MOLECULAR DYNAMICS SIMULATIONS OF NANOSCALE MECHANICAL PROCESSES

by

SIGURD WENNER

THESIS

for the degree of

MASTER OF SCIENCE

(Master in Computational Physics)

Faculty of Mathematics and Natural Sciences
Department of Physics
University of Oslo

Sept. 2010

Det matematisk-naturvitenskapelige fakultet
Universitetet i Oslo

Contents

Chapter 1

Introduction

Forces at work: electromagnetism! Neglect gravity and strong force. QED is the only thing! Elimination of degrees of freedom. Relativity, ..

Irreversibility!

Tribology! Adhesion is important in stick-slip motion (low sliding velocity).

...

The next chapter presents what is to be studied, nanoscale mechanical processes. How the systems are built up and what will happen to them is discussed. Details concerning the two materials, silicon and sodium chloride, is also mentioned. Chapter 3 presents the method for the study, molecular dynamics simulations. It explains how classical mechanics and thermodynamics can be applied on the nanoscale in order to predict the global behaviour of systems consisting of thousands of atoms. Chapter 4 shows how I try to model the interactions between atoms, and how to efficiently compute these interactions. This is the chapter with the most mathematical formulas and fewest pictures.

Moving on to more practical matters, chapter 5 presents the numerical setup. It contains information regarding how the system is represented inside my computer program, and some numerical methods used to treat the systems in a physically realistic ways. Chapter 6 contains technicalities of the program and shows how the code is structured for flexible and efficient calculations.

The results of the numerical simulations are presented in chapters 7 and 8, corresponding to the two main classes of systems mentioned in the next chapter. Finally, I put the results into perspective and make some concluding remarks regarding the outcome of this project and possible improvements and continued work.

Chapter 2

Systems of interest

In this project, I will adress the dynamics of solid state structures on the nanoscale. This requires taking into account the fact that matter is composed of atoms. The mathematical theory for the behaviour of atoms, quantum mechanics, was developed by Einstein, Bohr and de Broglie, amongst others. Rutherford and Thomson contributed with experimental verifications of the atomic nature of matter, and during the 20th century, quantum mechanics and atomism became established as proven facts. Modelling the behaviour of atoms with computationally efficient methods was and continues to be another challenge. This problem is approached in the next chapter.

The systems under study are bulk materials and surfaces. Solid state materials have a periodic structure referred to as crystalline. Perfect crystals are composed of a collection of atoms, the *basis* of the crystal, repeated in three directions. I will only consider cubic structures, where these directions are orthogonal and parallell to the cartesian axes of my coordinate system. Being cubic, the *unit cells* containing the basis being repeated is equal in extension in the three cartesian directions. In addition to the translational symmetry of the crystal, cubic structures have inherent rotational and reflectional symmetries. Some of these are broken when thet basis being repeated consists of more than one atom.

In any case, a real crystal is never perfect. All solids contain point defects and dislocations. A battle against these imperfections is futile except at very low temperatures. Even though a perfect crystal is the state with the least potential energy between atoms, a crystal with defects has a greater entropy, making it a more probable structure. Another type of defect is thermal vibrations. A crystal with finite temperature is always the subject of collective vibrations, known as *phonons*. These are waves of elastic energy with different frequencies propagating through the crystal, enabling the conduction of heat and sound. Atoms can also vibrate more or less independently of each other. The theory behind these two vibration phenomena have been used to develop theories for estimating the heat capacity of solids, the Debye and Einstein models, respectively.

My systems can be divided into two classes, bulk and surface systems. Fractures and other types of mechanical failure are the processes I bring under study in bulk solids. In the second case, interfaces between solids and how surfaces interact are the

matters of interest. The analysis is performed for two types of materials, to investigate the effect of different chemical bond types on these phenomena. I have chosen an ionic material, sodium chloride, and a semimetal with covalent bonds, silicon.

2.1 Fracture mechanics

It is a well-known fact that the usual reason for fracture in a solid state material is its defects. As discovered by A. A. Griffith in 1921, the only significant non-material-inherent quantity that influences the breaking stress is the linear size of the biggest flaw. By this, Griffith meant the length of a crack inside the material, with direction normal to the direction of inflicted stress. As I in my simulations apply periodic boundary conditions in such directions, the flaw is a void shaped as a sphere or cylindrical disk. The position of these flaws is not important due to translational symmetry, so they are always placed in the middle of the simulation box, for convenience when visualizing the structure. I use the term flaw size when referring to the radii of the voids.

Griffiths relation predicts a proportionality between the tensile stress at which a fracture occurs, σ_f, to the flaw size a: REFERANSE

$$\sigma_f \propto \frac{1}{\sqrt{a}}. \tag{2.1}$$

The expression for the proportionality constant was found by Griffith and improved by G. R. Irwin. The more general form used by Irwin is REFERANSE

$$\sigma_f \sqrt{a} = \sqrt{\frac{(2\gamma + G_p)E}{\pi}}. \tag{2.2}$$

Here, γ is the surface energy density in the material. When a surface is created inside the material, some potential energy corresponding to γ is lost. G_p is the energy dissipation to elastic waves and heat. The value of the parenthesis is the approximate total energy which is converted between different forms in the fracture process. Finally, E is Young's linear elasticity modulus of the material.

The strain ϵ and tensile stress σ can be measured in experiments and simulations in order to test that the stress reproduces Eq. (2.1), and at what strain fracture is achieved. The relation in Eq. (2.2) has proven a reliable approximation for both brittle and ductile materials. For brittle materials, the surface energy term dominates, while in ductile materials, the dissipation term is the most significant.

According to Hooke's law for elastic materials, there is a linear relationship between applied stress and resulting strain, when these quantities are sufficiently low. The proportionality constant is Young's modulus E:

$$\sigma = E\epsilon. \tag{2.3}$$

using this relation, it is an easy task to estimate E. Although it is not included it in this project, estimation of γ and G_p is also possible, so that Eq. (2.2) may be verified.

2.2 Material interfaces and friction

Friction between materials is a phenomenon which have been observed and measured for hundreds of years, but which have not yet been fully understood from basic principles. The problem is that it is a quite complex process and there are different questions that have to be asked and answered on different length scales. The only well-known law of friction that approximately describes macroscopic materials is the one formulated by both DaVinci, Amontons and Coloumb. It states that the force of friction is neither dependent on the sliding velocity v or object area A, but proportional to the normal force (or load) L:

$$F = \mu L. \tag{2.4}$$

The proportionality constant μ is called the friction coefficient. It is material dependent and most often takes values between 0.2 and 0.8.

Even if one sticks to the macroscopic scale, researchers have discovered several complications by the use of experiments and computer models. One needs for instance to consider the coupling between the mechanical process and heating of both surfaces in contact. Important mechanical phenomenons that appears when sliding two rough materials against each other are fractures and slips near the material interfae [1].

A general observation done in the course of the last decades is that the friction force indeed varies with the contact area between the materials. This is, however, the real area of contact A_C, which is much smaller than the geometrical area A of the material slabs. As surfaces have a roughness on all scales, only a very few asperities make contact to counteract the load L (see Fig. 2.1). Ringlein and Robbins [2] provide a refined friction formula based on their research on friction,

$$F = \mu L + cA_C. \tag{2.5}$$

They also point out that reason that Eq. (2.4) often work well for macroscopic materials is that the real area of contact A_C often is proportional to the load L, giving apparently only a load dependece.

Digging deeper into the origins of friction requires understanding of the forces between microscopic asperities, which I attempt to model in this thesis. This have been investigated by measuring the stress between a sphere and a plane or between two spheres of some type of atoms [3, 4]. Sphere segments are a good approximation of the form small tips can have on a microscopic scale. As sliding friction is complicated and time consuming to model, I focus mostly on investigating contact stress and adhesive forces for an ionic material and a semimetal. The distribution of stress between two spheres and between a sphere and a surface should have the same form, due to symmetry. The stresses will differ only by a multiplicative constant. In my case, the latter of these systems is the easiest one to model.

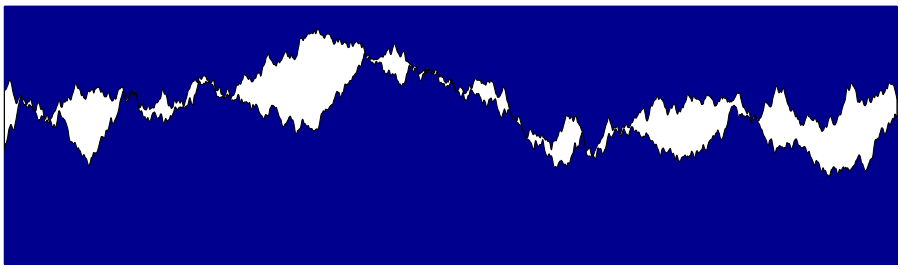


Figure 2.1: How the interface between appearantly flat material surfaces can look like on a small scale. Modelled using smoothed random walks.

2.2.1 Contact area

Defining the area of contact between materials becomes problematic on the atomic scale. How big is the contact area between an atom and a surface of atoms? This could be spproximated by a geometric approach of finding an area enclosing all atoms close to the surface, but such a method will require too many parameters and will not be reliable enough for my purpose. It will be more effective to assign a cross-sectional area to an atom based on how far it is from the surface atoms [4]. The method I use is based on this idea.

The area of contact A_C is measured in my simulations by calculating the potential energy between particles belonging to the sphere and the surface, U. This energy is then divided by the surface energy per area of bulk material, σ_0, to obtain the real area of contact:

$$A_C = \frac{U}{\sigma_0}. \tag{2.6}$$

The constant σ_0 is material dependent and must be measured in a computational experiment. In order to to this, I simultated bulk crystals with PBC in the y and z directions and walls of fixed particles in the x direction. The simulation was performed at a temperature of 10 K for the potential energies of the atoms not to have too great deviations from the minima. I then defined σ_0 as the calculated potential energy between simulated particles and left side fixed particles, divided by the area $L_y L_z$.

My results are $\sigma_0 = -0.2471$ eV/Å^2 for silicon and $\sigma_0 = -0.02522$ eV/Å^2 for sodium chloride. The NaCl calculations were done with four layers of unit cells with fixed particles on the sides of the simulation box (eight atomic layers). The numbers show no deviation when changing the system size up or down from 4096 particles.

When pushing a sphere and a surface of equal materials together, the A_C vs t graph is continous and seems to describe well what is going on. The system does not reach the same equilibrium state for slightly different initial conditions (because of a different random generator seed). The atoms in the interface will often be disordered and able to fall into many local energy minima. In such types of experiments, a contact area measurement will have a low precision and is only used to extract qualitative

information about trends for different cases.

2.3 Materials

The processes which I have described are qualitatively and quantitatively different for different materials. This is a result of their differing type of interatomic forces (chemical bonds) and crystal structure. These are affected by more important properties, as the number of elementary particles of each type inside the atoms. Some details characterizing the two materials under study are given below.

2.3.1 Sodium chloride

Sodium chloride is a solid composed of Na^+ and Cl^- ions forming an ionic bond lattice. The mean mass of sodium atoms and chlorine atoms are 22.9898 amu and 35.453 amu, respectively [5]. These numbers are achieved by averaging over the different isotopes of the chemical elements, weighted by their relative occurence in the universe. Sodium chloride has a density of 2165 kg/m^3, corresponding to a mixed particle density of 0.045 Å^{-3}.

The most stable crystal structure of sodium chloride is the double face centered cubic structure. Each of the ions are placed in a Bravais FCC lattice, displaced with a vector $\mathbf{u} = \frac{a}{2}\mathbf{e}_x + \frac{a}{2}\mathbf{e}_y + \frac{a}{2}\mathbf{e}_z$, where a is the lattice constant. The basis used in my program is arranged so that surfaces parallell to the cartesian axes will be charge neutral when copying a unit cell (visible in Fig. 5.4). a has the value 5.63 Å [6]. The conventional unit cell has four atoms of each type. One filled unit cell of a NaCl crystal is shown in Fig. 2.2.

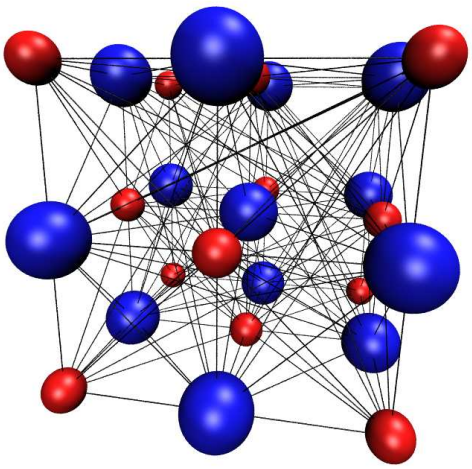


Figure 2.2: Conventional double face centered cubic unit cell with Na^+ and Cl^- ions.

For the interactions amongst sodium and chlorine ions, I use the LJ and Coloumb potentials (section 4). These will imitate the very short-range nuclear repulsion and the ionic bonding, respectively. The short-range Coloumb interaction is strong enough that the attractive force of the LJ interaction is almost negligble in the dynamics. The most important role of the LJ potential is stopping equally charged particles from crashing into each other. The LJ parameters can be found in section 4.1.

2.3.2 Silicon

Silicon is a semi-metal which forms covalent bonds between the atoms. The mean mass of a silicon atom is 28.0855 amu [5]. Its solid phase density is 2329 kg/m^3, corresponding to a particle density of 0.050 Å^{-3}.

The most stable crystal structure of silicon atoms is the diamond structure. The atoms are placed in two Bravais FCC lattices, displaced with a vector $\mathbf{u} = \frac{a}{4}\mathbf{e}_x + \frac{a}{4}\mathbf{e}_y + \frac{a}{4}\mathbf{e}_z$. The conventional unit cell has eight atoms and a lattice constant 5.43 Å [6]. Figure 2.3 shows one filled unit cell of a diamond crystal consisting of silicon atoms.

