A computational study of sequential deposition: A dynamic monte carlo process in statistical physics.

Indranil Ghosh* and Budhaditya Banerjee

Department of Physics, Jadavpur University, Kolkata - 700032

1. Abstract

Few of the many enthralling applications of dynamic monte carlo process is the study of crystallization of hard spheres due to increase in pressure or the response of Ising model due to an external magnetic field switched on at an initial time, where physical time plays an integral role. In our presentation, we study a phenomenon where particles (discs) are randomly attached to a 2-dimensional surface and stay attached only if they find a required free space. With time, the number of particles that get repelled by other particles, due to not enough free spacing, increases and takes longer time to fill up the surface. We study two algorithms, a naive and then a faster-than-clock algorithm, to simulate this phenomenon of particle saturation, and compare their performances in details. We carry out the simulation with Pygame, a python package. We also calculate the approximate value of the maximum coverage numerically, for these circular particles for different radii after the simulation is carried out for a certain recommended time, with our computer programs. At last we end by investigating how Pygame can be used to carry out simulations of various monte carlo methods in algorithmic statistical physics.

2. Dynamical Monte Carlo

Numerical studies corresponding to finding out the equilibrium close to the phase transion in Ising Model with the help of Monte Carlo methods can be *a priori* difficult, as the equilibrium configuration is never reached during simulation. So, Dynamical Monte Carlo Method is used where the system evolves with time from one local relaxation point to another. One is mostly focused on the numerical study of the balanced configuration from a given initial configuration. In Equilibrium Monte Carlo, we have unbounded choice of configurartions manifested by the *a priori* probability, whereas in the Dynamic Monte Carlo simulations the *a priori* probability is fixed, which provides them with more delicacy and simplicity.

3. Naive Random Deposition

On a 2D surface (initially a square), spherical disks are dropped randomly one after the aonther. This simulation and its analysis is congruent to the irreversible adsorption of spherical particles. Disks remain stuck to their positions where they have been deposited until and unless an overlap ocurs in which case they are removed, keeping the system unchanged. This has been implemented in Pygame that gives the position of each disks, stopping time of each sample and also calculates the maximum coverage as in [1]. Once, enough disks have been deposited, the deposition time increases exponentially for the successive disks, in this random process, which has no memory of rejection coordinates.

4. Results

Figure 1: Simulation Snapshot

The coverage values calculated numerically for different radii are as follows:

- 0.04: 0.5127079210658543
- 0.06: 0.49762827632862316
- 0.08: 0.4423362456254428

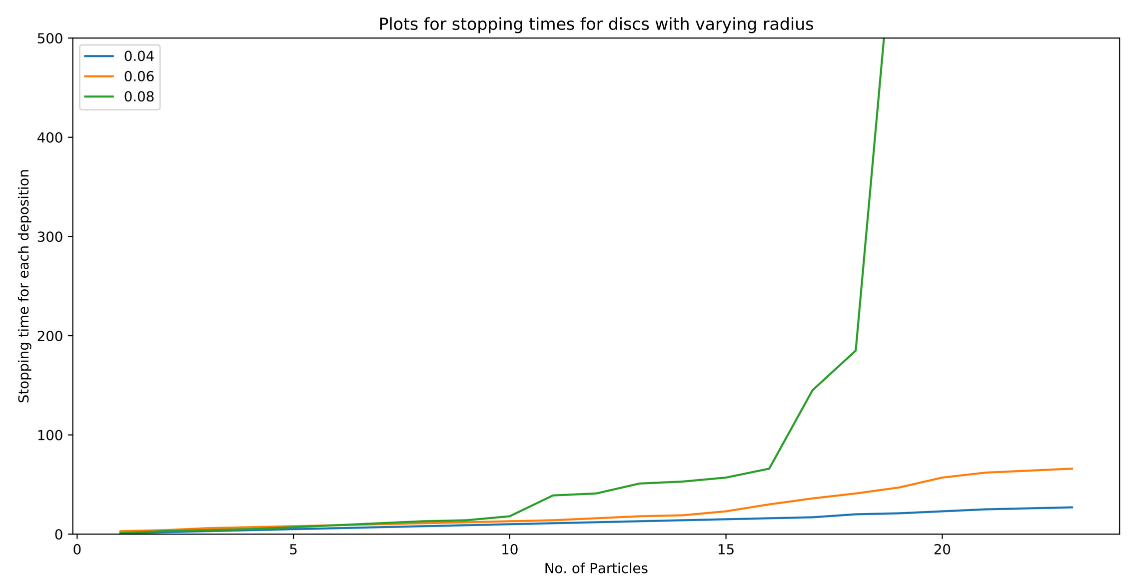

Figure 2: Deposition Stopping Time Plot

5. Faster Than The Clock Algorithm

As Werner Krauth [2] mentions, the DMC methods focus on the accessible regions in the simulation, where new particles can be deposited successfully. After a large number of depositions, Δ_t until the next successful deposition is a random variable. If λ is the probability of deposit rejection, then

$$\lambda = 1 - \frac{area\ of\ accessible\ region}{area\ of\ deposition\ region}, \Delta_t = 1 + int[\frac{log\ ran(0,1)}{log\ \lambda}] \tag{1}$$

The total area of accessible region is computed by cutting it up into smaller regions $\{R_1, R_2, ..., R_k\}$, such that these small regions do not contain any holes. The vectors $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ form the vertices of a polygon that spans the small region R_k. The polygon has the area:

$$A = \frac{x_1y_2 + ... + x_ny_1}{2} - \frac{x_2y_1 + ... + x_1y_n}{2} \tag{2}$$

The circular segments must then be subtracted to get the accessible region from R_k. The sum of all these areas gives the total accessible area which lets us calculate λ and Δ_t. A disk can now be placed successfully on the accessible space. The probability of a disk falling on a region R_k can be computed using Tower Sampling. The process is iteratively carried out unitil a disk is placed successfully. This algorithm advocating geometric computations is much faster than the naive approach. All the codes are available at https://github.com/indrag49/Computational-Stat-Mech and can be easily implemented with Pygame.

6. Pygame in Simulating Statistical Physics

Pygame[3] framework provides additional modularities for drawing graphics, playing sounds, handling mouse input, etc. i.e., creating programs with a Graphical User Interface. Many of the computational Statistical Mechanics simulations were run with the Pygame:

1. Perfect Sampling with Markov Chains was demonstrated with the simulation of the $3X3$ pebble game.
2. Event driven Molecular Dynamics simulation of the hard spheres manifesting their collisions were also carried out.
3. Simulation of sampling a path contributing to free density function, using Lévy construction was also brought about.

7. References

[1] G Zhang and Salvatore Torquato. Precise algorithm to generate random sequential addition of hard hyperspheres at saturation. *Physical Review E*, 88(5):053312, 2013.

[2] Werner Krauth. *Statistical mechanics: algorithms and computations*, volume 13. OUP Oxford, 2006.

[3] Albert Sweigart. *Making Games with Python & Pygame*. CreateSpace North Charleston, 2012.