Part 2: Spinodal Decomposition

Now, let's take a look at an example application of cellular automata in a materials science application.

Alloys are mixtures of components. For example, bronze is an alloy of copper and tin; the addition of tin increases the hardness and strength. Another well-known example is steel, which is an alloy of iron with a few percent of carbon. Other metals are also added to enhance materials properties such as corrosion resistance (e.g., stainless steel). It is also possible to make alloys of non-metallic elements. An example is InGaN, an alloy of indium nitride (InN) and gallium nitride (GaN), used in some light-emitting diode (LED) applications.

In materials science, alloys are often referred to as "solid solutions" because the components of the system are evenly mixed together- or at least that is what we hope for!

It turns out that many solid solutions do not always stay homogeneously mixed at all temperatures or pressure. Instead, they often segregate into different phases. In some cases below a critical temperature, the phases spontaneously separate into domains with complex microstructure through a process known as *spinodal decomposition.*

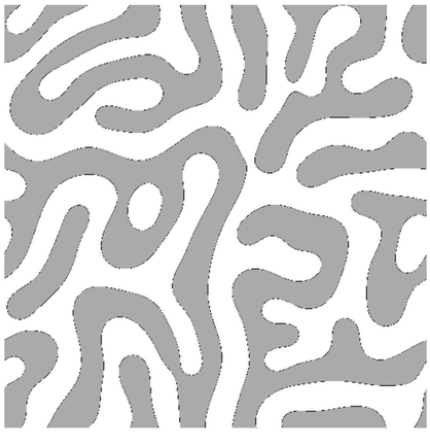


Figure 1: Example microstructure of spinodal decomposition

Compare and contrast with Game of Life

In this part, we will use cellular automata to emulate some features of spinodal decomposition for a binary alloy that contains two components. We will follow the implementation from Oono and Puri, who published a research article in 1987 [1] and Richard LeSar's *Introduction to Computational Materials Science*. Some features of our implementation for modeling spinodal decomposition will be similar to the Game of Life implementation:

- both implementations will use the 2D square grid

- both implementations will use the Moore environment
- the value of each cell at each time step depends on the values of the cells in the local environment

Some aspects for the model of spinodal decomposition will be different from that in the Game of Life:

- whereas cells in the Game of Life took on binary values of ‘alive’ or ‘dead’, cell values in the implementation for spinodal decomposition can take on a range of non-integer values.
- the implementation for spinodal decomposition will distinguish between the up, down, left, right neighbors from the neighbors on the diagonal. The up, down, left, right neighbors are nearest neighbors while the neighbors on the diagonal are next-nearest neighbors.
- the implementation spinodal decomposition will use periodic boundary conditions, which means we implicitly repeat our grid in all directions so that e.g., the neighbors of the cells at the far right of the grid are at the far left

There are several parameters relevant to the materials science aspect:

- *A* (`self.AA` in `spinodal_decomp.py`): a parameter that drives the extent (i.e., thermodynamic favorability) of phase separation
- *D* (`self.d` in `spinodal_decomp.py`): represents the diffusivity, which is a measure of how easily an atom is able to move or diffuse through a material. It is a materials-dependent parameter.
- η (`self.eta` in `spinodal_decomp.py`): the grid containing the concentration profile, which will be updated with each time step

Each of these parameters are controlled by:

```
SpinodalNonConserved(n=100, nstep=300, d=0.5, AA=1.3)
```

Our binary alloy is comprised of elements 1 and 2. In general elements 1 and 2 will not be homogeneously mixed; some areas will be higher in concentration of species 1 and others will be higher in concentration of species 2. We define concentration in the usual sense; if you have 100 atoms and 60 are species 1 then you have 60 atomic percent (at%) concentration of species 1 and 40 at% of species 2. For the purpose of this exercise, a single concentration represents a specific phase.

On our 2D grid, each cell is assigned a value η that represents a concentration difference between two different phases.

$$\eta = \Delta c = c_1 - c_2$$

If η ~ 0, then the cell is about equal concentration of phases 1 and 2. If η > 0, then we can define that the cell is richer in phase 1 corresponding to concentration c_1. Conversely, if η < 0, the cell is richer in phase 2 corresponding to concentration c_2.

Physically, when the concentration is changing spatially, atoms of elements 1 and 2 are moving through the material, i.e., diffusing.

Compare two rules for updating the cells

We've implemented two update rules for simulating spinodal decomposition. You can try each by instantiating the appropriate class:

`SpinodalNonConserved(n=100, nstep=300, d=0.5, AA=1.3)`
This creates a simulation that uses a non-conserved update rule. The concentration field, η, evolves based on a local free energy term, but does not conserve total concentration, so it does not accurately model diffusion.

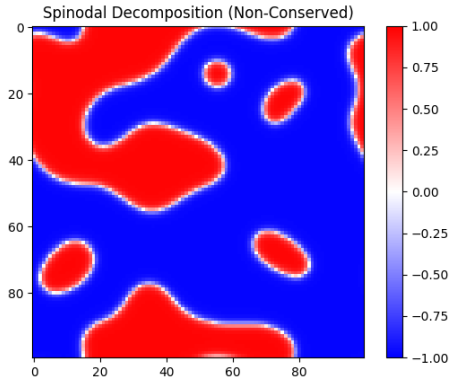
`SpinodalConserved(n=100, nstep=300, d=0.5, AA=1.3)`
This creates a simulation using a conserved update rule. It modifies the non-conserved evolution by introducing a second-order correction, ensuring that the total concentration is conserved, which better reflects physical diffusion behavior.