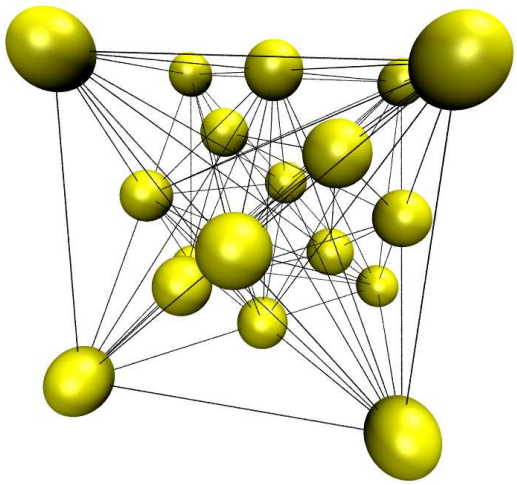


Figure 2.3: Conventional diamond unit cell with Si atoms.

The silicon atoms interact with the specialized SW potential (section 4.3), which should give a highly accurate description of the covalent bonds. The angular dependence in this potential is the reason for the equilibrium diamond structure. Silicon has a remarkable feature which it shares with water. The liquid phase has a higher density than the solid (crystalline) phase. Thus, a silicon crystal will melt when compressed to a sufficiently low density.

In summary, I am considering systems of Si and NaCl lattices undergoing mechanical deformations. These deformations are described in detail in section 5.2. Most of

the important quantities measured are energies which must be fed to a system in order to deform it or energy exterted by a system on its surroundings. Structural organization in the form of various-scale defects are also important to investigate, because it influences these energies and how the mechanical processes are taking place.

Chapter 3

Basics of molecular dynamics

Atoms are objects which have an extension in the order of one Ångström, 10^{-10} meters. They are composite of electrons and a nuclei which is also consists of more elementary particles. This is, even when classical mechanics is used to describe the motion of particles, a complicated system. And we know that what must be applied to systems of this small size is quantum mechanics. All elementary particles are quantum fields with an associated wavefunction and a quantized energy. In this project, I will study systems containing thousands of atoms. *Ab initio* calculations, containing only strictly controlled approximations, no parameters and a fully quantum mechanical description of the system, is no feasible approach.

The Born-Oppenheimer approximation says that the nucleus of an atom has a much greater mass than the electrons, and will stand perfectly still in a electronic time scale. The degrees of freedom of the electrons and the nucleus are separated. The energy of a collection of atoms can be calculated using only the electronic wavefunctions, with e.g. density functional theory (DFT). This enables calculation of forces on nuclei, and the atomic movements can be simulated. Such a DFT-MD coupling is very accurate, but also extremely time-consuming. More direct methods have also been developed, such as the Car-Parinello quantum molecular dynamics. Classical molecular dynamics (MD) takes the approximations one step further and considers not only nuclei, but entire atoms as point particles. Classical mechanics describe the motion of the atoms, but the potential between the atoms tries to take into account their electronic structure (more of this in section 4). The interatomic potentials can be improved to the point that the resulting dynamics is equal to what one would get using an ab initio approach.

Predicting values of macroscopic observables for homogenous matter in equilibrium has been the most important application of MD. This requires coupling the dynamics to certain external thermodynamic conditions. All ensembles of statistical mechanics can be simulated in some artificial way in order to study systems in various conditions. Properties of bulk matter is of great interest. Properties of phase transitions and pair correlation functions can be used to learn much of how materials behave on a microscopic scale. To a certain extent, dynamical processes involving the system as a whole can also be studied. Simulations of fracture, friction, corrosion,

diffusion and catalysis are examples of areas of earlier and current research.

If a system is in equilibrium, the ergodic hypothesis applies. In the MD approximation, a system can be uniquely determined by the positions and velocities of its contained atoms. The *phase space* of the system is a $6N$-dimensional cartesian coordinate space with the position and velocity components of all the N particles as variables. The phase space density $\rho(\{\mathbf{r}_i\}, \{\mathbf{p}_i\})$ determines the probability for the state of the system to reside at a certain point in phase space. The form of this function depends on which ensemble we are working in. The ergodic hypothesis says that, when a system is observed for a very long time, the distribution of states the system has been in coincides with the phase space density. Vaguely put, the system explores every region of its phase space. This implies an important fact about averages. Given an observable variable A, the ensemble average and time average are, respectively,

$$\langle A \rangle = \int \rho(\{\mathbf{r}_i\}, \{\mathbf{p}_i\}) A \mathrm{d}\omega, \tag{3.1}$$

$$\bar{A} = \lim_{t\to\infty} \frac{1}{t} \int_0^t A \mathrm{d}t. \tag{3.2}$$

since the distribution of the phase space variables will be equal in the ensemble and in time, these averages must be equal. Simulating the dynamics of the system and taking the time average over a reasonable long time therefore produces a representative equilibrium value of any variable A. If we were to calculate an ensemble average, we would need to know the form of the phase space density and evaluate a $6N$-dimensional integral, which is highly implausible.

3.1 Equations of motion

Lagrange's and Hamilton's formulations of classical mechanics are often used in molecular dynamics derivations. This is useful when simulating e.g. rigid molecules with constraints on the atoms' relative motion, using generalized coordinates in addition to cartesian ones. As I will not consider molecules in this project, but crystalline solids, I will stick to the simpler Newtonian formulation. In a collection of atoms i, the equation responsible for the dynamics is Newton's second law,

$$m_i \frac{\mathrm{d}^2 \mathbf{r}_i}{\mathrm{d}t^2} = \mathbf{F}_i, \tag{3.3}$$

where m_i is the mass of atom i, $\mathbf{F}_i$ is the force vector exerted on atom i by other particles or external conditions, and $\mathbf{r}_i$ is the position of the atom. All forces are dependent on the positions of all the atoms in the system.

The force acting on a particle with index i is the negative gradient of the potential energy associated with that particle, U_i. In the usual case of interaction pairs, this equals the sum of the negative gradient of interaction potentials U_{ij} from other particles j,

$$\mathbf{F}_i = -\nabla_i U_i(\mathbf{r}_i) = -\sum_{j\neq i} \nabla_i U_{ij}(\mathbf{r}_{ij}). \tag{3.4}$$

The meaning of ∇_i is the gradient with respect to the position of particle i. $\mathbf{r}_{ij}$ is shorthand for $\mathbf{r}_i - \mathbf{r}_j$. By Newton's third law, $\mathbf{F}_{ij} = -\mathbf{F}_{ji}$. By exploiting this, only half of the possible potential gradients need to be calculated when finding the forces on all particles. In a system of N particles, this means $\frac{1}{2}N(N-1)$ force terms must be calculated in a straightforward approach.

3.1.1 Time integration

In most cases, to integrate Newton's second law (Eq. (3.3)), molecular dynamics programs use *symplectic integrators* like the Verlet and Leapfrog methods [7]. All integrators conserve momentum exactly and energy approximately. Symplectic integrators are time reversible and also conserve phase space ($\{\mathbf{r}_i\}$-$\{\mathbf{p}_i\}$-space) volume between trajectories.

My program uses the velocity Verlet method, because of its numerical stability and ability to efficiently estimate the velocity and positions of particles at simultaneous times. Given initial positions, velocities and forces, the values of these quantities at the next time step is

$$\mathbf{v}(t + \Delta t/2) = \mathbf{v}(t) + \frac{\mathbf{F}(t)}{2m}\Delta t, \tag{3.5}$$

$$\mathbf{r}(t + \Delta t) = \mathbf{r}(t) + \mathbf{v}(t + \Delta t/2)\Delta t, \tag{3.6}$$

$$\mathbf{F}(t + \Delta t) = -\nabla U(\mathbf{r}(t + \Delta t)), \tag{3.7}$$

$$\mathbf{v}(t + \Delta t) = \mathbf{v}(t + \Delta t/2) + \frac{\mathbf{F}(t + \Delta t)}{2m}\Delta t. \tag{3.8}$$

The atom index i is dropped for brevity.

The time step should be set so that the thermal oscillation of atoms are well-described by the discrete dynamics. In my simulations, I put the time step to 2-4 fs (10^{-15} s), and one oscillation goes over approximately 20 time steps. An atom with room temperature thermal velocity will use about 100 time steps to reach from its own place to an adjacent atom's position. This ensures that atoms do not get too close to each other, preventing unphysically large forces.

3.2 Thermodynamic ensemble

The most fundamental principle of classical mechanics is conservation of energy. This will be approximately satisfied by discrete time integrators. Therefore, particles simulated using only such a time integrator will constitute a microcanonical (NVE) ensemble.

In macroscopic experiments, it is generally easier to keep the temperature constant istead of the energy. This is the most prominent reason for using the canonical (NVT) ensemble. This ensemble can be simulated by using methods popularly referred to as *thermostats*.

According to the *equipartition principle*, every degree of freedom which is quadratic in position or speed will contribute with an average energy of

$$E_{\text{dof}} = \frac{1}{2}k_B T \tag{3.9}$$

to the total energy, where k_B is the Boltzmann constant and T is the temperature of the system. This applies to a canonical ensemble in thermodynamic equilibrium. This formula can be summed over all kinetic degrees of freedom, and inverted to give an estimate of the system temperature. As the total translational kinetic energy is $U_k = \frac{1}{2}\sum_{i=1}^{N} m_i v_i^2 = \frac{3}{2}Nk_B T$, we get

$$T = \frac{1}{3Nk_B}\sum_{i=1}^{N} m_i v_i^2 \tag{3.10}$$

in three dimensions. The temperature T is estimated only from the translational degrees of freedom, while the degrees of freedom associated with potential energy are left out. Measuring the temperature exactly requires the usage of the fundamental thermodynamic relation,

$$\mathrm{d}U = T\mathrm{d}S - P\mathrm{d}V + \mu\mathrm{d}N. \tag{3.11}$$

This shows how the internal energy U changes when the entropy S, volume V and particle number N varies. Finding the temperature requires differentiating the energy with respect to entropy, which is non-trivial to calculate for general systems. Therefore, we assume equilibrium between the translational and potential degrees of freedom.

The temperature estimation can be used to verify how a thermostat performs over time. Regardless of the thermostat used, all velocities are initialized to random values obeying the Maxwell-Boltzmann distribution, with the standard deviation $\sqrt{\frac{T_{\text{target}}}{m_i}}$. As a simulation starts, kinetic energy will be transformed to potential energy, and the measured temperature will (in most cases) drop, if a thermostat is not applied.

3.2.1 Thermostats

One way of keeping the temperature constant is to use a simple rescaling factor. All the particle velocities are multiplied by a factor α so that $T \to \alpha^2 T = T_{\text{target}}$. This rescaling factor is easily found from Eq. (3.10) to be

$$\alpha = \sqrt{\frac{3Nk_B T_{\text{target}}}{\sum_{i=1}^{N} m_i v_i^2}}. \tag{3.12}$$

This rescaling will give the system the correct temperature, but it will not sample the Maxwell-Boltzmann-distribution correctly for all degrees of freedom, and does therefore not satisfy the ergodicity requirements of the canonical ensemble.

Many different thermostats have been proposed to fix this problem, but all have their weaknesses [7]. The Andersen thermostat replaces the velocities of random particles with velocities from the Maxwell-Boltzmann distribution. The Berendsen and Nosé-Hoover thermostats add a ficticious friction force which gradually decrease or increase the particles' velocities to get the correct temperature.

The Andersen thermostat is known to work well for equilibrating systems, although it should not be used when simulating dynamics. The algorithm is strinkingly simple. Define a collision time τ, significantly larger than the time step Δt, after which all particles in the system on average have exchanged energy with the heat bath. For each time step, the probability of replacing a particle's velocity by a Maxwell-Boltzmann distributed one is given by $\Delta t/\tau$. The energy absorbed from the heat bath is calculated by simply computing the kinetic energy before and after the thermostat has been used.

The thermostat I will use during dynamical processes is one recently developed by Bussi, Donadio and Parrinello [8] (I will refer to it as the BDP thermostat). It contains a friction term and a stochastic force, like in Langevin dynamics for e.g. solvent particles. In its original infinitesimal form, the change in kinetic energy due to the termostat is

$$\mathrm{d}U_k = \frac{U_k^{\text{target}} - U_k}{\tau}\mathrm{d}t + 2\sqrt{\frac{U_k U_k^{\text{target}}}{3N\tau}}\mathrm{d}W. \tag{3.13}$$

The first term is the one used in the Berendsen thermostat. The second term is the stochastic term, which can be shown to sample the canonical distribution for kinetic energy. $\mathrm{d}W$ is a infinitesimal Wiener process path element. The parameter τ defines a relaxation time for the thermostat, that is, how fast the system is being cooled or heated. The equation must be integrated using stochastic integration, and in the paper, the resulting rescaling factor is found:

$$\alpha^2 = D + \frac{T_{\text{target}}}{3NT}(1-D)\sum_{i=1}^{3N} R_i^2 + 2\sqrt{\frac{T_{\text{target}}}{3NT}D(1-D)}R_1, \tag{3.14}$$

where $D = e^{-\Delta t/\tau}$, $\{R_i\}$ are $3N$ Gaussian distributed random values, and R_1 is just one of these.

Velocities are rescaled by simply setting $\mathbf{v}_i \to \alpha\mathbf{v}_i$ for all simulated particles i. The result of using the BDP thermostat is ergodic sampling of the canonical distribution (when in equilibrium), undisturbed dynamics (which is important in my case) and tunability by changing the τ parameter. Figure 3.1 shows a histogram of the velocity distribution of particles after equilibration with the BDP-thermostat. Generating all the random values is a tedious task, so this thermostat will be more CPU-intensive than the others, but the workload will be negligble in comparision to the calculation of forces between particles.

