Applications for Simulations of Materials Microstructures

Olof Hellman

Abstract : We present two approaches toward simulation in Materials Science with implementations as Macintosh applications. Continuum simulations are used to model materials structures which can be described by a smoothly varying parameter which is a function of position. For example, the elemental composition of a material , its temperature, or its crystallinity can be expressed in this way. Modelling the evolution of such a parameter through time is approached through a phase-field approach.

An alternative approach is to identify discrete entities in a materials structure and model their individual behaviors through a set of equations which couple their interactions. At the most basic level, individual atoms in a material are treated as discrete objects. This atomistic approach can be used to calculate many macroscopic materials properies based on the fundamental interactions between particles.

Introduction

Materials science is the core set of knowledge describing why materials have the properties they do and how they come to be. It covers questions like why crystals form and why diamonds are so pretty, why glass is glassy and what makes glass in different colors, why pieces of metal get harder to bend the more you bend them, and why some metals corrode and others don't.

Essentially, materials scientists are in the business of understanding why materials work they way they do, and finding ways to make them work better. One of the fundamental principles of materials science is that the structure of materials determines their properties. For example, an abalone shell is made mostly from the same material as blackboard chalk, $CaCO_3$, yet the two materials differ in almost every way: color, hardness, strength, water absorption, etc. [1]. Structure can mean the atomic level arrangement at the nanometer scale, the variation in composition of a material, which can span from the nanometer s to centimeters, or the distribution of crystal defects such as grain boundaries and dislocations, generally on the micrometer scale.

Any sort of materials property might be of interest: strength, ductility, heat conduction, heat capacity, electrical conductivity, transparency, or any number of others. All these properties depend intimately on the microstructure of the material. Understanding the evolution of materials microstructures thus contributes to an understanding of how materials properties develop and can be engineered.

Simulation applications can be used to explore these structures, both as a research tool in conjunction with experimental techniques and as an education tool for developing intuition about important factors in microstructural evolution.

There are a wide variety of approaches towards simulation in materials science. This paper covers the two most widely-used types of simulations which are applied to microstructural evolution. The two kinds of simulation illustrated here are:

- Continuum -- A grid of points is used to model the continuum behavior of a material: each grid point represents a small region of space. Finite Element Method is a kind of continuum simulation. Applying a blur in photoshop is a continuum simulation of diffusion.

- Autonomous Cell -- A collection of objects interacting with each other according to a set of rules. Molecular Dynamics is this kind of simulation, where each atom is an object interacting with other particles through Newton's Laws. SimCity is another.

Phase Field Simulations: *CahnMan*

CahnMan is an application for phase field, continuum simulations. In a phase field simulation, a grid of points is defined, each of which is associated with one or more physical parameters, such as composition, temperature, crystallinity, etc. Parameters not assigned to the grid points apply to the grid as a whole: for example, if temperature is not a grid point parameter, it can be assumed to be constant for the whole and is a parameter for the whole simulation.

The grid of points can one-, two- or three-dimensional. It can also be regular or adaptive: the grid points can consist of a set of fixed, regularly spaced points, or they can be irregular and unfixed, with points being created and destoyed as called for by the simulation. It also has a boundary symmetry: In the simplest case, there is translational symmetry (i.e. wrap-around) where the points at the edges are assumed to be adjacent to the points on the opposite side. The image in figure 1 is an example of a two-dimensional grid of compositions as calculated with our application *CahnMan*.

Let's illustrate this type of simulation for the simplest possible case: a 2D system with two components, A and B. A single parameter represents the composition at each point on the grid. This composition parameter will vary from 0 (pure A) to 1 (pure B).

This system will evolve based on some simple rules about the interactions of the parameters. The most important rule is that the system will evolve to lower its energy, Just as in physics where masses are accelerated in a gravitational field toward positions of lower potential energy, materials systems are pushed toward configurations of lower energy. It is necessary, then, for a simulation to be able to calculate the energies for any configuration.

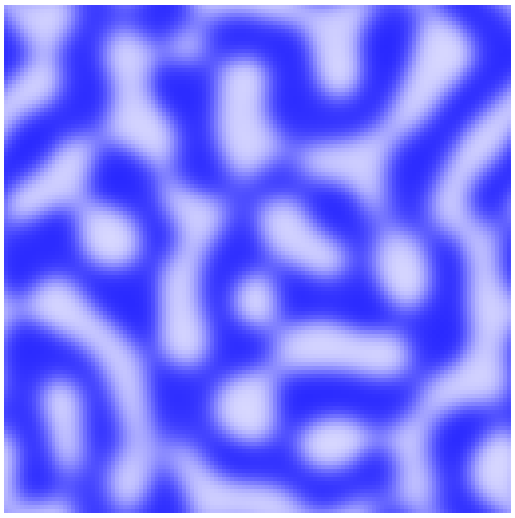

Figure 1. A two dimensional grid of composition values. The scale from blue to white corresponds to 0% to 100% of a component in the two component system.

Energy Models

The energy of the system is calculated based on the values of the parameters at each point. For example the regular solution model calculates the energy of mixtures of two components as

$$f(C) = \frac{T}{T_c}(C \ln C + (1-C)\ln(1-C))$$

Here, Tc is a critical temperature above which mixing occurs and below which a separation occurs, and C is the composition between 0 and 1. When performing the simulation, this energy can be calculated at each point on the grid, and the total energy of the grid is the sum of those energies. This model for compositional energy results in the phase diagram for the system as shown in figure 2.

The phase diagram gives information about what phases are expected to be present in a system of a given overall composition at a given temperature. Points above the curve in figure 2 are one-phase; points below the curve will exhibit phase separation into two-phases. For example, point X on the diagram represents a composition of Cx, roughly 60% of component B and 40% of component A, at a temperature Tx. For this combination of temperature and composition, two phases are expected to coexist: an A-rich phase of composition CxA, and a B-rich composition of composition CxB. The relative amounts of these phases will be proportional to the length of the line segments fA and fB: i.e. the fraction of the A-rich phase will be fA / (fB + fA). This is the so-called *lever rule*.

