CSE/ECE-6730
Modeling and Simulation:
Fundamentals & Implementation

Project 1: Checkpoint 1

Team 6:

Arman Afshar
D'arcy Wentworth Roper V
Shahrokh Shahi

February 2019

Problem statement

Recrystallization is a process by which deformed grains in poly-crystalline materials are replaced by a new set of defects-free grains that nucleate and grow until the original grains have been entirely consumed. Recrystallization is usually accompanied by a reduction in the strength and hardness of a material and a simultaneous increase in the ductility. In this project, a Cellular Automata\Monte Carlo algorithm is developed in Python to study the kinetics of recrystallization in two dimensions. The code will be capable of capturing nucleation, growth and slowing of grains. The model maps the microstructure onto a discrete square lattice, with each square having two states of either crystallized or not crystallized. A transformation rule is then applied to a random cell, and based on the state of cell, interaction energy between the cell and its neighbors, the switching probability is calculated. Figure 1 provides a typical cellular automata simulation that is the motivation behind this work.

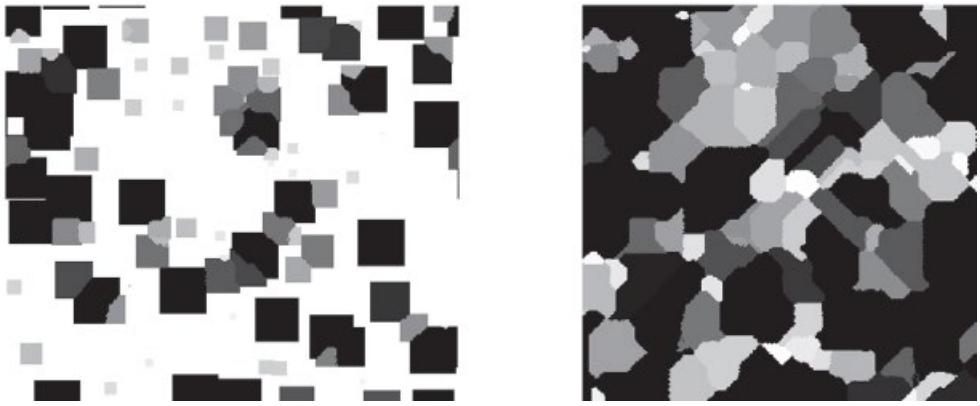

Figure 1. A typical cellular automata inspiring this work, REF [1]

As mentioned, the local interaction of neighboring lattice sites in the algorithm is specified through a set of transformation (switching) rules. The value of each particular lattice site at a time $t_0 + \Delta t$ is determined by its present state (t_0) (or its last few states t_0, $t_0 - \Delta t$, etc.) and the state of its neighbors. Updating scheme could be one site at a time (Metropolis Monte Carlo) or all at the same time (Cellular Automata). We try to use both methods and capture the following goals:

1. nucleation of grains
2. growth of grains
3. the slowing of growth owing to the impingement of grains
4. bifurcation (if any)

Approach

In order to capture these, the following assumptions are made:

1. The geometry of the cells is assumed to be a two-dimensional square lattice

2. The number and the kind of states a cell can possess are two states per site: either recrystallized or not recrystallized (0 or 1)

3. We considered the two-dimensional neighborhoods (5 by5) in Figure 2 for the definition of the neighborhood of a cell (two layers of neighbors)

4. The rules that determine the state of each cell in the next time step are also provided below. These rules govern nucleation of new grains, growth of grains, and the impingement of grains.

