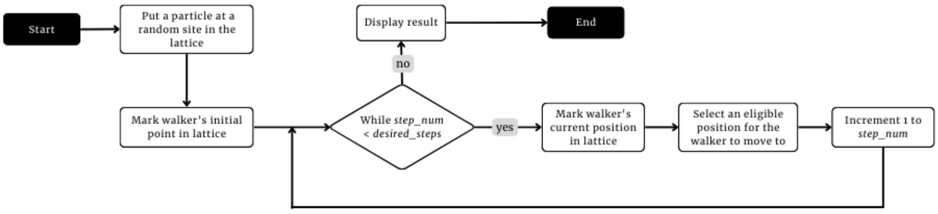

Start
Put a particle at a random site in the lattice
Display result
End
no
Mark walker's initial point in lattice
While *step_num* < *desired_steps*
yes
Mark walker's current position in lattice
Select an eligible position for the walker to move to
Increment 1 to *step_num*

Figure 1: Flowchart, created with canva.com, for the random-walk program.

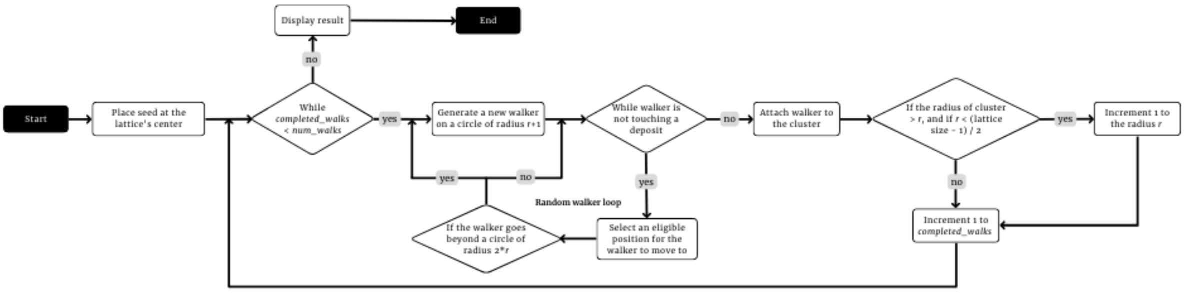
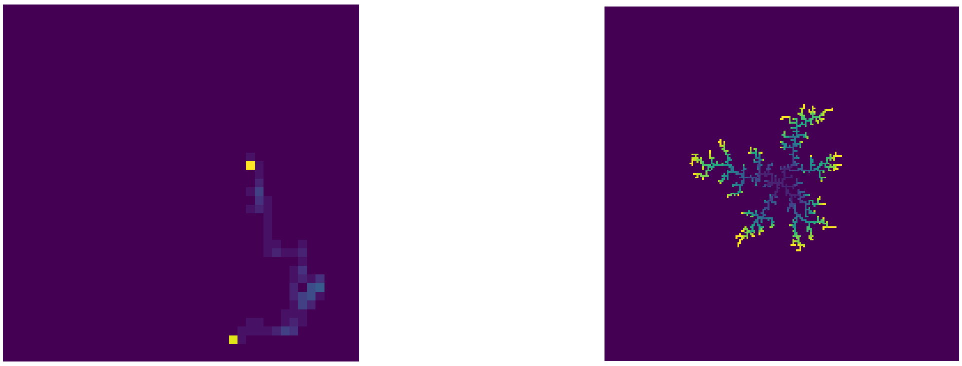

Display result
End
no
Start
Place seed at the lattice's center
While completed_walks < num_walks
yes
Generate a new walker on a circle of radius r+1
While walker is not touching a deposit
no
Attach walker to the cluster
If the radius of cluster > r, and if r < (lattice size - 1) / 2
yes
Increment 1 to the radius r
yes
no
Random walker loop
yes
no
If the walker goes beyond a circle of radius 2*r
Select an eligible position for the walker to move to
Increment 1 to completed_walks

Figure 2: Flowchart, created with canva.com, for the diffusion-limited aggregation program.



(a) Random-walk. The yellow dots indicate the walker's first and last position. Grid points visited often become lighter in color.

(b) DLA cluster in a 201 × 201 lattice with 1001 particles. Color indicates time of arrival with the yellow particles having attached most recently.

Figure 3: Simulation results.

within the lattice. Then, adjacent sites are chosen for the particle's new positions, whose corresponding array entries will be incremented by a small positive amount. At the boundary, the particle's movement is simply restricted, so that it does not go out of bounds. The particle's initial and final positions are marked with relatively large numbers in the array to distinguish them.

The diffusion-limited aggregation program simulated a given number of particles in the lattice, which roamed about until they attached themselves to a deposited point such as the seed initially placed at the lattice's center. To decrease simulation times, I took a suggestion in Exercise 10.13 of Newman's *Computational Physics*, which is made available online by its author [3]. The idea is to place new particles just outside a growing circle that bounds the existing cluster, and replace them if they stray beyond a larger growing circle [4]. Figure 2 shows a flowchart of the program logic which implements this method. The array entries corresponding to the deposited points are not incremented by a constant amount, but rather in such a way that the sequence in which they become attached can be visualized.

Each site adjacent to a particle's current position is equally likely to be chosen for the particle's new position in both programs.

3 Results and discussion

The random-walking program is successfully. Figure 3a shows a single simulation of one randomly walking particle executing 100 steps. Noticeably, this particular result has the particle's initial and final points (which are marked yellow) are far apart, and several grid points were visited multiple times (as marked by lighter hues of blue). The former is notable because particle's expected final position (as calculated from probability theory) should be near its initial location, given that the probabilities of moving along the available directions is the same. The latter suggests a different avenue of study, on random paths that avoid intersecting themselves, although this condition is not likely to be observed in nature.