There are a number of actual systems

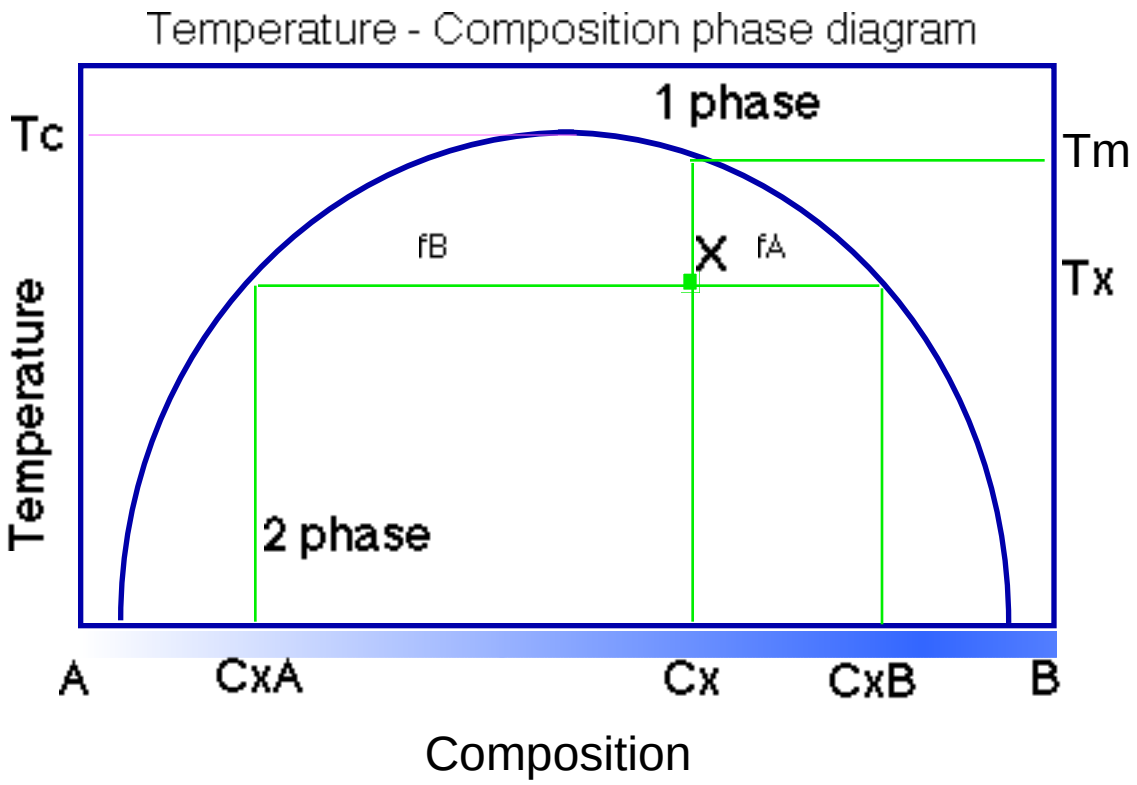

Figure 2. Schematic phase diagram for a phase separating system.

which exhibit similar types of behavior: the two components might be water and a light oil, or metal alloys such as Aluminum-Copper or Copper-Cobalt, or they might be two types of polymer. In all cases, these systems exist as a single phase at a higher temperature and separate into two phases at a lower temperature.

One constraint on composition is that the total amount of components A and B can not change. So, change in the compositionat one grid point must be balanced by a of the opposite magnitude at another point. Such a parameter is called a conservative quantity. Other parameters, such as crystallinity, need not be conservative. For any parameter, the rate at which it can change is also a property of the material. For changes in composition, this corresponds to a diffusivity or mobility of each component in the material.

Calculating Rates of Change

For conservative quantities like composition C, the change in that quantity can be estimated from the spatial curvature of *f*©: If the curvature is positive, energy is reduced by homogenization: two regions of different composition can lower their energy by exchanging matter so that they have the same composition. If the curvature is negative, phase separation is expected: two regions of similar composition can lower their energy by exchanging matter so that their compositions differ.
Assuming that matter can move with a mobility M , the change in composition as a function of time t due to compositional energy is then

$$\frac{\partial C}{\partial t} = M\nabla^2\left[\frac{\partial f}{\partial C}\right]$$

where $\nabla^2$is the laplacian: i.e. the second derivative taken in all spatial dimensions.

However, in most systems, compositional energy is not the only contribution to the energy. There is also a contribution from gradients in the composition field. That is, a region of the grid in which the composition is changing has a positive energy, and thus the system will tend to evolve to get rid of boundaries between regions of different compositions. The contribution of this gradient energy is taken to be proportional to the curvature of the concentration:

where κc is a constant proportional to the boundary energy. The effect of this assumption is that the composition must vary smoothly from a region of one concentration to another: an abrupt change in composition would imply an infinite curvature in the composition and thus an infinite boundary energy.

Now all the pieces are in place to find

$$\frac{\partial C}{\partial t} = M\nabla^2\left[\frac{\partial f}{\partial C} - \kappa_c\nabla^2 C\right]$$

how the system evolves with time. The change in composition as a function of time at any point on the grid is given by

which is known as the Cahn-Hilliard equation.[2] Essentially, the right hand side is the mobility M times a vector which points downhill on a plot of the energy as a function of all N2 - 1 composition variables. All those composition variables will change so that the system will move down that slope at a speed proportional to M, which is the mobility of material moving from one point to another. This function is rather curious because it involves a fourth order spatial derivative that directly describes a physical phenomenon. Efficient calculation of the time evolution of

systems described by this equation are a matter of current research.

For one iteration of the simulation, we examine each grid point, calculating the instantaneous value of dC/dt. We approximate the derivatives numerically, such that in two dimensions, the laplacian is just

$$\nabla^2 u_{i,j} = \frac{1}{h^2}\left(u_{i+1,j} + u_{i,j+1} + u_{i-1,j} + u_{i,j-1} - 4u_{i,j}\right)$$

where i and j denote the indices of grid points in the two dimensions and h is the grid point spacing.