For one velocity rescaling, the energy absorbed from the heat bath is $U_k(\alpha^2 - 1)$, where U_k is the total kinetic energy of the system. Assuming that the velocities are

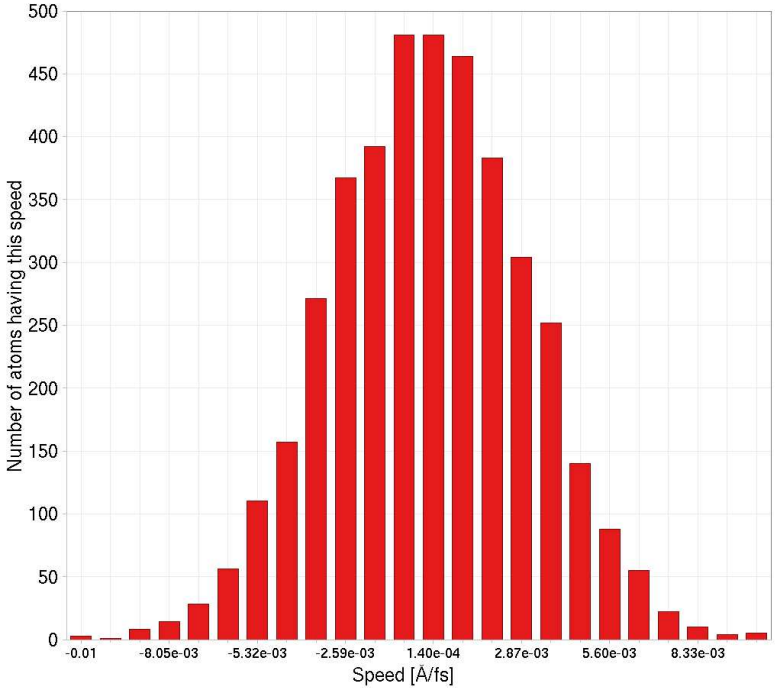


Figure 3.1: The distribution of one velocity component for 4096 particles in a simulation using the BDP thermostat.

rescaled every time step, the power exerted on the system by the heat bath is

$$P = \frac{1}{\Delta t} U_k(\alpha^2 - 1). \tag{3.15}$$

3.3 Equilibration

It is crucial to equilibrate a molecular dynamics system before measurements can be made or dynamics can be simulated. The standard approach I use is to simulate the dynamics in a usual fashion with a heat bath coupling to all the particles using the BDP thermostat. However, some systems are harder to equilibrate than others and require special treatment.

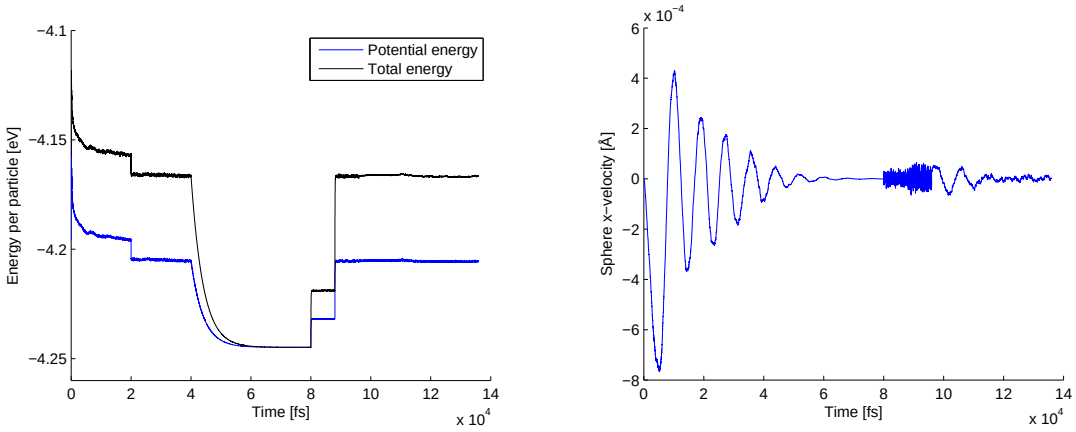
In the sphere-surface interface experiments, it is hard to keep the sphere from bouncing, rolling and vibrating internally after being pushed down towards the surface. This type of computational experiments therefore requires an extensive equlilibration epoch. An example of a method to equlilibrate this system is the following phases:

1. Initial dynamics, letting the sphere "fall" onto the surface so a contact is established. Using the BDP thermostat with global coupling. Ended when the sphere is slightly compressed and ready to start an vertically oscillating bounce motion.

2. Energy dissipation phase, adding a force term $-\gamma \mathbf{v}_i$ to all particles to simulate viscocity. The friction parameter is chosen so that $\gamma \Delta t / m_i \ll 1$. To avoid requiring a "hard" implementation of this in the equation of motion integration, the velocity of the previous time step is used in the force, so the dynamics will have an error comparable to that of the Euler scheme for integrating the equations of motion. The temperature is decreased and the oscillations die out gradually, optimally making the system totally static. No thermostat is used in this phase.

3. As velocity rescaling would have restarted the oscillations, the Andersen thermostat is now used to make the atoms vibrate more independently. I often divide this phase into two phases where the heat bath temperature is increased abruptly from 50-100 K to e.g. 300 K.

4. The sphere should be in equilibrium with the heat bath of the desired temperature, so the data gathering phase can start. The BDP thermostat is used in a heat conduction simulation form (see section 5.4). Collective vibrations will reappear in the sphere, but these are naturally occuring phonons, and have an amplitude which is comparable to the thermal fluctuations of independent particle positions.



(a) (b)

Figure 3.2: Time development of energy and collective velocity of a big Si sphere pushed towards an Si surface during all simulation phases.

Figure 3.2 shows the energy and center of mass velocity in the x direction during a typical silicon adhesion experiment. The initial drops in potential energy are caused by introduction of a gravitational field of discontinually increasing strength. During the dissipative dynamics phase, both the kinetic and potential energy reach

a minimum where the sphere is frozen (at zero temperature). Subsequently, the Andersen thermostat heats up the system in two phases by giving them independent random kinetic energies. The transition to the data gathering phase where the BDP thermostat is used is seamless.

The center of mass velocity x component show that the sphere oscillates as it hits the fixed particle surface, as expected. These oscillations are killed by the dissipative dynamics, and there is no collective motion apart from fluctuations when the Andersen thermostat is applied. The rightmost section of the graph shows the combination of the mentioned phonons and fluctuations caused by particles "going against the flow" because of thermal excitations (the phonon oscillations do not die out at the end, even if it appears so in this particular case). This is how a crystal should behave in equilibrium, and good conditions for gathering e.g. mean stress data.

Molecular dynamics is beautifully simple method. Its main principles can be explained to high school students without having to mention quantum mechanics, even though it is responsible for everything that happens. In fact, our coupled differential equations, which I call equations of classical mechanics, can be derived as the average behaviour of a quantum system. Using the canonical quantum momentum operator $\hat{p} = -i\hbar\nabla$ and a wavefunction description of a particle, Ehrenfests theorem says that [9]

$$\frac{\mathrm{d}\langle p\rangle}{\mathrm{d}t} = \langle -\nabla V\rangle\,. \tag{3.16}$$

This is equivalent to Newtons second law for average dynamical quantities. Of course, atoms are small enough that averages does not tell the whole story. Atoms, and even big molecules have a wave-like nature which allow them to bend around corners and diffract through small gratings, in the same way as electromagnetic radiation. However, in collections of thousands of atoms, averages measured in simulations using the Born-Oppenheimer approxmation are reliable enough for most purposes.

I have explained the most essential features of the MD method, which have been in use for many decades. In addition to the atoms directly simulated by time integration, effects of the existence of the rest of the universe is also taken into account. This is done in subtle ways, as small amounts of energy transfered into or out of the system. Energy exchange is the most useful for these types of simulations, but including volume changes and particle exchange in equilibrium is also possible, in order to simulate the pressure canonical and grand canonical ensembles.

Chapter 4

Interaction potentials

Molecular dynamics is a classical theory, but tries to describe quantum mechanical phenomena with specially constructed interaction potentials for the atoms. The electronic degrees of freedom are not taken into account explicitly in the dynamics, even though both the nucleus and the electrons effect how atoms will interact with each other. Using quantum mechanical energy calculations or measurements from scattering experiments, researchers have tried to create approximate potential forms which will give the correct dynamics even though only classical mechanics is used. These potentials usually contain parameters which are determined by trial and error. When the parameters give the same macroscopic behaviour for a material as measured in experiments, they are accepted.

The interaction potential between two atoms will be determined by functions which have a continous dependence only on the distance between these atoms. The total potential energy is the sum of interaction potentials between all pairs of atoms. In the case of periodic boundary conditions (see section 5.3), this sum also goes over infinitely many copies of the atoms in the system,

$$U = \sum_{\mathbf{R}} \sum_{i=1}^{N} \sum_{j=1}^{i} U_{ij} \left(\| \mathbf{r}_{ij} + \mathbf{R} \| \right). \tag{4.1}$$

Here U_{ij} is the two-particle interaction potential and $r_{ij} = \sqrt{\mathbf{r}_{ij} \cdot \mathbf{r}_{ij}}$ the distance between particles i and j. $\mathbf{R}$ represents all translation vectors to copies of the system ($\mathbf{R} = n_x L_x \mathbf{e}_x + n_y L_y \mathbf{e}_y + n_z L_z \mathbf{e}_z$). Particles don't interact with themselves, so the $i = j$ term is skipped if $\mathbf{R} = 0$.

The force exerted on particle i by particle j is

$$\mathbf{F}_{ij} = -\nabla_i U_{ij} = -\frac{\partial U_{ij}}{\partial r_{ij}} \frac{\mathbf{r}_{ij}}{r_{ij}}. \tag{4.2}$$

In covalent bonds, the potentials also have angular dependencies because the electron densities are not spherically symmetric around atomic nuclei. An easy way to model this is by including three-body forces. The Stillinger-Weber potential (section 4.3) is one example.

4.1 The Lennard-Jones potential

One of the simplest and most sucessful interatomic potentials constructed is the Lennard-Jones (LJ) potential. It contains a repulsive part caused by the Pauli exclusion principle, which must be taken into account when electron wavefunctions have a significant overlap. It also contains an attractive part, because electron densities are slightly higher between a pair of atoms than elsewhere, inducing dipole moments which interact. This is the theoretical interpretation of the potential's mathematical form, but originally, it has roots in experimental studies.

Scattering experiments are usually carried out with the purpose of finding either an interaction cross-section for a pair of objects or a *structure factor* $S(\mathbf{k})$ for one object. For example, high energy photons, electrons or neutrons can be used to find the structure factor of a crystal, and neutrons and even neutrinos are fired at atomic nuclei to find its structure factor. If this function is inversely Fourier-transformed, the form of the potential around the object can be found. Atomic scattering is done in a slightly different way. We are interested in the interaction potential between two atoms, so they have to be fired at each other and collide. As free composite particles, the atoms will act as plane waves, but after a collision, the phase of these waves have changed. The total phase change will tell how stong potential fields the atoms have encountered. If the experiment is done with several different initial momenta $\mathbf{p}$, or equivalently, different wave vectors $\mathbf{k}$, the minimum distance between the atoms will change. The data of phase change for different minimum distances can be used to obtain an interatomic potential.

The LJ potential shows a good correspondence between underlying theory and experimental measurements. With optimal parameters, the potential is very close to the actual potential between pairs of noble gas atoms. The first one who used a continuous potential in a MD simulation was Rahman [10], who studied liquid argon using the LJ potential. It has become a standard to use this potential to model generic short-range repulsion for all sorts of atoms.

The form of the potential is

$$U_{ij}(r_{ij}) = 4\epsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right], \tag{4.3}$$

where ϵ_{ij} and σ_{ij} are parameters specific to the the different atomic elements in a simulation.

The derivative of this is straightforward to calculate, and gives rise to the force

$$\mathbf{F}_{ij} = -24\frac{\epsilon_{ij}}{\sigma_{ij}^2}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{14} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{8}\right]\mathbf{r}_{ij}. \tag{4.4}$$

Table 4.1 shows the two LJ parameters for interactions between pairs of equal atoms. The parameters for atoms of two different elements are calculated by two combination rules. Given parameters for interactions between atoms of the same

Atom	ϵ_{ii} [eV]	σ_{ii} [Å]	Source
Ar	$1.0318 \cdot 10^{-2}$	3.405	[10]
Na^+	$5.6478 \cdot 10^{-3}$	2.35	[11]
K^+	$4.3501 \cdot 10^{-3}$	3.33	[12]
Cl^-	$4.3501 \cdot 10^{-3}$	4.40	[11]

Table 4.1: LJ parameters for a few atoms and ions of interest.

element, homogenous interactions, geometric and arithmetic means give reasonable values for the parameters of heterogenous interactions [13]. The combination rules are

$$\epsilon_{ij} = \sqrt{\epsilon_{ii}\epsilon_{jj}} \qquad \text{and} \qquad \sigma_{ij} = \frac{1}{2}\left(\sigma_{ii} + \sigma_{jj}\right). \tag{4.5}$$

4.1.1 Implementation details

The simplest form of real space cutoff possible with periodic boundary conditions (see section 5.3.1) is the minimum image convention. Distances are in each direction calculated as the smallest one of x_{ij}, $x_{ij} + L_x$ and $x_{ij} - L_x$. In words, the particles are only interacting with the nearest of their copies. This is used in my program, but together with a neighbour list algorithm.