The modified diffusion-limited aggregation program is also successful as it delivers the expected results and finishes comparatively quickly (without the modifications, I had a simulation that finished in half and hour, while the modified program finishes in under five minutes). Although, Offringa notes that these simplifications slightly change the space of possible final states because certain positions become more difficult for particles to reach as they may be removed for moving too far from the cluster [4].

The simulation particles executing a random walk (also known as Brownian motion) is a good model for the motion of real particles because the many fluctuations and collisions occurring in liquids and gases do cause individual particles to move in random directions with equal likelihood [5].

Figure 3b shows an example simulation in which one observes that many of the latest attachments to the cluster occurs at its tips as indicated by the yellow grid points. It was observed that as the cluster grows the rate at which it grows also increases, especially the cluster's outermost structures because the surface area to which new particles attach to also quickly grows. The structure formed by DLA demonstrates the shielding effect of the fast growing tips of teh cluster. Eventually "branches" will grow out of this tips and then shield the resulting inner surfaces which also manifests as empty valleys within the cluster. So, the shielding effect is responsible for the self-similar nature of the structure as it continues however large the cluster becomes [1].

In physical systems, one can also reasonably imagine that this shielding effect comes into play as the outer structures tend to be easier to reach and thus they grow fastest; this explains the similarity between the results of DLA and the appearances of low-density systems whose transport is dominated by diffusion. One such example occurs with the growth of crystals from electrodeposition processes: the movement of ions in solution is determined by diffusion [6]. (A photo gallery by the University of Michigan of such crystals is linked in [7].)

4 Conclusions

The random-walking and diffusion-limited aggregation programs executed successfully, with the method used in DLA program cutting simulation times by a large amount. The DLA-simulation result showed the that the latest particles tend to attach to the aggregate at its outermost tips rather than its inner surface. This shielding effect produces the characteristic tips, branches, and valleys one also sees in low-density systems whose transport is dominated by diffusion such as electrodeposition. Hence, the similarities in appearance between the DLA results and these systems are explained.

5 Reflection

From this programming challenge, I realized that data structures such as lists, numpy arrays, and others are the skeleton of a program. Well-chosen data structures facilitate development by preventing many headaches from occurring as it provides convenient interfaces, storage, and retrieval. In the two programs I developed, using a numpy array to represent the square lattice, and the array's entries for the path traversed by a randomly walking particle or the location of a deposited particle was a good choice. As I could easily retrieve the state of adjacent sites in the array using code constructions such as

```
directions = [[1, 0], [-1, 0], [0, 1], [0, -1]]
for direction in directions:
    print(
        lattice[tuple(current pos + directions)]
    )
```

where `lattice` is a numpy array. Using a plain Python list of lists would be difficult comparatively, at least in my approach.

Moreover, this challenge highlight the need to be familiar with the programming language's quirks and most important packages or modules. Python has many tricks and conveniences that have propelled it to becoming one of the most used languages globally, such as list comprehensions and another example is found in how the for loop is written above; both used in my programs. However, it is my familarity with numpy that decreased the difficulty somewhat.

For the diffusion-limited aggregation program, the long amount of time before the simulation completed became a problem. I initially wrote down a bare-bones version where the particle can be generated anywhere and roam as long as it liked before attaching to the aggregate. This arises because simply the empty regions of the lattice is initially large compared to the seed (and even the cluster for small particle numbers). This is the case where physical intuition can guide programming tricks: if a particle moves too far, it becomes very unlikely to attach itself, and we might as well consider different closer particle instead; and the probability of a particle become attached scales with the surface area of the cluster.

To construct these programs, I used first tried to write down and manually simulate the various actions that would have to be done. This allowed me to plan, preempt edge cases, and group together actions into a task which I can translate into functions in the code. In the DLA program I defined a function possible_directions which is only called in two other functions, and not directly in the program loop itself (so I did not show it in the flowchart). I noticed that the two tasks required the same piece of code as the possible directions for the particle's new positions is required to choose the particle's next position and to check if the particle is touching the cluster.

Additionally, while I do not know if it is a good programming pattern, I found a way to indirectly modify a list that I am iterating over by defining a new array and appending the desired elements to it. Finally, persistence is the key to doing such challenges. I had a difficult time of actually getting the nearest grid point on the bounding circle for the initial spawn of a particle; it required many tries of trial and error and going back to debug.

References

[1] T. C. Halsey, Diffusion limited aggregation: A model for pattern formation, **53**, 36 (2000), ISSN 0031-9228.

[2] T. A. Witten and L. M. Sander, Diffusion-limited aggregation, a kinetic critical phenomenon, **47**, 1400 (1981), publisher: American Physical Society.

[3] M. Newman, *Computational Physics* (Createspace Independent Pub, USA, 2013), ISBN 978-1-4801-4551-1, https://books.google.com.ph/books?id=SS6uNAEACAAJ.

[4] A. Offringa, Diffusion limited aggregation, https://www.astro.rug.nl/~offringa/Diffusion%20Limited%20Aggregation.pdf.

[5] Encyclopedia Britannica, Brownian motion | physics | britannica, https://www.britannica.com/science/Brownian-motion.

[6] T. C. Halsey and M. Leibig, Electrodeposition and diffusion-limited aggregation, **92**, 3756 (1990), ISSN 0021-9606.

[7] UniversityofMichigan, Experiment picture library - thumbnails, https://sharepoint.umich.edu/lsa/physics/demolab/experiment%20picture%20library/forms/thumbnails.aspx.