Then we choose a timestep small enough that the changes in composition over that amount of time are smaller than a threshold, and change the composition values accordingly. The result is a physically accurate description of the evolutions of isotropic systems determined by compositional and boundary energies.

Using *CahnMan*

The parameters which are controlled by the user in a *CahnMan* simulation are the same parameters which might be accessible in a real experiment, as well as others that are not accessible: For example, the temperature would be controlled by an external heater: in *CahnMan* it is a property of the simulation object which is linked to a GUI control. The interface energy parameter κc is reflective of a materials property which is fixed for a given material. In the simulation, however, it can be freely changed. All of these parametrs are global for the simulation: they do not vary from grid point to grid point.

CahnMan calculates and presents to the user the quantities which are evaluated at each grid point. The visual representation of these structures over time is the qualitative output of the simulation. These values can also be accessed quantitatively through interaction by AppleScript.

For students exploring the basics of microstructural evolution, *CahnMan* represents the first interactive application with a combination of realtime visual feedback as well as quantitative the more traditional numerical output.

Atomistic Simulations with kSan

Materials can also be models on the atomistic level, the most basic level of interactions which determine most physical properties. On this level, a set of atoms interact with each other by forming bonds, the energy of which are approximated with an *interatomic potential* (the interatomic potential is also called the *force field* in atomistic organic chemistry modelling). That is, it is assumed that each particle can be assigned an energy based on the local configuration of atoms. Forces on particles are determined from the derivative of the potential as a function of position of the particle in question.

The goal of most atomistic modelling is to calculate a macroscopic property of a large collection of atoms by making only simple of assumptions about how particles interact. These macroscopic properties can also be measurable properties, so that direct comparison of calculation and experimental values can be made. For example, the heat capacity of a material is the amount of energy per mass of material required to raise its temperature, expressed as energy per mole per Kelvin. On the atomic scale, we can simulate the motions of a collection of atoms and explicitly calculate their energies at different temperatures, resulting in a calculation of energy per atom per Kelvin. Just as it is difficult to experimentally measure the energy of a

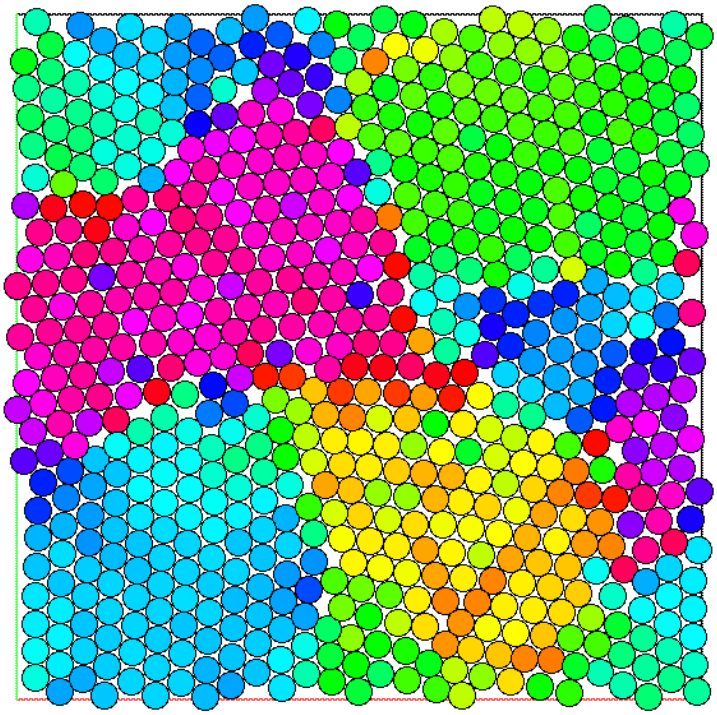

Figure 3. Local arrangement of particles in a 2D nanocrystal. Color is applied based on the local rotational orientation of a hexagonal lattice. This image was generated by placing 1000 particles at random positions in the simulation cell and perforing an energy minimization.

single atom, it is impractical to simulate a mole of atoms (6.02×10^{23}).

Thus, simulation and experiment are often performed at different length or time scales. It can be a simple matter to extrapolate from one extreme to the other. On the other hand, advanced experimental techniques are bringing the experimental length scales down just as fast as increasing computation power is increasing the length scales accessible to simulation.

Some Examples of Simulation

1. Grain structure in a nanocrystal.

Recent experiments have shown that the properties of materials can change in

surprising ways when the grain size is very small. This occurs in a rapidly solidified material, or in supersaturated alloys, or in materials which have been mechanically deformed so that the original grain structure is destroyed. The resulting grains can be smaller than a simulation cell, and thus simulation can be performed to predict the atomic arrangements in these materials. Figure 3 shows a 2D simulation where the initial positions of the particles are chosen at random, and the energy is minimized. The equilibrium configuration for this 2D solid is a hexagonal lattice. The color of each particle is based on the local rotation of this lattice: thus the colors represent different grains.

2. Energy and Stress Grain Boundaries

The region in a crystal where the local rotation of a lattice changes is called a grain boundary. These boundaries are unstable with respect to the undefective crystal: i.e. a lower energy can be obtained by getting rid of the boundary. Thus the boundaries have an excess energy which can be calculated in a simulation. This boundary energy can

vary as a function of the relative rotation of the lattices on either side of the boundary. Such a calculation is illustrated in figure 5.

The magnitude of the boundary energy can be calculated from the difference between the actual sum of the energies of the particles, and the sum of their energies of there were no boundary: the result is about 1 Joule per square meter .

Stress is defined as a change in energy associated with a volume expansion or contraction. If the energy is negative for expansion it is a compressive stress. This can be evaluated in a simulation on a point by point basis. Visualization of this stress distribution results in the figure shown in figure 6.