Neighbour lists, or Verlet lists, are lists of all particles that are closer to each other than a specified cutoff length. In my implementation, all particles i have their own lists of particles $j < i$ that are close to them. The force evaluations happen only between neighbouring pairs of particles, and the lists are updated every 10th timestep or so. The cutoff length must be set so that short range potentials like the LJ potential are negligble beyond it. I normally put the cutoff for this potential to 12 Å.

Regardless of how big the cutoff length is, cutting off the potential will introduce a discontinuity. There are many possible modifications of the LJ potential that avoid a discontinuity, but I will not go into this matter. The systems I am investigating will be strongly bound, and only the very short-range part of the LJ potential will make an important contribution to the dynamics.

4.2 The Coloumb potential

Electrically charged atoms, ions, interact through the Coloumb potential in addition to short-range quantum mechanically derived potentials like the LJ potential. The Coloumb potential is exact in classical theory, if the quantization of electromagnetic radiation is neglected. However, a more critical approximation is that of describing the abundant or deficient electron as a point charge. This must be done in order to complete the calculations in a reasonable amount of time. Therefore I model ions as point particles with unit elementary charge, $q = \pm e$. It is, though, possible to use

partial charges for ions and atoms to model inhomogenous concentrations of electrons (see e.g. [14]).

In reduced units (see section 5.1), the Coloumb potential between two particles with charge q_i and q_j is

$$U_{ij}(r_{ij}) = \frac{q_i q_j}{r_{ij}}. \tag{4.6}$$

Regardless of how simple it may look, this very important potential causes problems with PBC because of its long ranged nature. To see this, it is useful to write down the total electrostatic energy due to the Coloumb interactions,

$$U = \sum_{\mathbf{R}} \sum_{i=1}^{N} \sum_{j=1}^{i} \frac{q_i q_j}{\|\mathbf{r}_{ij} + \mathbf{R}\|}. \tag{4.7}$$

The symbol $\mathbf{R}$ plays the role of a generic translation vector of the simulation box, so that $r_{ij} = \|\mathbf{r}_{ij} + \mathbf{R}\|$ can point between any atom i in the box to any atom j within the infinitely many copies of the system when $\mathbf{R}$ is adjusted.

Even for the simple case of $N = 1$ in one dimension, we encounter infinities. The total potential energy will then be $2q_1^2\zeta(1)$, where $\zeta(n)$ is the Riemann zeta function. $\zeta(1)$ is the harmonic series, which barely diverges. For the case of every other particle on a line being positively charged and the rest negatively charged, the interaction energy of one particle will be an approximation to $2q_1^2 \ln 2$. Even if this is finite, the convergence rate is so slow that the required precision for this kind of calculations will not be achieved in a reasonable amount of time. So in all cases with periodic boundary conditions in one or more directions, the electrostatic energy of the particles can not be computed using a real space cutoff approach. Many methods have been used to solve this problem. In this project, I use the particle-particle Ewald summation technique, which is the simplest one.

4.2.1 3D Ewald summation

The Ewald summation technique is used to efficiently capture the long range interaction properties of the Coloumb interaction with a finite number of summation terms [7]. It can also be applied to other long range interactions like the van der Waals potential.

First, we need to express the pair potential as a Fourier transform:

$$U_{ij}(\|\mathbf{r}_{ij} + \mathbf{R}\|) = \frac{q_i q_j}{(2\pi)^3} \int \frac{4\pi}{k'^2} e^{i\mathbf{k}'\cdot(\mathbf{r}_{ij}+\mathbf{R})} \mathrm{d}^3\mathbf{k}'. \tag{4.8}$$

The integration goes over the entire reciprocal space. The following identity comes straight from the definition of the reciprocal lattice vectors $\mathbf{k}$:

$$\sum_{\mathbf{R}} e^{i\mathbf{k}'\cdot\mathbf{R}} = \frac{(2\pi)^3}{V} \sum_{\mathbf{k}} \delta^3(\mathbf{k}' - \mathbf{k}). \tag{4.9}$$

The **k**-vectors are reciprocal to the translation vectors **R** for the entire system. V is the volume of the entire system. Using the identities in Eq. (4.8) and Eq. (4.9) in Eq. (4.7), we will get rid of the **R**-vector sum and replace it with a sum over **k**-vectors. The $\mathbf{k} = 0$ term corresponds to infinite-range interactions. Due to the charge neutrality of the systems I am considering, these interactions consist of infinite terms that cancel each other out, enabling me to drop the $\mathbf{k} = 0$ term from the summation:

$$U = \frac{1}{V} \sum_{\mathbf{k} \neq 0} \sum_{i=1}^{N} \sum_{j=1}^{i} q_i q_j \frac{e^{i\mathbf{k}\cdot\mathbf{r}_{ij}}}{k^2}. \tag{4.10}$$

We now have a sum which is computable, but will converge very slowly. The charge distributions of point particles are Dirac delta functions, which linear combination of plane waves $e^{i\mathbf{k}\cdot\mathbf{r}_{ij}}$ do a terrible job at describing. We will need so many plane wave terms that carrying out the sum directly is not feasible.

Now for the the trick of the Ewald summation technique. Imagine that the charge distribution of the atoms were instead a Gaussian,

$$\rho_i(\mathbf{r}) = q_i \left(\frac{\alpha^2}{\pi}\right)^{3/2} e^{-\alpha^2 |\mathbf{r}-\mathbf{r}_i|^2}. \tag{4.11}$$

The parameter α adjusts the standard deviation, which has the value $\frac{1}{\sqrt{2}\alpha}$. This distribution is Fourier transformed, and the resulting potential is added to and subtracted from Eq. (4.8). By doing this, the total energy will be given by three summation terms:

$$U = \frac{2\pi}{V} \sum_{\mathbf{k} \neq 0} \frac{1}{k^2} |S(\mathbf{k})|^2 e^{-\frac{k^2}{4\alpha^2}} + \sum_{i=1}^{N} \sum_{j=1}^{i-1} q_i q_j \frac{\operatorname{erfc}(\alpha r_{ij})}{r_{ij}} - \frac{\alpha}{\sqrt{\pi}} \sum_{i=1}^{N} q_i^2. \tag{4.12}$$

The first one, which comes from summing up the Gaussian distribution energies, is long ranged. The second one, which is the correction term (dirac delta minus Gaussian), is very short ranged and can be summed with neglecting other copies of the system. The third term removes reciprocal space interactions between a particle and itself, and is constant.

The structure factor $S(\mathbf{k})$ is defined as

$$S(\mathbf{k}) = \sum_{j=1} q_j e^{i\mathbf{k}\cdot\mathbf{r}_j} \tag{4.13}$$

and the complimentary error function is $\operatorname{erfc}(x) = 1 - \operatorname{erf}(x)$, where

$$\operatorname{erf}(x) = \frac{2}{\pi} \int_0^x e^{-t^2} \mathrm{d}t. \tag{4.14}$$

The parameter α can be freely set and will decide which sum converges the fastest. If it is big, the second term converges with fewer terms, but the first one will require more terms, and vise versa for a small α [15].

The total electrostatic energy in Eq. (4.12) contains not only the distance between particles, but the actual position vector between them. The force arising from the first term must therefore be calculated by taking the gradient directly. This gives terms which can be written as sums of sines and cosines of $\mathbf{k} \cdot \mathbf{r}_{ij}$, but I choose to use complex numbers explicitly, both in derivations and the numerical implementation. The derivative of the error function is a Gaussian. We get the following expression for the force exerted on particle i:

$$\begin{aligned} \mathbf{F}_i = &- \frac{4\pi q_i q_j}{V} \sum_{\mathbf{k}\neq 0} \frac{1}{k^2} \mathrm{Im}\{e^{-i\mathbf{k}\cdot\mathbf{r}_i} S(\mathbf{k})\} e^{-\frac{k^2}{4\alpha^2}} \mathbf{k} \\ &+ q_i \sum_{i\neq j}^{N} q_j \left(\frac{2\alpha}{\sqrt{\pi}} \frac{e^{-\alpha^2 r_{ij}^2}}{r_{ij}^2} + \frac{\mathrm{erfc}(\alpha r_{ij})}{r_{ij}^2} \right) \mathbf{r}_{ij}, \end{aligned} \tag{4.15}$$

where $\mathrm{Im}\{z\}$ is the imaginary part of a complex number z. The particle index i is always used as an index and should not be confused with the imaginary unit i.

4.2.2 2D Ewald summation

In some cases, two-dimensional PBC are used even if the simulated system is in three dimensions. This actually make the Ewald summation more difficult and time-consuming. I have looked at two methods, Kawata and Mikamis (KM method) [16] and the PHL method, presented in the same paper.

For the reciprocal space term of $\mathbf{k} = 0$, which for the 2D case is non-zero, I use the PHL method. For the $\mathbf{k} \neq 0$ terms, I have implemented both methods, but use the KM method for the calculations in this project. This is because the terms are computationally demanding, and the KM method of computing them has a favourable scaling.

The direction without PBC will in my case be along the x axis, and I refer to the lengths of the simulation box in the other directions as L_y and L_z. The short-range sum will be exactly the same as in the 3D case, so I only present the long range reciprocal space terms here.

The potential energy of the charged systems is the sum of three contributions. The first one is the quasi-2D analog of the reciprocal 3D sum. The $\mathbf{k} \neq 0$ term corrects for that the real-space potential sum has too short a reach in the x direction, where a regular Coloumb potential should have been used. The last term is the same

self-interaction correction term as in the 3D case.

$$U^L_{\mathbf{k}\neq 0} = \frac{1}{L_y L_z}\sum_{\mathbf{k}\neq 0}\int_{-\infty}^{\infty}\frac{1}{k^2+h^2}e^{-\frac{1}{4\alpha^2}\left(k^2+h^2\right)}\left|S(\mathbf{k},h)\right|^2\mathrm{d}h, \tag{4.16}$$

$$U^L_{\mathbf{k}=0} = -\frac{1}{L_y L_z}\sum_{i=1}^{N}\sum_{j=1}^{N}q_i q_j\left[\frac{\sqrt{\pi}}{\alpha}e^{-\alpha^2 x_{ij}^2}+\pi x_{ij}\mathrm{erf}(\alpha x_{ij})\right], \tag{4.17}$$

$$U^L_{\mathrm{const}} = \frac{\alpha}{\sqrt{\pi}}\sum_{i=1}^{N}q_i^2. \tag{4.18}$$

Here, x_{ij} is the distance from particle j to particle i in the x direction. The integration variable h acts as a continous reciprocal lattice vector component in the x direction. The modified quasi-2D structure factor has the form

$$S(\mathbf{k},h) = \sum_{j=1}^{N}q_j e^{-i(\mathbf{k}\cdot\mathbf{r}_j+hz_j)}. \tag{4.19}$$

The force on particle i will have independent contributions in the y-z and x directions. For $\lambda = y, z$, with reciprocal vector components k_λ, the force components are

$$\left(\mathbf{F}^L_{\mathbf{k}\neq 0,i}\right)_\lambda = -\frac{2}{L_y L_z}\sum_{\mathbf{k}\neq 0}k_\lambda\int_{-\infty}^{\infty}\frac{1}{k^2+h^2}e^{-\frac{1}{4\alpha^2}\left(k^2+h^2\right)}\mathrm{Im}\{e^{-i(\mathbf{k}\cdot\mathbf{r}_i+hz_i)}S(\mathbf{k},h)\}\mathrm{d}h. \tag{4.20}$$

The force in the x direction get contributions from both $U^L_{\mathbf{k}\neq 0}$ and $U^L_{\mathbf{k}=0}$. The two contributions are

$$\left(\mathbf{F}^L_{\mathbf{k}\neq 0,i}\right)_x = -\frac{2}{L_y L_z}\sum_{\mathbf{k}\neq 0}\int_{-\infty}^{\infty}\frac{h}{k^2+h^2}e^{-\frac{1}{4\alpha^2}\left(k^2+h^2\right)}\mathrm{Im}\{e^{-i(\mathbf{k}\cdot\mathbf{r}_i+hz_i)}S(\mathbf{k},h)\}\mathrm{d}h, \tag{4.21}$$

$$\left(\mathbf{F}^L_{\mathbf{k}=0,i}\right)_x = \frac{2\pi q_i}{L_y L_z}\sum_{j=1}^{N}q_j\mathrm{erf}(\alpha z_{ij}). \tag{4.22}$$

An much more efficient way to model repeating two-dimensional patterns would be to insert lengths of vacuum in the x direction directly outside the system. The slabs would then lie in a layerwise fashion, interacting only slightly between each other. This introduces some complications in some of my systems variations, so I have not investigated this method, allthough it has been used in several cases of numerical surface research. It is definetly a matter worth pursuing if bigger ionic lattice simulations are needed.

4.2.3 Implementation details

The short ranged parts of the Coloumb interaction sums are carried out with the same neighbour lists as with the LJ potential. The cutoff length r_c from Eq. (6.5) is used.

The long ranged parts are summed over **k**-vectors of small length, prioritizing the contributions of longest range, which are the dominant ones. To keep the isotropic nature of the infinite crystal, a spherical cutoff is used, so that $k_x^2 + k_y^2 + k_z^2 \leq k_{\text{cutoff}}$ In the 2D case, k_x is simply put to zero. There will be thousands of vectors in the sum for the 3D case, and hundreds for the 2D case, to put it roughly.

In both the 3D energy and force sums, the symmetry operation $\mathbf{k} \rightarrow -\mathbf{k}$ does not change the inner expression. This is used to reduce the computational cost by a factor of 2. Three sums are being carried out, a triple sum with $k_x > 0$, a double sum with $k_x = 0$ and $k_y > 0$ and a single sum with $k_x = 0$, $k_y = 0$ and $k_z > 0$.

The integrals in the 2D $\mathbf{k} \neq 0$ sums are approximated by using Simpsons composite rule, although the more sophisticated Gaussian quadrature would work very well here.