For the Cellular automata algorithm, at the beginning of the simulation, all sites were set to zero (not recrystallized) and then N_i embryo grain "embryos" were placed in the system by assigning non-zero values to randomly selected sites on the lattice. Defining the activity as the sum of recrystallized neighbors of the central site based on the neighborhood in Figure 2, a simple rule to describe growth: if activity is greater than one at time step t_0, then the central site would be considered recrystallized at time step $t_0 + \Delta t$ and take on the identity of the grain that extends into its neighborhood. Near impingement, when more than one grain might be growing into a neighborhood, a model for controlling the growth rate of the grains is required. In this study, we say that the growth happens not with a probability P = 1, but with some lower probability. The number of recrystallized sites within the 25-neighbor environment is denoted as A_{25} . If $A_{25} > 11$, then the probability P of growth was reduced and a linear probabilistic model was used:

$$P = \left(1 - \frac{\alpha A_{25}}{100}\right)$$

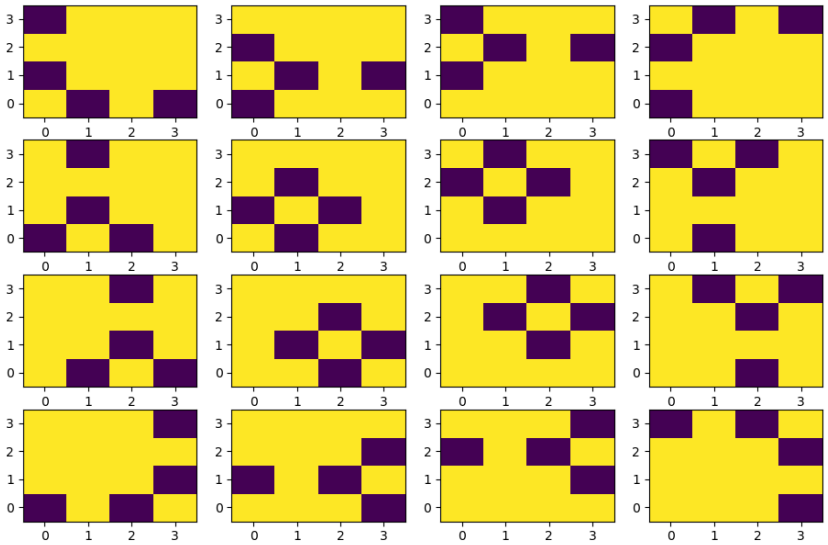

Figure 2. Periodic interaction with neighbors in the algorithm

Monte Carlo algorithm acts similar to the cellular automata, with the only difference being that sites are updated one at a time, based on an energy argument. Any move that make the system goes to a lower energy is considered an acceptable move, so calculating energy on the whole lattice is essential to this algorithm. More of this will be presented in the initial results section. While cellular automata is very good for grain growth simulation, Monte Carlo is more appropriate for capturing bifurcation in the dynamics of the system.

Platform of Development

All the computations and most of the post processing of the results are being developed in Python, with an object-oriented structure, and Microsoft Visual Studio is the IDE that we have agreed on to develop the code. Occasionally however, some of the post processing might be presented in MATLAB for the sake of clarity. Version control and code management will be handled by using GitHub. The computations are not expected to be heavy as we aim to qualitatively show the power of the approach and hence keep the size of the model relatively small, but an effort is being made to keep the computation cost as low as possible by vectorization, smart search algorithms, etc.

Link to GitHub repository: https://github.gatech.edu/sshahi3/CSE6730Project1.git (All instructors and TAs have been added as collaborators)

Initial analysis

This section is dedicated to present some of the initial results obtained by the code, mostly how energy is calculated in each step. Calculating energy accurately is extremely important in the algorithm as it provides us with the minimum energy configuration. The energy is calculated using the following relation:

$$E = -\frac{1}{2}\sum s_i s_j$$

where $s_i s_j$ are the crystallization state of each site. Calculating based on the energy, the following plots are provided, where the x-axis shows the sequence of the steps, and the y-axis displays the energy. It can be seen the after a transient state of the simulation, the simulation goes to a steady state where the dynamics of systems becomes stationary. Physically this means that the material passes the kinetic phase of nucleation and growth and reaches thermodynamics equilibrium.