3. Dynamics of atomic motion

Using a molecular dynamics simulation, the atomic motions can be tracked on the scale of nanoseconds. This is sufficient time for a surface atom to make several hops from one surface

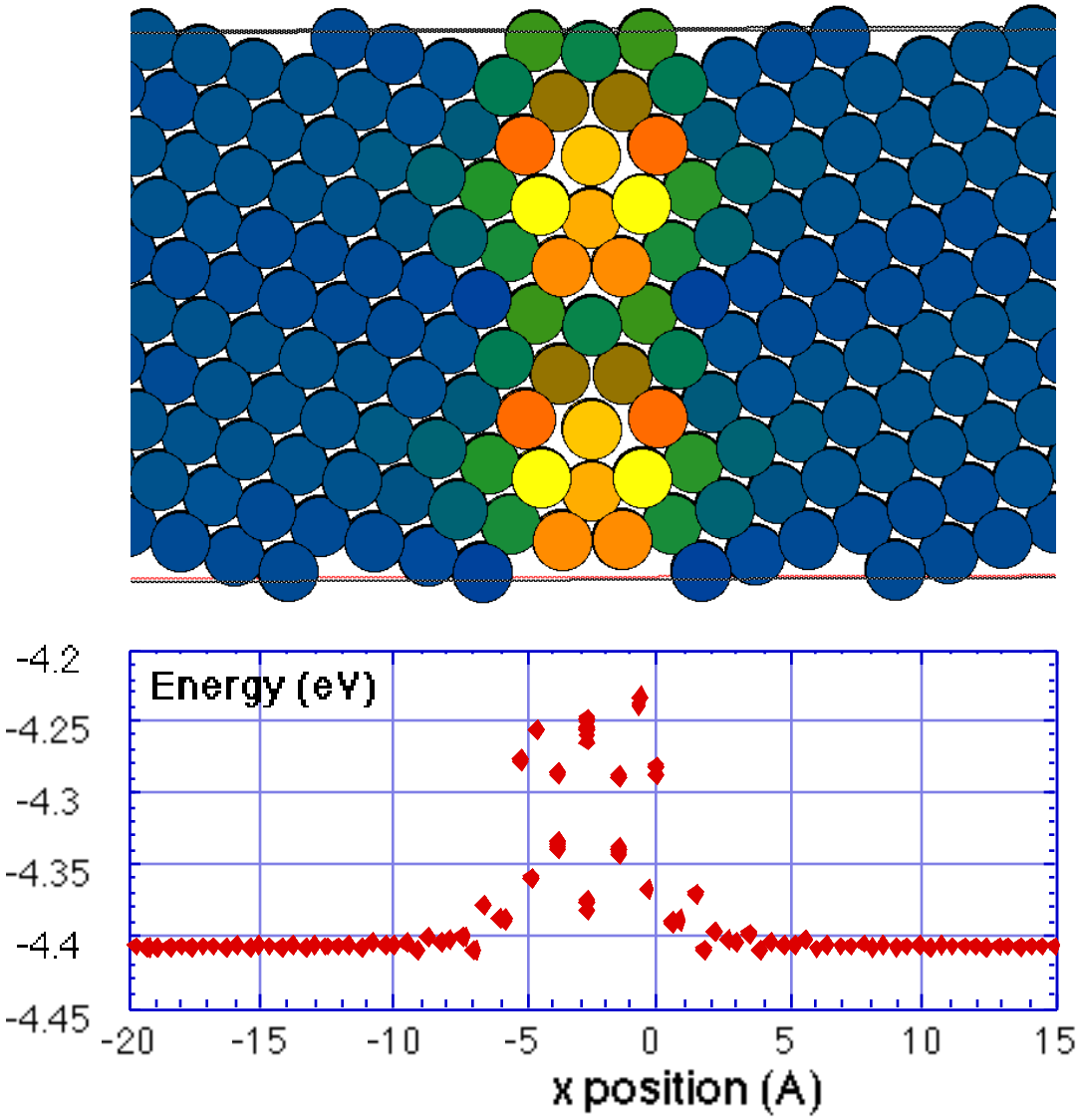

Figure 4. Energy Map of Particles near a grain boundary. A projection in one dimension parallel to the boundary is shown above. Colors are applied based on the energy of each particle. A quantitative plot is shown in the graph.