In some cases, the simulation will contain both particles with and without PBC interactions. The particles which are not periodically repeated will interact amongst each other using a direct Coloumb term, $q_i q_j / r_{ij}$.

In the `cmath` header from the C++ standard library, the functions exp, sin, cos, erf and erfc are all defined and implemented in an optimized way, so these are what I use in the program.

4.3 The Stillinger-Weber potential

Atoms have in general interatomic potentials with angular dependence. This can be seen even when solving the time-independent Schrödinger equation for a single electron orbiting a nucleus. Higher energy states, starting wit the so-called $2p$ states, includes a spherical harmonic function, which is not spherically symmetric. Thus the electron distributions around atoms of elements heavier than beryllium have an angular dependence. Further, the electron densities are shifted when several atoms come close. This is especially important for atoms which do not fullfill the octet rule and create covalent bonds. In order to model the interaction properly, this must be taken into account, in the form of many-body potentials. Consider the sum

$$U = \sum_{b=1}^{N} U^{(b)} \tag{4.23}$$

where $U^{(b)}$ is the b-body potential. Each term is a sum of all b-body energy between all combinations of b atoms, dependent on their positions. In principle, with all the right b-body potentials, the correct atomic dynamics can be produced, even though we approximate the atoms as point particles. We use this sum perturbatively and put $U^{(b)} = 0$ for $b > 3$. That is, to model covalent bonds, we move one step ahead from the LJ potential and consider a three-body potential.

For this purpose, I use the Stillinger-Weber (SW) potential created especially for silicon [17]. This potential reproduces the cohesive energy and melting point of silicon matter. Its history is not completely clear, but it was likely produced in the same way as the LJ potential. I use a slightly different notation and organization than the standard one when presenting the mathematical form.

The total SW potential energy has three terms (system replica sums are dropped for convenience):

$$U = \sum_{i=1}^{N} U_i^{(1)} + \sum_{i=1}^{N} \sum_{j=1}^{i-1} U_{ij}^{(2)} + \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{j-1} U_{ijk}^{(3)}. \tag{4.24}$$

In the last term, the sum does not include the terms where $j = i$ or $k = i$. The double counting of index j occures because particle i occupies a special position, the middle of the atom triplet. There is a symmetry between the atoms not in the middle position, $U_{ijk}^{(3)} = U_{ikj}^{(3)}$, but no symmetries between i and j or k.

The first term is constant, and included to get the correct cohesive energy of silicon's crystal structure. There are $\frac{1}{2}N(N-1)$ terms in the double sum, as before, and $\frac{1}{2}N(N-1)(N-2)$ in the triple sum. These are drastically reduced by the potential's natural cutoff, which encourages the use of neighbour lists.

The two-body part of the potential have the form

$$U_{ij}^{(2)} = \begin{cases} A\left(\frac{B}{r_{ij}^4} - 1\right) \exp\left(\frac{\sigma}{r_{ij} - r_c}\right) & \text{for} \quad r_{ij} < r_c, \\ 0 & \text{for} \quad r_{ij} \geq r_c. \end{cases} \tag{4.25}$$

All derivatives of the potential are continuous at $r_{ij} = r_c$. The potential can act attractively at and repulsively, like the LJ potential.

The SW three-body potential has the form

$$U_{ijk}^{(3)} = \begin{cases} \lambda \exp\left(\frac{\gamma}{r_{ij} - r_c} + \frac{\gamma}{r_{ik} - r_c}\right) \left(C_{ijk} + \frac{1}{3}\right)^2 & \text{for} \quad r_{ij}, r_{ik} < r_c, \\ 0 & \text{else.} \end{cases} \tag{4.26}$$

There is a term similar to the exponential term in $U_{ij}^{(2)}$, but also a multiplicative term of angular dependence. This can force smaller or greater angles between triplets of atoms, and therefore act both attractively and repulsively in collections of several atoms. The three-body dot product term C_{ijk} is the cosine of the angle between the vectors $\mathbf{r}_{ij}$ and $\mathbf{r}_{ik}$:

$$C_{ijk} = \cos\theta_{jik} = \frac{\mathbf{r}_{ij} \cdot \mathbf{r}_{ik}}{r_{ij} r_{ik}}. \tag{4.27}$$

The $\frac{1}{3}$ in Eq. (4.26) is there because solid state silicon atoms are arranged in the diamond structure (section 2.3.2). In this equilibrium structure, each atom has four bonds (atoms within cutoff distance) and the angles between atom pair vectors are such that $\cos\theta_{\text{eq}} = -\frac{1}{3}$ ($\theta_{\text{eq}} \simeq 70.53°$). Other structures can be produced by setting θ_{eq} to a different value.

The constants in the SW potential are adjusted to fit experimental data [17]. They are summarised in Tbl. 4.2 with the units I use externally with my program.

r_c [Å]	A [eV]	B [Å^4]	σ [Å]	λ [eV]	γ [Å]
3.7712	12.840	11.603	2.0951	38.248	2.5141

Table 4.2: The six constants required in the SW potential (Eq. (4.25) and Eq. (4.26)).

4.3.1 Forces

The two-body force between atoms i and j can be derived straightforwardly from Eq. (4.25), and produces the force

$$\mathbf{F}_{ij}^{(2)} = \frac{A}{r_{ij}}\left[\frac{4B}{r_{ij}^5} + \left(\frac{B}{r_{ij}^4} - 1\right)\frac{\sigma}{(r_{ij}-r_c)^2}\right]\exp\left(\frac{\sigma}{r_{ij}-r_c}\right)\mathbf{r}_{ij}. \tag{4.28}$$

In calculating the three-body force, one can use that the forces arising from a potential term must be zero if you sum over all three particles: $(\nabla_i+\nabla_j+\nabla_k)U_{ijk}^{(3)} = 0$ [18]. Therefore, the most complicated term, $\nabla_i U_{ijk}^{(3)}$, does not have to be calculated. The forces arising from $U_{ijk}^{(3)}$ are

$$\begin{aligned}\mathbf{F}_i^{(3)} &= (\nabla_j + \nabla_k)U_{ijk}^{(3)},\\ \mathbf{F}_j^{(3)} &= -\nabla_j U_{ijk}^{(3)},\\ \mathbf{F}_k^{(3)} &= -\nabla_k U_{ijk}^{(3)}.\end{aligned} \tag{4.29}$$

Using a new shortcut, the force on particle k can be gained by switching the indices j and k in the force on particle j. In this way, only one of nine terms needs to be analytically differentiated. This property is also used to speed up the numerical calculation of the forces.

In differenting the potential, one needs the derivative of the three-body dot product term,

$$\nabla_j C_{ijk} = \frac{\mathbf{r}_{ij}\cdot\mathbf{r}_{ik}}{r_{ik}}\nabla_j\frac{1}{r_{ij}} + \frac{\nabla_j(\mathbf{r}_{ij}\cdot\mathbf{r}_{ik})}{r_{ij}r_{ik}}.$$

The first term is done in the same way as with two-body potential (Eq. (4.2)). The second term is problematic, but can be calculated using the general index notation formula

$$\nabla(\mathbf{A}\cdot\mathbf{B}) = \sum_{ij}\left(A_j\frac{\partial^2 B_j}{\partial x_i^2} + B_j\frac{\partial^2 A_j}{\partial x_i^2}\right)\mathbf{e}_i, \tag{4.30}$$

where $\mathbf{e}_i$ is the unit vector in the ith direction. Using this, I obtain $\nabla_j(\mathbf{r}_{ij}\cdot\mathbf{r}_{ik}) = \mathbf{r}_{ik}$, and the whole derivative becomes:

$$\nabla_j C_{ijk} = \frac{1}{r_{ij}}\left(\frac{C_{ijk}\mathbf{r}_{ij}}{r_{ij}} - \frac{\mathbf{r}_{ik}}{r_{ik}}\right) \tag{4.31}$$

With Eq. (4.31), the required potential derivative can be calculated, and we finally obtain

$$\begin{aligned}\nabla_j U_{ijk}^{(3)} = \lambda \exp\left(\frac{\gamma}{r_{ij}-r_c}+\frac{\gamma}{r_{ik}-r_c}\right)\left(C_{ijk}+\frac{1}{3}\right) \\ \cdot\left[\left(C_{ijk}+\frac{1}{3}\right)\frac{\gamma}{(r_{ij}-r_c)^2}\frac{\mathbf{r}_{ij}}{r_{ij}}+\frac{2}{r_{ij}}\left(\frac{C_{ijk}\mathbf{r}_{ij}}{r_{ij}}-\frac{\mathbf{r}_{ik}}{r_{ik}}\right)\right],\end{aligned} \tag{4.32}$$

which can be inserted into Eq. (4.29) to find all SW three-body forces.

4.3.2 Implementation details

For storing the interacting neighbours, the neighbour lists developed for LJ interactions are used. The interaction cutoff length is set to r_c (Tbl. 4.2), and the neighbour list cutoff length is slightly higher. Each particle now keeps track of the neighbours with both lower and higher indices than their own. The particle checking for neighbours is always the middle particle, the one with the first index in Eq. (4.26) and Eq. (4.29). The third index will always be smaller than the second one, as when calculating forces between pairs of particles. In that way, all bonds between simulated particles are accounted for exactly once.

Sometimes, an open ends boundary condition with fixed particles on the sides are necessary (see section 5.3). Calculation of three-body forces between simulated and fixed particles requires some extra thought. Four types of three body fixed particle terms (TBFPT) are possible, all illustrated in Fig. 4.1. By having fixed particles in the neighbour list, the first two terms are automatically included in the sums. TBFPT4 requires fixed particles to have their own neighbour lists, specifying which simulated particles that are close to them. TBFPT3 requires these lists to also include other fixed particles, but force terms are of course only calculated when at least one of the interacting particles are not fixed.

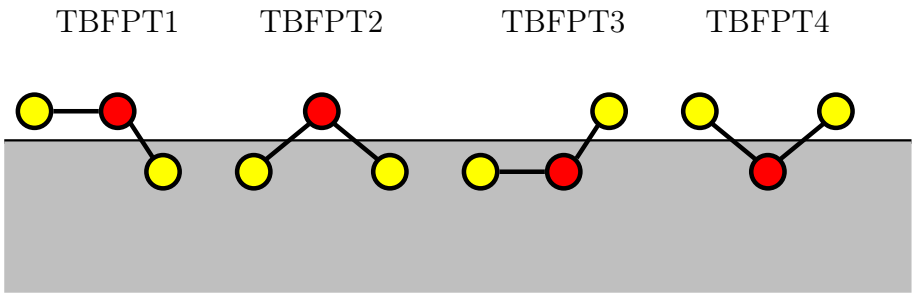


Figure 4.1: The four different three-body fixed particle terms in force and potential calculations. Particles in the gray area are fixed.

4.4 Comparision of the potentials

I have now presented all the interatomic potentials used in this project. Their mathematical forms are very different in nature. The LJ and Coloumb potentials are both spherically symmetric, in the sense that only the distance between the interacting pair of atoms vary the potential energy. The Columb potential, however, have different signs when equally and oppositely charged particles interact. The range of the Columb potential is very long, while the LJ potential has a stronger divergence when atoms are close. The SW potential has one part much like the LJ potential, but also a part which is not spherically symmetric around an atom.

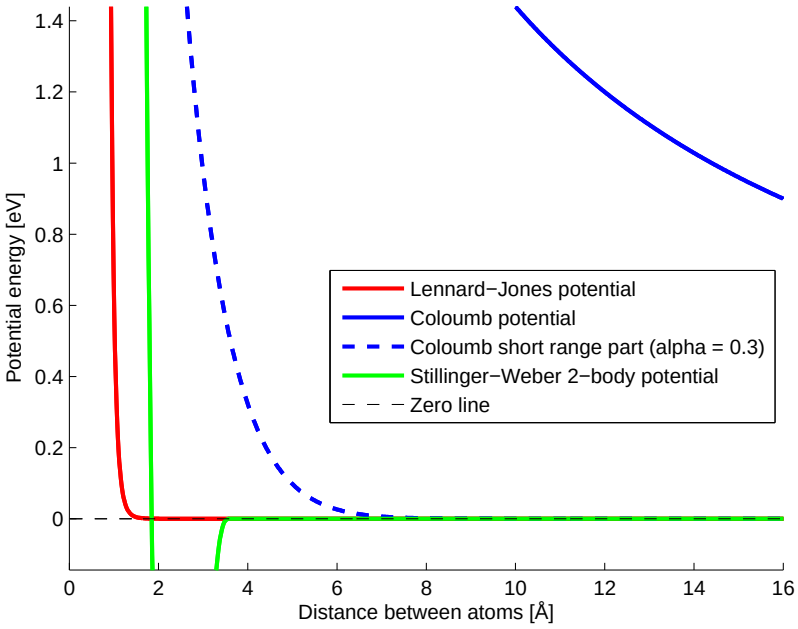


Figure 4.2: Comparision of interaction potentials as a function of the distance between two sodium/silicon atoms.

Figure 4.2 shows radial plots of the various two-body potentials used in this project. Sodium parameters are used in the LJ potential, and silicon parameters in the SW potential. As the potentials behave quite differently, plotting them in the same diagram is not very informative. The bottom of the SW potential well is far below what's seen (-1.819 eV) while the LJ well is too small to be seen at all ($-\epsilon = -5.6478 \cdot 10^{-6}$ eV).

Chapter 5

Numerical setup

The physical systems and dynamical equations have been presented, and now we must go into how the numerical experiments are performed. Some mathematical methods and constraints need to be chosen in any simulation, and the choices are not always easy. For example, one often have choices between more complex and reliable methods and methods which are easier to implement, or between methods with a stronger physical foundation and computationally efficient methods. Generally, I have restricted myself to simple and efficient methods which do not necessarily aim to reproduce the conditions in a real experiment. This project is more about investigating chemical bonds and nanoscale structures than predicting the mechanical strengt of a macroscopic piece of material.