Exercise: Let's get a feel for what happens in our simulation of the spinodal decomposition process.

Run the following code snippet (already provided in the notebook as well):

```
sim = SpinodalNonConserved=(n=100, nstep=300, d=0.5, AA=1.3)
sim.animate
```

A snapshot of what you should expect to see is below. If your simulation is working, you should see the microstructure change with time!



Now run the conserved updater.

Run the following code snippet (already provided in the notebook as well):

```
sim = SpinodalConserved=(n=100, nstep=300, d=0.5, AA=1.3)
sim.animate
```

Wang Materials Group
UT Austin

Guiding questions:
Compare and contrast the non-conserved with the conserved updater. What do you notice about the pattern of phase separation (aka microstructure)? What do you notice about the evolution of the microstructure with time? Compare your observations with your neighbors.

Exercise: So far, the initial concentration profile consisted of small random and symmetric perturbations around concentration parity (η ~ 0). Let's see what happens when we start with different concentration profiles.

Change the initial concentration profile from random to one of the other options. What the code snippet below does is it redefine the initial condition by overriding "`self.eta`." You can also implement this by directly editing the `spinodal_decomp.py` file yourself.

==========

Bonus: So far, we have assumed certain parameters for the diffusivity *D* (`self.d` in `spinodal_decomp.py`). Try varying self.d. Take note of your observations on the microstructure and it evolution over time. You can also vary the diffusivity in the Colab notebook directly.

Bonus: Let's take a closer look at the implementation, which reveals a bit more about the underlying materials physics. We define a parameter that defines the state of each cell. We choose the parameter

$$\eta = \Delta c = c_1 - c_2 \quad ,$$

which represents a *concentration difference* between the two phases. For example, phase 1 could be a phase of steel that has a concentration c_1 of 0.8 at% Cr (i.e., for every 125 iron atoms, there is a chromium atom) and phase 2 could be a phase of steel that has a concentration c_2 of 1 at% Cr.

For updating η at each time step, a rule that emulates the thermodynamic process of spinodal decomposition could be

$$\eta_i(t+\Delta t) = A \tanh(\eta_i(t))$$

where $A > 0$ and represents an effective interaction parameter between the components. Consider the expected mathematical behavior of this. The form of the hyberbolic tangent function tanh() is shown in Figure 3. If $\eta_i(t) < 0$, then $\eta_i(t+\Delta t)$ will be driven towards $-A$ with several iterations. Similarly, if $\eta_i(t) > 0$, then $\eta_i(t+\Delta t)$ will be driven towards $+A$ with several iterations. Thus, this rule emulates the thermodynamic driving force that pushes the system to segregate into two difference phases. From a materials perspective, we can think of the tendency to phase separate as the bonding between components *a* and *b* to be different extents of unfavorable.

However, this rule alone does not include any influence of the neighboring cells.

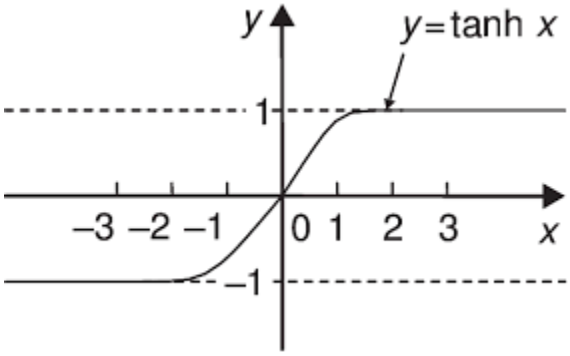


Figure 2

To include the influence of the local environment, we recognize that in order for concentration to vary as a function of time step in a real material, atoms must move between neighboring cells. This process is known as diffusion, and is linearly proportional to the concentration difference; that is, if there is a large concentration difference, more diffusion occurs. In terms of materials parameters, we use the diffusivity *D*, which is a measure for how easily an atom is able to move through the material.

The total rule for updating cell values is

$$\eta^{nc}(t+1)=f[\eta_i(t)]=A\tanh(\eta_i(t))+D(\langle\langle\eta(t)\rangle\rangle_i-\eta_i(t)) \quad (1)$$

where $\langle\langle\eta_i(t)\rangle\rangle=\frac{1}{6}\sum_{j\in r,u,l,d}\eta_i+\frac{1}{12}\sum_{j\in ru,lu,ld,rd}\eta_i$. The first sum is over nearest-neighbors (right, up, left, down) and the second sum is over next-nearest neighbors along the diagonals. The double brackets indicates a sort of average over nearest neighbors.

However, it turns out that Eq. (1) is not quite correct because the concentration (i.e., matter) is not conserved; hence, we call this the non-conserving (nc) rule for spinodal decomposition. The concentration changes in one cell should be compensated by opposite changes in adjacent cells, which would be a truer reflection of the diffusion process. In order to include this aspect, we can add the term in double brackets giving rise to the conserving (c) rule for spinodal decomposition in Eq. (2).

$$\eta_i^c(t+1)=f[\eta_i(t)]-\langle\langle f[\eta_i(t)]-\eta_i(t)\rangle\rangle \quad (2)$$

The second term in double brackets calculates the change at each site and then subtracts the average of that change from the concentration of neighboring sites, thus conserving concentration, i.e., matter.

[1] Y. Oono and S. Puri, “Computationally Efficient Modeling of Ordering of Quenched Phases”, Phys. Rev. Lett., 58, 836 - 839 (1987)