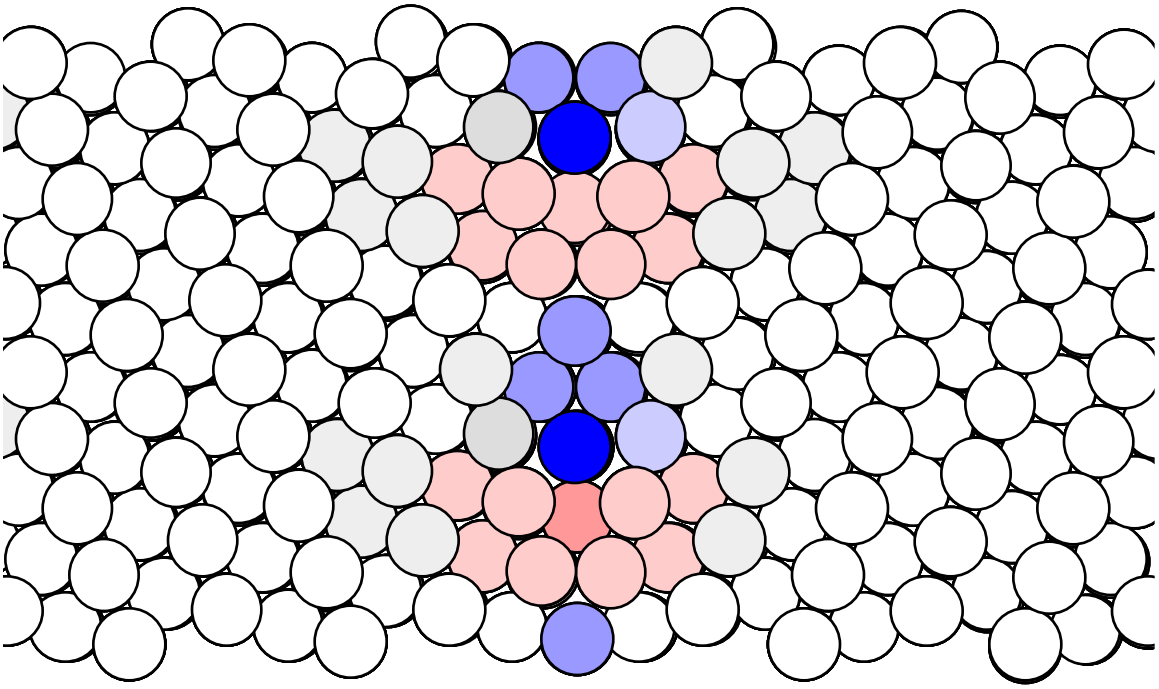

Figure 5. Stress distribution near a grain boundary. Blue sites are in tension, red sites in compression. The magnitude of the stress is highest at the tensile stress, but the spatial extent of the compression extends further from the grain boundary

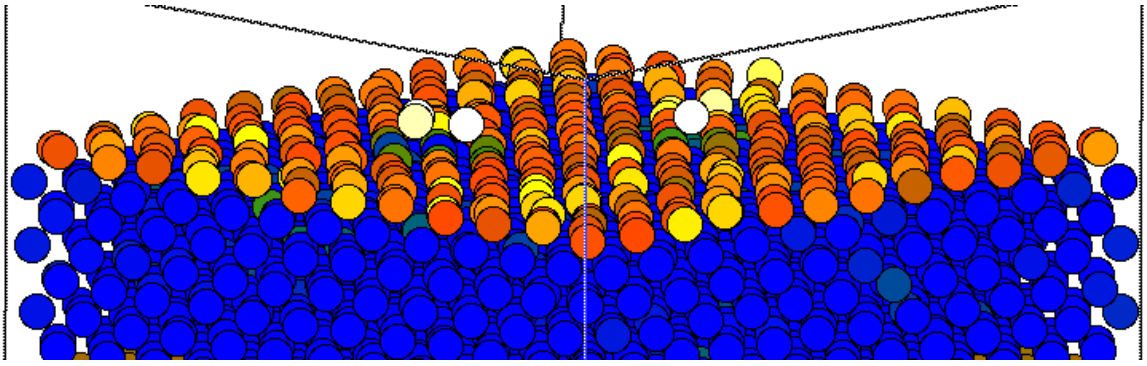

Figure 6. Frame from an animation of simulation of atoms moving on a surface. The lower atoms are in a face-packed cubic structure, and individual atoms are placed on the surface. The mobilities of surface atoms will determine how materials grow during a thin film growth process such as vapor deposition.

location to another, or for a crystal defect such as a vacancy to exchange position with neighboring atoms. Movies can be generated of this process.

A still from one such animation is shown in Figure 7. This is an animation of atomic motion at a surface. During such a simulation, the positions of the surface atoms are tracked over time to estimate their mobility. Mobility of surface atoms is crucial in many of the electronic materials processing which makes semiconductor devices. Low mobilities will prevent good crystal

growth. High mobilities will allow interdiffusion, which can destroy the carefully engineered nanostructures of a device.

Components of a Simulation

A number of different pieces are put together to make an atomistic simulation. We discuss them here in terms of the programmatic object model used in our application *kSan*.

1. Interatomic Potential

We have already mentioned that the energies of particles and the forces on them are calculated by an interatomic potential. Here we review the basic types in common use. One common feature of most potentials is that they are limited in spatial extent to a small radius around any given particle, typically encompassing anywhere from 4 to 100 neighboring particles. This is usually appropriate for most materials, the only exception being ionic solids. All potentials depend in part on the distance between particles, and the form for this dependence is usually repulsive at short distances, and attractive at long distances, with a minimum somewhere close to the equilibrium atomic spacing.

At the simplest end is the pair potential, in which the energy of each particle is the sum of the energies of interaction of that particle with its neighbors. The term pair potential implies that each two-particle interaction is calculated independent of the positions of any of the other neighboring particles. Pair potentials such as the Lennard-Jones [3] potential are easy to calculate, as they are based only on the distance between particles but they fail to predict most of the real properties of materials, although they seem to work well for liquids and solids of noble gases like neon or argon. Pair potenital are speedy: Systems of hundreds of thousands of particles can be treated in seconds.

There are a number of different ways to incorporate many body effects into a potential. One approach is to explicitly calculate the angles between bonds formed by the particle, and express each bond energy as a function of the bond angles as well as a function of the radial separation. This is the approach taken in Tersoff's potentials for silicon[4] . This is computationally simple only for a small number of neighbors: in the case of silicon, the structure is the four-fold tetrahedrally coordinated diamond structure, so this is not a problem. This is also the approach used in a number of potentials used for organic molecules, where the coordination numbers, i.e. the number of bonds formed by each particle, are small. If the number of bonds is high, this calculation becomes slower because the number of bond angles which must be calculated increases by the number of neighbors squared ($N^2/2$). A potential like this can also be speedy, but the calculation of bond angles slows the calculation by one order of magnitude compared with a pair potential.

There is also a method for implicitly including a dependence on bond angle: this is the basis of the Modified Embedded Atom Potential[5]. This potential is designed for metals in various crystal structures. This scheme uses some mathematical sleight of hand to incorporate a bond angle dependence without explicitly calculatingbond angles. Thus, a bond angle dependence can be efficiently calculated without calculating N2/2 bond angles. One limitation is that all the bond angles are equally weighted, but this can be tolerated, especially in metals where the number of effective neighbors is usually between 12 and 14.

A more general potential for metals is the glue potential, also called the Embedded Atom Method (EAM) potential [6]. It incorporates a many body effect by calculating a, "electron density" created by the neighboring atoms. The energy of a particle is composed of a sum of pair interactions plus an embedding term or glue term which is a function of the "electron density". This kind of potential has a "feel-good" factor in that the math resembles the math used for a quantum mechanical calculation of the energies, even if all the parameters used in the potential are actually empirically determined. Some recent research has focussed on deriving parameters for these potentials from quantum mechanical calculations.