The systems I consider are atoms initialized to an energetically preferrable crystal structure. The starting positions of the atoms are calculated by iterating over conventional unit cells in the x, y and z directions and the basis vectors (see Fig. 2.2 and Fig. 2.3). This creates a cuboid lattice, which can be chopped, stretched or cut to a sphere. Velocities are generated according to the Maxwell-Boltzmann speed distribution for a given initial temperature. Thus the translational degrees of freedom are closer to equilibrium that they would be for e.g. a uniform distribution. The total linear momentum of the system is set to zero before a simulation starts, to prevent drift when periodic boundary conditions are applied.

5.1 Physical units

The physical units used in input and output, external units, are chosen so that the order of the values will be close to, or mostly larger than, 1. These units must be converted to and from an additional set of units used for calculations internally in the program. The most prominent reason to use a different set of units when calculations are done is that equations will be simpler and require less multiplications with constants. Also, it is possible to extract multiple datasets from one simulation by rescaling the internal units. If mass, length and time are rescaled, one can predict how much faster particles with a smaller mass would move, for example.

Variables in internal units $\bar{A}$ are related to the corresponding variables in external units A in this way: $A = A_0\bar{A}$. Since position and velocity are the most frequently output variables, I use the same units externally and internally for length and time. I also decided to put the most important constant in force calculations, the Coloumb force constant, to 1. The charge of particles will be measured in elementary charges, $e = 1.602176487 \cdot 10^{-19}$C. The Coloumb force law will give rise to a constraint in the following way:

$$F = k_e \frac{q_1 q_2}{r^2} \quad \Rightarrow \quad F_0 \bar{F} = k_e \frac{e^2 \bar{q}_1 \bar{q}_2}{r_0^2 \bar{r}^2}, \tag{5.1}$$

$$\bar{F} = \frac{\bar{q}_1 \bar{q}_2}{\bar{r}^2} \quad \text{and} \quad \frac{k_e e^2}{r_0^2 F_0} = \frac{k_e e^2 t_0^2}{m_0 r_0^3} = 1. \tag{5.2}$$

This determines the mass conversion factor, m_0. Other conversion factors are combinations of r_0, t_0, e and m_0. Table 5.1 gives a detailed list of used units.

	External	Internal
Length	Å	$r_0 = 1$Å
Time	fs	$t_0 = 1$fs
Charge	e	$q_0 = e$
Mass	amu	$m_0 = k_e t_0^2 e^2 / r_0^3 = 0.138935459$ amu
Force	eV/Å	$F_0 = m_0 r_0 / t_0^2 = 14.399645$ eV/Å
Energy	eV	$E_0 = m_0 r_0^2 / t_0^2 = 14.399645$ eV
Temperature	K	$T_0 = m_0 r_0^2 / k_B t_0^2 = 1.6710075 \cdot 10^5$K

Table 5.1: Physical units for I/O and calculations in the program.

5.2 Applied deformations

5.2.1 Tensile stress

To see how much stretching a material can take before it fractures, I apply a steadily increasing strain to the system. The typical speed at which the system is elongated is 5-10% of the speed of sound in the material.

The x direction boundary condition is open ends with fixed particles. The distances between the fixed particles are not stretched, as they exist only to keep the simulated particles at the edges in place. The distances between simulated particles are uniformly stretched to correspond to the stretching of the simulation box and moving of the fixed particles. If ξ denotes the distance from the middle to a particle, the displacement rate caused by tensile strain will be $\dot{\xi} = \frac{\xi v}{L}$, where L is the length of the simulation box, and $v/2$ is the expansion rate of the box in one direction. Figure 5.1 shows the expansion schematically.

The displacements by the tensile strain function will not give any forced contribution to the velocity of particles. This would disturb the dynamics and give higher temperature measurements.

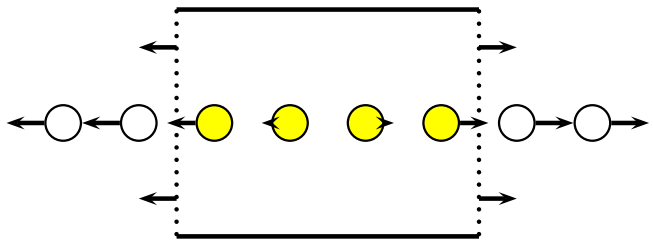


Figure 5.1: Displacements caused by tensile strain.

Let us consider the case of an extension in the x direction. I call the equilibrium length of the simulation box L_x and the deviation from this length ΔL_x. The strain is the easiest to calculate:

$$\epsilon = \frac{\Delta L_x}{L_x}. \tag{5.3}$$

Before calculating the stress, the energy balance between to time steps t and $t+\Delta t$ must be examined. I assume that the only processes changing the system energy is heat conduction and applied stress. For each time step, the energy transfered to the system by these processes are denoted ΔU_T and ΔU_σ, respectively. With U denoting the sum of potential and kinetic energy in the system, the energy balance equation reads

$$U(t + \Delta t) - U(t) = \Delta U_T + \Delta U_\sigma. \tag{5.4}$$

Thus, by storing the energy of the previous time step and calculating the energy absorbed from the heat bath, the difference in energy caused by stress can be found. The exerted force from the deformation is calculated with the discretized derviative $F_\sigma \approx \frac{\Delta U_\sigma}{\Delta L_x}$, where the minus sign is discarded because the force from the simulated particles act against the elongation. The one-dimensional stress is nothing but force per area, so the average stress (over the system volume) in the x direction is

$$\sigma = \frac{\Delta U_\sigma}{L_y L_z \Delta L_x}. \tag{5.5}$$

I have also tested a method of calculating the stress on a specific y-z-cross-section in the middle of the volume. For atom pairs interacting across this cross-section, the force working on the atoms on the right side is added up to a total cross-sectional force F_{cs}. For particle triplets, the three-body force (see section 4.3) working on a middle atom is added up if it is on the right side of the cross-section and either one of the other two atoms is on the left side. The cross-sectional stress is

$$\sigma_{\text{cs}} = \frac{F_{\text{cs}}}{L_y L_z}. \tag{5.6}$$

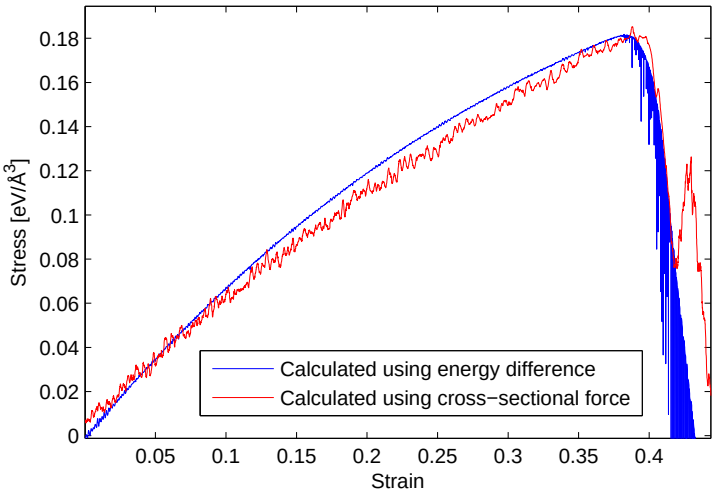


Figure 5.2: Comparision between the two methods of stress calculation, showing a good correspondence.

The stresses σ and σ_{cs} are compared for a 4096-Si-atom system undergoing a fracture in Fig. 5.2. The energy-calculated stress σ is more averaged (also over the x direction), and has much smaller fluctuations from the general trend than the cross-sectional stress σ_{cs}. Both methods give the same maximal stress, though the slope is slightly different over the whole curve. The results after the fracture are irrelevant.

In the rest of the thesis, I will only mention the x-direction-averaged stress σ. The choice of a cross-section is arbitrary, and the cross-sectional breaking stress will give too low results when the material has defects. These can cause the area around the cross-section to have an atomic deficiency in comparision with ordinary bulk matter. The breaking stress is estimated as the highest achieved stress, which for the solid starts to show signs of failiure. This can be pinpointed by examining when a few particles start to move rapidly in the area of the fracture. The recent motion quantity (section 6.3) is ideal for this.

5.2.2 Shear stress

A shear deformation exerts inhomogenous stress on different parts of a material. With fixed particles outside the x direction edges, I move these particles in opposite directions in the y direction, as shown in Fig. 5.3. This alone will move the simulated particles correspondingly at each side. The interesting thing is what happens in the middle of the simulation box. Again, the speed of the fixed particles are constant.

The fixed particles themselves experience PBC in the y direction, so that they appear at the opposite y side if they are displaced out of the box.

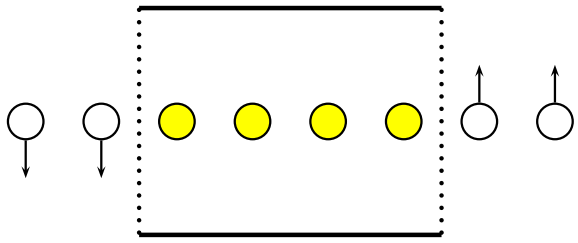


Figure 5.3: Displacements caused by shear strain.

5.2.3 Contact, adhesion and friction

In numerical experiments regarding adhesion and friction, I look at the interaction between a sphere or sphere segment of particles and a plane surface of the same types of particles. The attraction between such surfaces with the same atomic species is called cohesion, but I will stick to the more general word adhesion. The sphere form is meant to model an asperity on a rough material surface. It corresponds to the stepped crystal surface in a similar model by Luan and Robbins [3]. The surface consists of some atomic layers of fixed particles, as mentioned in section 5.3.3. Additional layers of simulated particles can be added for the surface not to be completely rigid. The surface particles have PBC in the directions parallell to the surface, while the sphere particles have no PBC.

The sphere is pushed down towards the surface by a constant force on the upper half of the particles. This force corresponds to a homogenous gravitational field, although it is several thousands times stronger. An alternative would be a force on all particles in the sphere, but this is unphysical and could cause the lower half of the sphere to disintegrate in a unrealistic fashion. Another alternative, which I also use, is pushing only a lower sphere segment downwards using e.g. another surface of the same material.

The sphere method is used to simulate the reaction from indenting the sphere into the fixed particle layers with forces of different magnitude. Potential energy from the constant force in the negative x-direction is summed up and added to the total and single particle energies. However, the number of particles this force affects is not constant. Only particles positioned higher than the mass center of the sphere are pushed. This is checked for each time step. If the sphere begins to roll, as it may do in friction experiments, the upper half of the sphere is still the only part that is forced downwards, giving a greater stability. The total force, which I for simplicity call $\mathbf{F}$, is distributed over the atoms above the sphere's center of mass in the following way:

$$\mathbf{F}_i = \frac{m_i}{\sum_j m_j}\mathbf{F}, \tag{5.7}$$

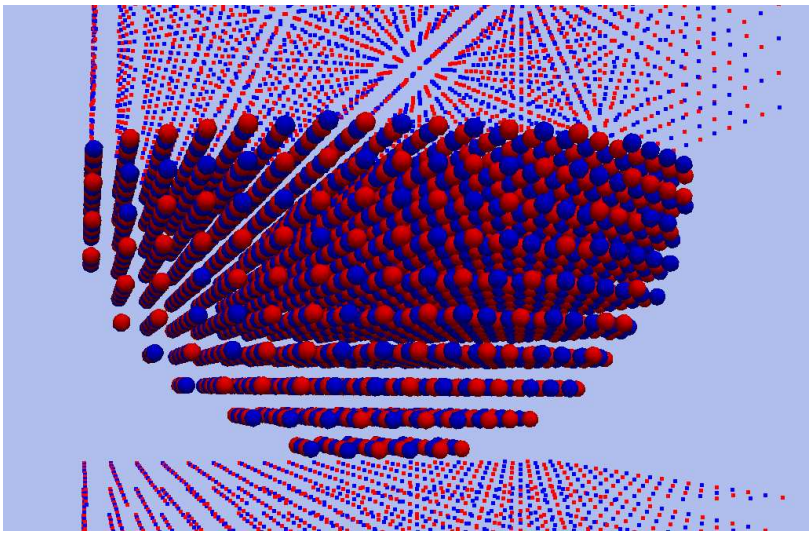


Figure 5.4: A microscopic sphere segment (here a half-sphere) and surface of NaCl. The initial positions in a simulation.

where m_i is the mass of an atom i and j runs over all the inflicted particles. This will cause the force to be distributed non-uniformly across the cross-section of the sphere, because there are more particles above the middle than at the edges. The effect of this unphysical force distribution on the lowermost part of the sphere is small, since inner tension in the sphere will even the forces out.

The sphere segment method will simulate a smaller number of dynamical particles and demand less computing time for the same sphere curvature. The input force distribution will also be uniform. In the simulations, The upper fixed particle layers will be moved downwards very slowly with a constant speed and then stop for measurements to be performed. The applied force is difficult to control when this approach is used. Though, the force can be measured, and the contact stress graphs give results very similar to those obtained with the sphere method. A possible method which I have not tried in practice is basing the movement of the upper fixed particles on the force between the simulated particles and the lower fixed particles. In this way, the entire surface could be moved until the forces reached a specific value.