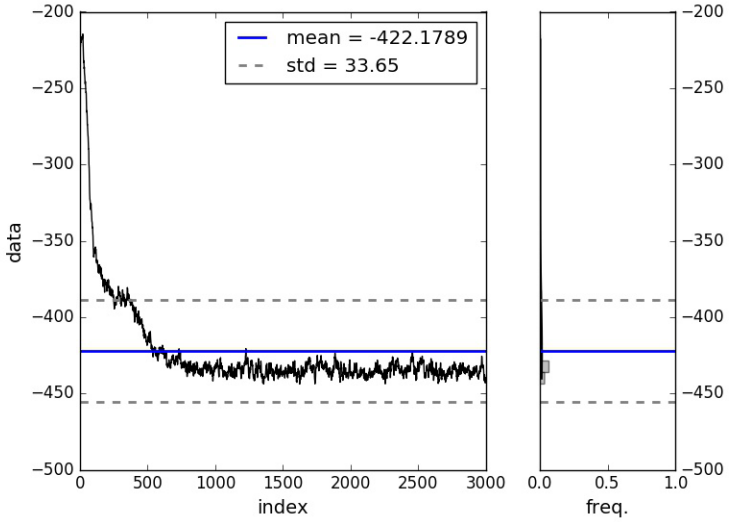

Figure 3. Energy evolution during simulation

It should be noted that these are preliminary results but also of utmost importance. With the energy at hand, we can calculate the minimum configuration of the system, and predict a detailed distribution of the dynamic system at different times, which are ultimate goal of this project. We can also calculated the correlation between different sites, which are presented in the figure below:

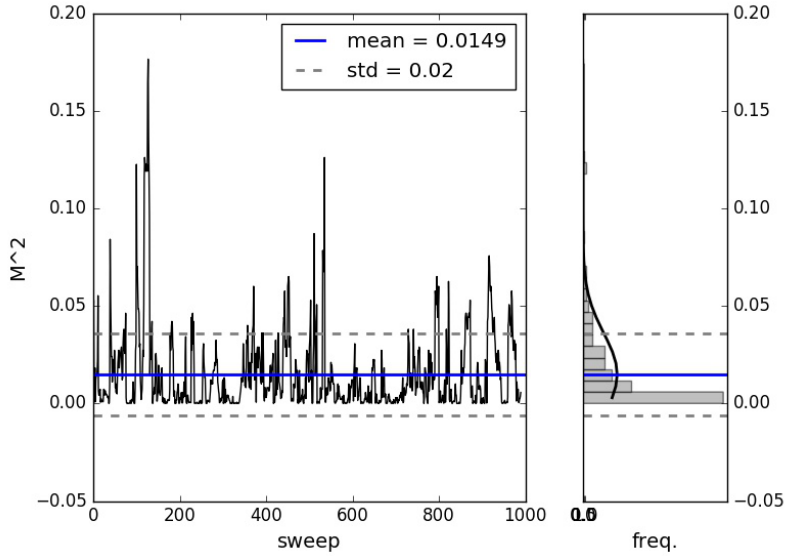

Figure 4. Correlation during transient part of the dynamic evolution

Division of Labor

1. Arman Afshar

Background: Theoretical and Computational Mechanics

Establishing the theoretical framework of the project, physical foundation of the model and how it relates to the computational model. Discretizing the continuum geometry in space and also discretizing in time. Coming up with proper evolution rule based on the literature and the general flow of the algorithm. Implementing the body of the discrete dynamic code in Python.

2. Shahrokh Shahi

Background: Numerical Analysis/Scientific Computing

Implementing the loops in Python, building appropriate classes and objects (basically making the code object-oriented), implementing the Cellular Automata algorithm evolution rule, obtaining initial simulation results in terms of energy and correlation function, making plots with statistical analysis of the results.

3. D'arcy Wentworth Roper

Background: Computer Science

Implementing the Monte Carlo algorithm evolution rule, implementing the switching probability, speeding up the loops and search algorithm in the code, uploading the code in GitHub with appropriate documents preparation

Reference

Frenkel, Daan, and Berend Smit. *Understanding molecular simulation: from algorithms to applications*. Vol. 1. Elsevier, 2001.