These potentials are rather efficient to calculate because they require only a single iteration through each neighboring atom to get the pair interaction and the contribution to the electron density. Accurate EAM potentials have been developed for a number of pure metals, including Aluminum, Nickel, Magnesium and Gold, all of which are close packed crystal structures with 12 equivalent nearest neighbors.

The EAM is also fairly good at describing the interactions in alloy systems. This is because the electron density can be calculated no matter what the type of atom at each neighboring site, and the pair interaction between atoms of different types can be tuned to account for stronger or weaker bonding behavior. These potentials are essentially an order of magnitude slower than pair potentials.

Although most potentials are short ranged, ionic solids are an exception. In an ionic material, electrons are transferred from cation atoms to anion atoms, creating packings of discrete electrically charged particles. These charges interact with other through Coulombic forces, which are long-ranged. Rather than calculate all of these long range forces discretely, which would involve a prohibitively large number of pair interactions, the calculation is split into near-range interaction calculated the traditional way, and long range interactions calculated in reciprocal space. This is the method of the *Ewald sum*, and is only used when necessary for these long-range interactions.

There are a number of approaches for a more rigorous incorporation of quantum mechanics in the interatomic potentials. One is the *bond-order potential*, which essentially attempts to determine the valence of a particular atom in a particular configuration before calculating the energy. A second approach is called semi-empirical *tight-binding*, in which multiple particle effects are accounted for in a simple solution of single atom electron orbitals and the pair interactions are determined empirically. These methods are much slower than pair potentials, typically three orders of maginitude slower.

Still more rigorous approaches are used with no empirical input. These are known as *ab initio* methods. Essentially, they are different ways to solve the Schrödinger equation so as to calculate the states of all the electrons involved in interatomic bonds. Here ther are more acronyms than you can imagine, including LCAO (Linear Combination of Atomic Orbitals), FLAPW methods (Fully Linearized Augmented Plane Wave) , LMTO (Linear Muffin Tin Orbital). These calculation really require a lot of computing power, often requiring months of computer time to perform the most basic of calculations on a system of only hundreds of particles.

The application *kSan* treats the interatomic potential as a property of

the simulation, and is implemented as an AppleEvent object. Each interatomic potential class inherits from the interatomic potential class, and it required to provide functions to calculate particle energies and forces on particles given a particular configuration. different interatomic potentials can be implemented as plug-ins to the application.

2. Iterator

Most atomistic simulations use an iterative process to generate different atomic configurations. The exact nature of the iterative process can be different depending on the goal of the simulation. Each iterative process is similar in that a collection of particles with positions, velocities, atom types, etc. are transformed into a similar collection of particles with, perhaps, new positions, new velocities, new atomtypes, etc.

2a. Minimization

There are three traditional iterators. The first is the energy minimizer. In this iterator, the particle positions are changed such that the total potential energy of the system is reduced, just as the iteration method in *CahnMan* proceeded to reduce the total energy of the system. The enrgy minimizer is often implemented by a steepest gradient algorithm or a congugate gradient algorithm. Both algorithms terminate in a local energy minimum which may or may not be the global minimum. Particle velocities are ignored.
Note that energy in the atomistic simulation can come from the potential energy of the particles as calculated by the interatomic potential, and from the kinetic energies of all the particles.

2b. Molecular Dynamics

The second traditional iterator is the Molecular Dynamics (MD) iterator [7]. This is a straightforward application of kinematics on the atomic scale: knowing each particle's position (i.e. r) and velocity (i.e. dr / dt), the interatomic potential can be used to calculate forces on particles and hence accelerations (i.e. d2r / dt2). All this information can be used to predict the position and velocity of the particle after some very short amount of time. With new positions, a new acceleration can be calculated, and the iterative process can continue.

In one implementation of the MD iterator known as the Verlet method [8], the effect of the third derivative can be made to disappear, and hence the error in the calculation is of the order of the fourth derivative. In another method known as the predictor / corrector method, third and fourth derivatives from previous iterations are stored and used to predict future positions, velocities and accelerations. The actual future accelerations are then calculated, the error in the predicted quantities are estimated, and all the predicted quantities are corrected by this factor. In both of these MD methods, error in the calculation results in the non-conservation of energy, which results in extra kinetic energy being released into the system (i.e. heating) . To account for this, MD methods must provide some sort of thermal equilibration. Using smaller timesteps results in less error but uses more CPU time. The tradeoff is using a stronger thermal equilibration. However, this is an artificial constraint and will tend to produce less real trajectories of particles. Finding the right balance depends on the goal of the simulation.

2c. Monte Carlo methods

The third traditional iterator is called *Monte Carlo*, or often *simulated annealing*. This method is by far the most powerful and most elegant of the three. The result of a Monte Carlo simulation is a sequence of configurations which are representative of the system at a certain temperature. The theory behind the method is both simple and subtle: Imagine a system which can have two different configurations, A and B. There will be an energy associated with state A and an energy associate with state B, and accordingly a change in energy when moving from state A to state B. Now we ask " what is the probability that there is sufficient thermal energy available to move from A to B?"

Some of the time, moving from A to B represents a decrease in energy, and so no thermal energy is required: the probability is 100%. If, however, the change is an increase in energy, then the answer is more difficult. In any material at some temperature, there is a distribution of particle energies, both potential energies and kinetic energies. The actual amount of energy is rought kT, where k is the boltzmann constant and T is the temperature. At room temperature, kT is about 0.025 electron volts, or 4 x 10 -21 Joules. For comparison, the energy in one photon of green light is about 2 electron volts, much higher than kT (this is why most objects do not naturally glow green at room temperature).

Of course, there is a broad distribution: sometimes there is less than kT available, sometimes much more. However, we believe that we know what this distribution is, and we can predict that the probability of having a certain amount of energy E available any particular time is exactly

$$P = e^{\left(\frac{-E}{kT}\right)}$$

because this is the kind of distribution generated by random exchanges in energy from one particle to another. Graphically this means that at a temperature of 1, the probability of having thermal energy E / kT is shown in figure 3.