The indentation speed of the upper layer can be compared to the root mean square thermal speed of the atoms. Using the equipartition principle, we have

$$\bar{U}_k = \frac{1}{2}m\bar{v^2} = \frac{3}{2}k_{\mathrm{B}}T \quad \Rightarrow \quad v_{\mathrm{RMS}} \equiv \sqrt{\bar{v^2}} = \sqrt{\frac{3k_{\mathrm{B}}T}{m}} \tag{5.8}$$

for one atom. The atoms I am simulating have thermal speeds of $1.63 \cdot 10^{-3}$ Å/fs for silicon, $1.80 \cdot 10^{-3}$ Å/fs for sodium and $1.45 \cdot 10^{-3}$ Å/fs for chlorine. The indentation speed is normally put to $0.20 \cdot 10^{-3}$ Å/fs, so the indentation process is quasistatic. The pushing of the upper layer will not create unwanted elastic waves or other dynamics which is not allready happening in equilibrium due to thermal motion.

The contact area between the sphere and the surface A_C is measured as explained in section 2.2.1. The stress between the sphere and the surface σ is measured in a similar way, but using microscopic forces instead of potential energy. The surface is divided into rectangular bins, about 20 in each direction. For each time step, the sphere particles are distributed among these bins according to their positions, and the forces from the surface particles are calculated. For each bin b, all forces on sphere particles are summed up and divided with the area of the bin to find the stress,

$$\sigma_b = \frac{N_b}{L_y L_z} \sum_{i \in b} \sum_{j \in \text{surface}} F_{ij}, \tag{5.9}$$

where N_b is the number of bins and $L_y L_z$ is the area of the surface (inside the simulation box). This gives an approximation to the stress normal to the surface. I have also attempted to simplify this to stress in donut-shaped bins for specific radii $\sigma(r)$, with less stable results. Long range Ewald forces between simulated and lower fixed particles are neglected when the contact stress is calculated. Early measurements show that these are many orders of magnitude (10^{-16}) smaller than the other force contributions.

Adhesive forces will be studied using the sphere segment method. The materials are brought in close contact and then separated. Some particles will stick to the lower surface of fixed particles, and some will form strings between the two surfaces, eventually broken by the gradual separation. A quantity of interest in these experiments is the energy needed to push the sphere segment up or down between time steps. Summing these energy increments for a whole phase of movement can determine the energy needed to press the materials together and push them apart. The energies are found exactly as in the bulk stress experiments (see section 2.1):

$$\Delta U_{\text{push}} = U(t + \Delta t) - U(t) - \Delta U_T. \tag{5.10}$$

The energy gained by pushing the upper fixed layer of particles will be used to pull particles out of potential fields, and in greater amount, to induce movement which is transfered out of the system as heat.

Once the sphere is pushed towards the surface and the system is near equilibrium, the simulation of friction dynamics can take place. Friction force is most often measured in the case that two surfaces have a constant relative velocity. This condition is applied by adding a small velocity to all particles so that the collective velocity of the sphere $\mathbf{v}_{\text{coll}}$ is constant. The only large scale forces that are at work tangential to the surface are the driving force that keeps the constant velocity condition fulfilled and the friction force. By Newton's first law, these forces are equal. For each time step, the friction force is therefore calculated as

$$F_{\text{friction}} = \frac{\Delta U_{\text{drive}}}{\Delta r_{\text{COM}}} = \frac{\Delta U_{\text{drive}}}{v_{\text{coll}} \Delta t}. \tag{5.11}$$

This is a discretized derivative with no minus sign, so the force is positive when it works in the same direction as $\mathbf{v}_{\text{coll}}$. U_{drive} is simply the change in kinetic energy when driving the particles to keep a constant velocity.

5.3 Boundary conditions

In my molecular dynamics simulations, all the particles are either free or in some way confined to a cuboid, a rectangular prism with right angles. This cuboid, which I call the simulation box, has side lengths L_x, L_y and L_z in the three cartesian directions.

5.3.1 Periodic boundary conditions (PBC)

In materials science MD simulations, the most used kind of boundary condition is the periodic one. Call the length of a particular PBC direction of the simulation box L. If a particle escapes the box on the left or right side, its position will be shifted in that direction with $+L$ or $-L$, respectively. Particles will also interact with an infinite number of copies of themselves and other particles in the system. If all three directions have PBC, the vectors to these system copies are $\mathbf{R} = nL_x\mathbf{e}_x + lL_y\mathbf{e}_y + mL_z\mathbf{e}_z$, for integers n, l and m. There will not be any edges, only an infinite amount of bulk particles.

The reason for using PBC is the interest of what is happening inside materials. A great amount of research connected to the properties of surfaces are taking place, but PBC is still used in this case in the directions parallell to the surface, to have only one e.g. material-air-interface. A box with PBC in all directions will enable the possibility of bulk particle simulation. The simulated atoms will behave like atoms in the middle of a material of macroscopic size.

In the program, all particles have flags which tell in which directions they experience PBC. This is useful in adhesion and friction experiments, where a group of particles with no PBC is in contact with a surface. The surface particles, either if fixed or simulated (semi-fixed), will always have PBC in the directions tangential to the surface. When a pair of particles interact, the interaction itself will have PBC in a certain direction if one of the particles experience PBC in that direction. As an example, sliding particles will not fall of the edge of a surface, but continue to move as if the surface continued to infinity in its PBC-directions.

5.3.2 Fixed particles

The simplest kind of boundary condition is open ends, that is, nothing is done to prevent particles from escaping. In fracture experiments, I use this boundary condition in one direction in tandem with fixed particle forces. A perfect lattice of fixed particles are placed at the sides in the direction without PBC. This will not strictly contain the particles in the box, but prevent them from dispersing towards infinity.

The thickness of the fixed particle lattice layer will always be greater than the cutoff length for forces and potentials. If the simulated particles are in a crystalline state, the difference between the dynamics with PBC and open ends with fixed particles in one direction is almost negligble.

An application of this boundary condition is to study a crack propagating in the y-z-plane in an otherwise perfect crystal. Two-dimensional y-z-PBC with open

ends and fixed particles in the x direction will limit the number of crack copies in the x direction to one. Figure 5.5 shows the two-dimensional analog of the setup schematically.

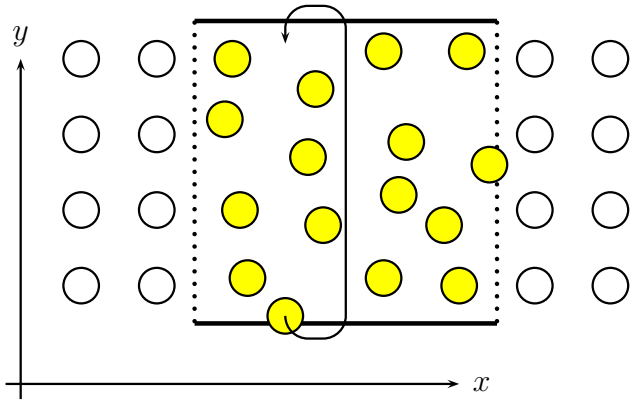


Figure 5.5: The boundary conditions for a system with fixed particles on the x-sides. The thermal motion is greatly exaggerated.

5.3.3 Friction boundary conditions

In adhesion and friction experiments, I am interested in a solid with a surface at $x = 0$ and an infinity of unit cells in the $-x$ direction. The electrostatic interactions are in this case hard to compute with the Ewald summation method. There has been attempts, e.g. [19]. However, it is stated in the results section of this paper that there is only a 1% difference in the interaction energy between a single ion and my hypothetical half-infinite NaCl solid and between a single ion and a two-dimensional monolayer of NaCl. The reason for this is the obvious cancellation of forces, making the effective range of interaction for a neutral layer of NaCl approximately one lattice constant a. For this reason, I model the surface which the sphere of atoms interact with as a slab of thickness equal to the cutoff range of the potentials summed in real space, for both the Si and NaCl systems. In practice, this means about 4 atomic layers.

With simulation tests, the lattice energy of such a system was measured to be -4.014 eV per particle, a difference of 0.55% from the bulk system with PBC in all directions (see section 7). For this test, layers of fixed particles were put on both x-sides, and PBC was applied in the y-z-plane. When a sphere of particles is used, the positive x direction layers of fixed particles are removed and replaced by vacuum.

5.4 Heat conduction

When simulating the dynamics of systems far from equilibrium, one can not assume that the heat bath has an equal thermal coupling to all the simulated particles. In most cases, I am interested in thermal coupling to the perfect crystals on the sides of the simulation box in one direction, as explained in the previous section. This can be accomplished by performing a temperature rescaling only on the atoms near the box sides. For this purpose, I use the BDP thermostat introduced in section 3.2.1.

To avoid an unreasonably sudden cutoff, the relaxation time τ_i of the thermostat is choosen differently for each particle. An atom directly on the side wall will have a relaxation time τ_0 read in as a parameter. Diverging at the cutoff point, the relaxation time decays from this point as $1/x$. Atoms which are further inside the box than the cutoff point have no thermal coupling to the outside. A $1/x$ form is chosen because the relaxation time enters the energy difference equation (Eq. (3.13)) multiplicatively as $1/\tau$. So the energy coupling to the outside heat bath decreases linearly inwards from the box sides.

Since the relaxation time varies with the distance from the sides, how temperature is measured before a rescaling must be changed. Fouriers (macrocopic) law of heat conduction says that the rate at which heat enters a system is proportional to the difference between the temperatures of the system and the surroundings. Equation (3.13) takes this directly into account. As the measured temperature affects the thermostat, the temperature must be measured only near the sides of the box. If not, the edges would be cooled down too much when the temperature of the middle of the box rises.

I choose to weight the energies of the atoms by a linearly decreasing factor w_i in the temperature measurement. These weights are normalized so that the measured side temperature is

$$T = \frac{1}{3k_\mathrm{B}} \sum_{i=1}^{N_\mathrm{incl}} w_i m_i (v_i - v_\mathrm{coll})^2 \quad \text{with} \quad \sum_{i=1}^{N_\mathrm{incl}} w_i = 1, \tag{5.12}$$

where N_incl is the number of atoms included in the rescaling. v_coll is the collective (average) velocity of the particles on one of the sides. Removing this component is necessary when all the atoms are moving in one direction on one of the sides, e.g. when shear stress is applied. Figure 5.6 shows the values taken by τ_i and w_i at different places inside the simulation box.

This method has no solid physical foundation, but serves as a tunable way to consistently conduct heat in and out of the system. The amount of heat conducted can be seen by plotting the power absorbed from the heat bath, as in the case of uniform atom to heat bath coupling.

This chapter concludes the physical and mathematical foundations for my simulation program. With it and precious chapters, I have mentioned everything the program contains and explicitly in what way all quantities are calculated. The next chapter deals with how the program is written and structured to perform all the calculations.

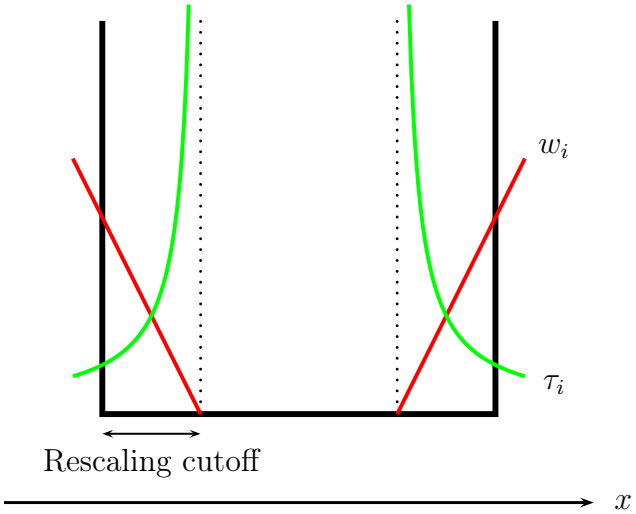


Figure 5.6: The variation of the temperature measurement weight w_i and thermostat relaxation time τ_i for a particle i.

Chapter 6

Program details

My MD simulation program is written in C++, contains 10 classes and a total of about 5500 source code lines. (CHECK THIS!) I will not include any portion of the code itself, as it is mostly composed of traditional MD techniques implemented in standard ways. Instead, I attempt to describe the main concepts and skeleton of the program.

6.1 Random numbers

Randomly distributed numbers are required for generating of velocity values from the canonical distribution and usage of stochastically driven thermostats. For numerical simulations, deterministic pseudo-random number generators are sufficient. My program uses a complex long period generator developed by L'Ecuyer and Bays-Durham. The implementation from *Numerical Recipes* [20], which I have copied, contains some added safeguards. The generator gives me uniformly distributed values in the open interval $(0, 1)$.

Normally distributed random numbers are obtained by performing a Box-Muller transform on the uniformly distributed numbers. Let u and v be uniform numbers in the interval $(-1, 1)$. These numbers will only be accepted for the transformation if $s = u^2 + v^2$ is in the interval $(0, 1)$. In that case, we obtain two normally distributed numbers n_1 and n_2 by multiplying u and v with a constant,

$$n_1 = Su, \qquad n_2 = Sv, \qquad S = \sqrt{\frac{-\ln s}{s}}. \tag{6.1}$$

n_1 and n_2 will have standard deviations of 1, but multiplying all generated numbers with a constant will give a distribution with that constant as the standard deviation.

6.2 Parallellization

As my program requires a considerable amount of CPU time for the systems I am simulating, parallell computations with several processors are required. I therefore

use MPI to pass data between the processor cores (called nodes) involved in the calculations.

For the most CPU-intensive tasks, calculations are split into work lists for each node. This happens to quantities that are sums of many terms. Each node performs its own sum and returns the result to a master node. The master node, which also does calculations, sums up the partial sums and broadcasts total results to all nodes.

The most important section of the program that is parallellized is the computation of forces. Especially with the very intensive Ewald summation, this has a high degree of parallel efficiency. The forces that must be computed makes an upper triagonal matrix with the index of particles in an interaction i and j. The rows of this matrix are put into work lists so that each node gets an equal amount of force terms F_{ij} to calculate. The exception is the last node, which gets "what's left". Figure 6.1 shows this splitting of work.