The meaning of this distribution is interesting: most of the time the thermal energy available is about kT. However, it is possible but infrequent to have much higher energies available. The energy of 10kT is available one chance in 20,000. energy of 20kT is available one chance in 500 million. These improbable events do actually occur and are responsible for chemical

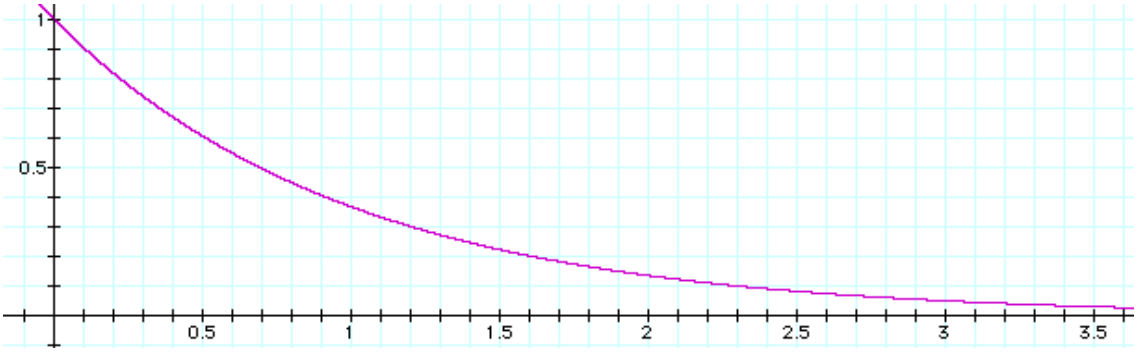

Figure 7. Graph of the likelihood that sufficient energy is available for a particular transition to take place as a function of the transition energy E over kT.

reactions which are thermally activated processes. Common examples include evaporation of water and burning of wood. The secret is that nature tries a lot. For example, most molecules vibrate at a frequency of 10 thousand Gigahertz or so. If you played the Big Game Lottery at the same rate, you would hit the jackpot 150,000 times per second.

For the purposes of simulated annealing, the meaning is much more subtle. When coming up with the comfigurations A and B, we don't really care about how those configurations came to be: all we care about is their energy. For the same reasons that random fluctuations generate the probability chart shown above, we can say that the relative probabilities of configurations A and B are also given by

$$P_{\Delta} = e^{\left(\frac{-\Delta E}{kT}\right)}$$

where ΔE is the difference in energies. i.e. if configuration A has a certain probability Pa, then configuration B has the probability Pb = Pa * PΔ. All configurations of equal energy have the same probability. Then here's the algorithm for collecting a representative sampling of configurations: starting with configuration A, use some random method to generate a configuration B. Calculate the difference in energies between configuration A and configuration B. If the change in energy is negative, always accept it. If the change in energy is positive, choose a random real number between 0 and 1. If the random number is less than exp(-ΔE /kT) then accept the change, otherwise reject it.

Following this rule iteratively will generate a sequence of configurations representative of all the possible configurations for a given temperature T if the method used to generate configuration B is sufficiently flexible and not biased among configurations. This general scheme for simulated annealing is called the Metropolis algorithm [9].

Metropolis can be applied to situations which are obviously not real and yet produce exacly correct answers. For example, one way to generate configuration B is to change the identity of one particle from one element to another, i.e. transmutation. Even though transmutation does not actually occur readily in a real system (i.e. it is not a real kinetic pathway for an atom of another element to occupy an atomic site), as long as some pathway exists which can cause this switching (e.g. solid state diffusion) the comparison of the two states is valid and the relative probabilities of both states is equal to exp(-ΔE /kT) .

2d. Iterators in kSan

The application *kSan* treats the iterator as a property of the simulation, and is implemented as an AppleEvent object. Each iterator class inherits from the basic iterator class, and it required to provide functions that transform the current configuration of particles into a new configuration. different iterator can be implemented as plug-ins to the application.

3. Simulation Cell

Each simulation represents the particles in a small region of three-dimensional space. To accurately represent the conditions of being inside a solid, kSan uses periodic boundary conditions to avoid the existence of artificial simulation walls or edges. That is, replicas of the simulation cell appear at positions +x, -x, +y, -y, +z, -z, +xz, -xz, +xy, -xy, etc., where xyz are the vectors

of the cell edges. The simulation cell is a parallelpiped, which can be but does not need to be orthorhombic.

The cell volume is defined by the three vectors of its edges. Simulations can be run under various conditions which affect the cell size. A fixed cell size implies a constant volume simulation, and therefore the stress in the cell will be a function of the contents.

The cell can also be allowed to change size or shape in response to the pressure. Simple elongation or contraction in one direction is called simple strain. Note that simple strain on an axis which is not a principal axis can change the shape of the cell: i.e. if a cell starts out as orthorhombic, elongating on a cell diagonal will produce a non-orthorhobic cell. A cell can also be deformed so as to preserve its volume. This is called shear strain: an elongation in one axis accompanied by a contraction in an orthogonal direction. Strain can occur in six axes for a three dimensional cell: three simple strain directions (x,y,z) and three shear directions (xy, yz, zx) . Accordingly, there are six components of the stress which form the stress tensor.

As each simulation has only one cell, properties of the cell are treated as properties of the simulation itself. Constraints on the cell dimension, such as whether changes are allowed in each direction, whether shear strain is allowed and the magnitude of the applied stresses are also properties of the simulation. Each iteration, the cell size may change by some algorithm, and the object which implements this algorithm is also a property of the simulation, and can be changed. Cell resizer objects might use an algorithm based on a Monte Carlo method, as outlined above, a dynamical method, where the cell walls are treated as objects with mass which respond to forces, or some other algorithm.

4. Particles

Particles are elements of the simulation object. They have properties of position, velocity, energy, force, selected and atomtype. The position, velocity and force quantities are vectors, and are represented as lists of values. Position, velocity and atomtype can be modified by the iterator. These are also read / write properties in the object model. Force and energy are calculated by the interatomic potential, and so these are read-only properties

As a simulation can have thousands of particles, and the speed of a calculation depends on the efficient packing of the particle data in memory, particles are not created as distinct programmatic objects. Instead, their properties are cared for by the simulation object and any plug-ins. When an iterator object or interatomic potential object is called upon to perform a calculation, there are access functions provided to gain access to the arrays of position, velocity, energy, force, atomtype, etc.

The atomtype property is special, as this is a reference to another object:. Atomtypes are another class of object which can be created in the simulation, and the internal representation of the atomtype property in kSan in just a one-byte integer, which represents the index of the atomtype in the simulation's list of atomtypes.

When these properties are accessed from an AppleEvent, the event is handled by the simulation object, and a AppleEvent Object token is generated which refers both to the simulation object and to information sufficient to specify the particle(s) and the desired property. In order to make AppleEvent processing more efficient, the tokens can represent lists of particles as an

array of integer values indicating the indices of the particles.

In addition, plug-ins can define additional properties for the particles. A plug-in can register a particular property with the simulation object. For example, the energy minimization iterator defines a most-recent displacement for each particle. To expose this property to the scripting interface,. this iterator registers a four-byte code for this property with the simulation. If this property is involved in a GetData or SetData event, the event will be handed off to the iterator object.

5. Graphics displays

Part of the value of using MacOS for a simulation is that the visualization of the simulation can be performed in real time with the simulation. Any number of windows can be opened with a different kind of simulation visualization. Although the simulation can be done in three dimensions, it is seldom of value to create a three dimensional shaded representation of the particles themselves, as can be done with OpenGL. This is because the color can be more effectively used to illustrate some other property. Thus, we use Quick-Draw based graphics and draw the particles on the screen in reverse order of their distance from the viewer.

The algorithm used to decide on a color for each particle is part of the colorizer object which is the coloring method property of the window. Colors might be applied based on energy, stress, force, velocity, volume, or any other property which can be calculated from the simulation data. The windows also have properties of magnification particle size factor, view angle, and a list of particles which are visible within the window.

The examples in Figures3 through 6 provide examples of different coloring schemes. In each case, the color applied to the particles can illustrate different physical phenomena. In figure 3, one sees the spatial grouping of regions with the same crystallographic orientation: in essence this is a discretized visualization of the grain structure which can be seen under a microscope. Figures 4 and 5 show the energy and stresses associated with a grain boundary: The visualization by energy shows the spatial extent of energy fluctuation caused by the grain boundary. Visualization of the strain highlights the opposing forces of tension and compression.

The computational time to render this type of scene for the boundary is much smaller than the time taken to perform the actual calculation. Thus, real time visualization of simulations is a natural extension of existing simulations.

Because the quantitative data of the simulation is equally accessible from kSan simulations, no compromises are made in the ability to analyze data, On the other hand, insights gained from qualitative representations of the data can give simulations meaning which the numerical data cannot. This is especially true for educational applications, where teaching concepts is more important than extracting quantitative information.

Challenges for the Future

The recent past in Mac hardware development has provided the computational muscle to contemplate the kinds of simulations which were once reserved for supercomputers. The future of the MacOS is likely to bring compelling advances with CPUs based on multiprocessor architectures and a foundation which allows for multiple-CPU supercomputing clusters.

In general, parallelization of the algorithms used for materials simulations is straightforward. One of the simplifying assumptions of most simulations is that interactions between particles or gridpoints are local. Thus, domain decomposition can be used to spread tasks over multiple processors: the physical space modelled in the simulation by divided up among the CPUs, and message passing is used to communicate between CPUs about the changes in the conditions of the border regions. Message passing overhead can be reduced to a small fraction of the whole simulation effort for systems with a moderate number of CPUs and a large physical space, because the ratio of the amount of border to amount of space will be small.

There is a continuing challenge to match these calculation techniques with real time visualization. Current parallel processing techniques generally ignore visualization not to mention any sort of human interface, in an attempt to optimize the calculation. This offers a significant entry point for new software which can capitalize on both the processing power of MacOS and its best-of-class user interface features.

Conclusions

Simulation is rapidly becoming a standard tool for materials scientists to understand the behavior of materials on an atomic scale and on the microscale. The integration of these simulation tools with the human interface of the MacOS promises to offer a powerful combination of usability accessibility and computational power.

Bibliography

1. M. Sarikaya, J. Liu, and I. A. Aksay, "Nacre: Properties, Crystallography, Morphology, and Formation," in *Biomimetics: Design and Processing of Materials*, eds. M. Sarikaya and I. A. Aksay, AIP Press, Woodbury, NY, 1995, pp. 35-90.

2. Jean E. Taylor, John W. Cahn, and Carol A. Handwerker. “Geometric models of crystal growth” *Acta Metallurgica Materialia*, 40:1443-1474, 1992.

3. J.E. Lennard-Jones, “The Determination of Molecular Fields” *Proceedings of the Royal Society (London)* 106A 462 (1924).

4. J. Tersoff, “Empirical interatomic potential for silicon with improved elastic properties” *Physical Review* B **38** 9902 (1988).

5. M. I. Baskes, J. S. Nelson, A. F. Wright, "Semiempirical modified embedded-atom potentials for silicon and germanium", *Physical Review* B, 40, 9, 1989, 6085.

6.M.S. Daw, S.M. Foiles and M.I. Baskes, “The embedded atom method: a review of theory and applications” Materials Science Reports 9 251 (1993).

7. J.M. Haile, *Molecular Dynamics Simulation*, John Wiley & Sons, NewYork, (1992).

8. L. Verlet, “’Computer Experiments’ on Classical Fluids”, *Physical Review* 159 98 (1967).

9. M.E.J. Newman and G.T. Barkema, *Monte Carlo Methods in Statistical Physics,* Oxford University Press, London, 1999.