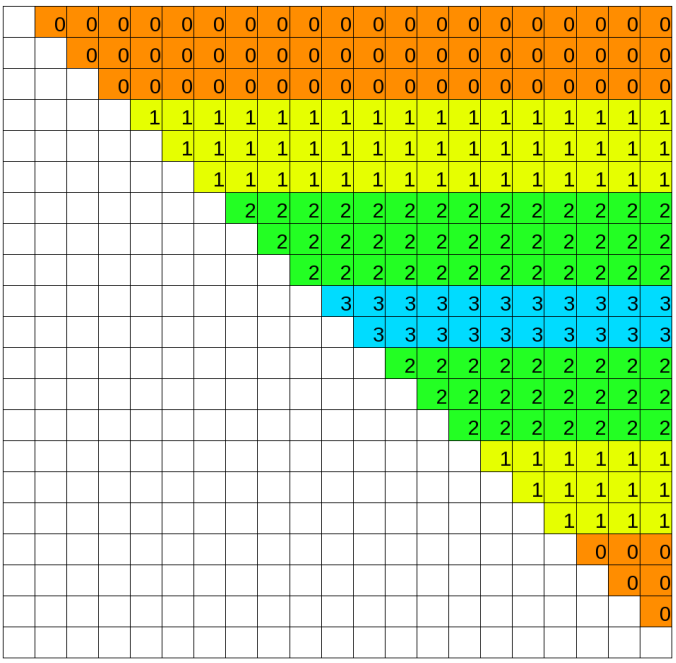


Figure 6.1: Splitting the calculation of the forces between 20 particles amongst four nodes.

The same procedure is used in calculating the total potential energy of the system. The BDP thermostat requires the sum of many squared random numbers. This is also done distributively, with different seeds for the random generators. The random numbers for the initial velocities are calculated in serial, and drawn from random generators with the same seed, producing the same numbers for all nodes.

Everything that is not parallellized, like performing the Verlet time integration of positions and velocities, is done simultaneously at all nodes. This reduces the amount of communication between different memory registers. On heterogenous clusters,

this could cause deviations between the nodes induced by unequal round-off errors. Output is done only from the master node.

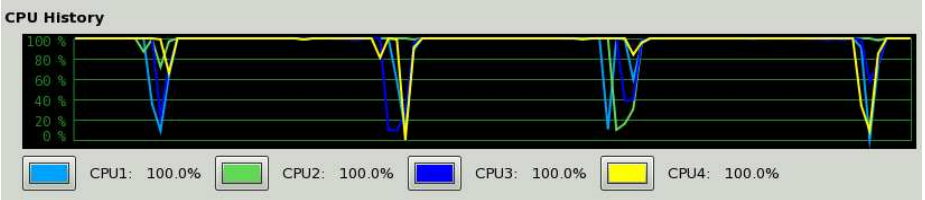


Figure 6.2: CPU usage in a big silicon system.

For small systems, parallellization will require too much time for communication between nodes in comparision to the actual computations. The force calculations must therefore be quite extensive for parallellization to give an increased efficiency. Figure 6.2 shows the CPU usage on a quad-core computer running a 15581-Si-atom simulation. The calculations involved in one time step takes about 25 seconds. 20 seconds of the time, all the cores are busy with force calculations, and the last 5 seconds are apparently spent at communication, output and waiting for other cores. This suggests a rough estimate of the efficiency. A single core could have done the calculations for one time step in 80 seconds, giving a parallellization efficiency of $80/(4 \cdot 25) = 80\%$ for 4 nodes. Simulations with the more demanding NaCl systems on a various number of cores give similar numbers when the time usage is compared.

The neighbour list update process is another program segment which could gain a speed-up from parallellization. This however involve big difficulties and require much communication between nodes. Therefore, it continues to be a $O(N^2)$ bottleneck for very big systems. This could be fixed by creating a hierarchy of different sized neighbour lists or adding a domain decomposition algorithm, which is well suited for parallellism.

The reason why I'm not using parallellization in Si experiments is that I make these physically big (like the one mentioned), which require a significant amount of memory. This memory amount is still small enough so that the most frequently accessed data can be stored in the CPU's cache, which enables the program to work about ten times faster than it would otherwise. The cores of the CPU have their individual memory, so a four-core parallell run would require four times as much memory, and the most frequently used variables and arrays will not fit in the cache, slowing the program down. This problem does not not occur for the smaller NaCl systems.

6.3 Visualization

For visualization of atom trajectories, I use the program ParaView. It is a general purpose visualization program built on the C++ code VTK (Visualization toolkit). My output class writes to files of the legacy `.vtk` format.

Plotting of atom positions is performed using `polydata` with simple vertices. In this report, I usually show the positions with a parallell prosjection. Scalar values for each atom includes electric charge, potential energy and recent motion. They also have velocity and force vectors associated with them, which can be easily visualized in ParaView. For the images included in this report, I use both orthographic (parallell) and perspective projections.

Recent motion is an ad hoc quantity for seeing where things happen to which particles. For one particle, I define $E_{\text{over}} = E_k - n_\sigma E_{\text{thermal}}$, where $E_{\text{thermal}} = 3k_{\text{B}}T_{\text{bath}}$. If E_{over} is negative, I put it to zero. This way, it gives a measure of how much the kinetic energy of a particle exceeds a certain number of standard deviations in the Maxwell-Boltzmann distribution for the heat bath temperature. Good values are $n_\sigma = 2$-3. Recent motion is then defined as

$$\texttt{recmot} = \sum_{i=0}^{\infty} a^i E_{\text{over}}(t - i\Delta t), \tag{6.2}$$

where a is a decay parameter, set to e.g. 0.98. The quantity will be mostly zero when the particles are in equilibrium, and significant if particles have moved recently, as in dynamical processes like fracture. When visualizing, this gives a better picture of what is happening than the magnitude of the instantaneous velocity.

Field values are stored as `structured points`. The types of fields I have been working with is potential energy, density and temperature. The volume of the system is cut into cells where these quantities are approximately evaluated. Fields can be viewed as three-dimensional scalar functions or two-dimensional slices. These provide excellent supplementary information about local dynamics, but are excluded from my simulation plots. Two-dimensional still images of the fields are less useful than coloured particle positions for the small systems I am working with.

In some cases, atom positions can also be output as files of the simpler `.xyz` format for visualization in VMD (Visual Molecular Dynamics). Macroscopic quantities for the whole system are treated in a more basic way. Energies, temperature, etc. are output to a file for each time step for later plotting with Matlab.

6.4 Structure overview

The `main`() function reads parameters from a file, acts as a stopwatch, manages and backs up the files in the output folder and outputs most of the cosole messages. It contains the equilibration and dynamics time loops, and administers what is happening and what is being measured through the public methods of a `dynamics` object. Some error checks are performed, and great increases in kinetic energy (implying a critical event) are reported. A high-level view of the algorithmic steps are presented in Fig. 6.3. The methods of all classes having to do with physical entities are listed in the next subsection, and the public ones are designated letters to show where in the flowchart they are excecuted.

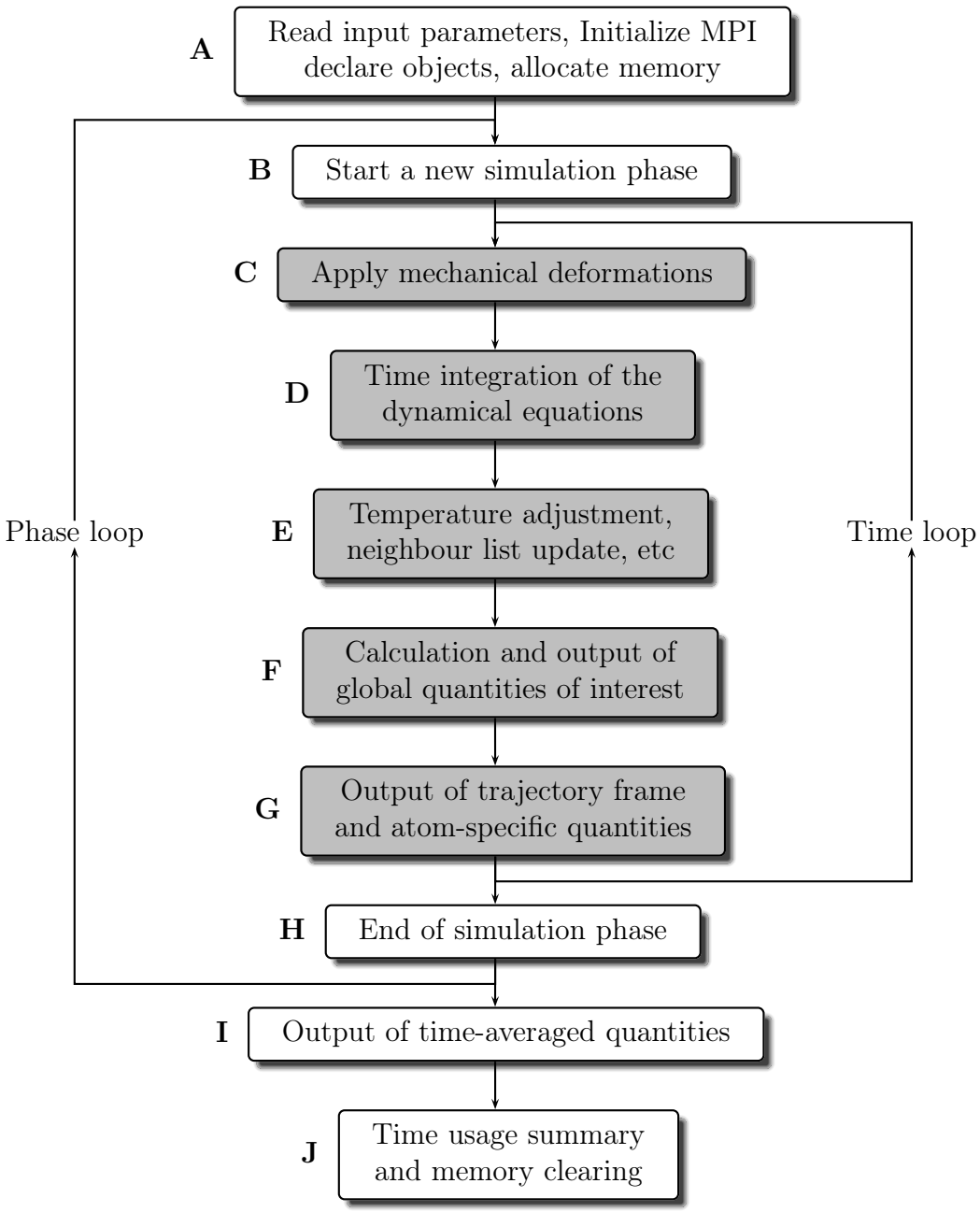


Figure 6.3: Flowchart for excecution of the MD program.

Molecular dynamics classes

- `dynamics`: Master class which contains most of the essential calculations. Contains pointers to objects of most classes below.
  - `set_parameters`() (**A**): Transfers all parameters needed at simulation start from the input file to this class and determines Ewald summation parameters (see section 6.5).
  - `initialize`() (**A**): Declares all required objects, arrays and output files. Also sets the initial positions and velocities of the particles.
  - `velocity_verlet_1`() (**D**): Calculates half-step velocities and new positions (see section 3.1.1).
  - `velocity_verlet_2`() (**A,D**): Calculates new velocities. Run after the first force evaluation, causing the Maxwell-Boltzmann-velocities to be treated as half-step velocities.
  - `position_pbc`() (**A,D**): Applies periodic boundary conditions to dynamic and fixed particle positions. Called at each time step.
  - `define_equilibrium`() (**B**): Defines equilibrium values for some quantities. In minimal use.
  - `calc_properties`() (**F**): Calculates some macroscopic quantities, e.g. stress and strain, and the recent motion of particles.
  - `forces_potentials`() (**A**): Calculates forces and potential energy between all pairs and triplets of atoms, also between fixed and non-fixed particles.
  - `update_neighbours`() (**A,E**): Updates the neighbour lists of all particles. Called e.g. every 10 time steps.
  - `rescale_temperature`() (**E**): Simple temperature rescaling. Usually not in use.
  - `andersen_thermostat`() (**E**): Adjusts the temperature using the Andersen thermostat. Used for equilibration in specific cases.
  - `bdp_thermostat`() (**E**): Adjusts the temperature using the BDP thermostat (see section 3.2.1).
  - `bdp_thermostat_sides`() (**E**): Adjusts the temperature of the simulation box sides (see section 5.4).
  - `dissipative_force`() (**D**): Adds a friction force term to all particles for damping oscillations.
  - `expand_volume`() (**C**): Expands the simulation box in the x direction with a constant length. Used every time step in tensile stress simulations.
  - `displace_lattice`() (**C**): Displaces the fixed particles in the perfect crystals on the sides of the simulation box. Used every time step in shear stress simulations.

  - `insert_gap`() (**A**): Inserts a gap between crystal layers in the x direction.
  - `add_force`() (**D**): Adds a constant force on specific particles and adds up the corresponding potential energy.
  - `adjust_vcoll`() (**D**): Forces the collective velocity to a certain vector by adding a correction term to certain particles.
  - `output_pos`() (**G**): Outputs the positions of all atoms, including the vector and scalar values mentioned in section 6.3.
  - `output_field`() (**G**): Outputs the temperature and density fields.
  - `output_xyz`() (**G**): Outputs the positions of the simulated particles to a simple `.xyz` file for visualization with VMD.
  - (pvt.) `kin_energy`(): Calculates the kinetic energy of the particles.
  - (pvt.) `temperature`(): Calculates the average temperature of atoms belonging to a specific type, subtracting the collective velocity.
  - (pvt.) `fpot_short`(): Adds up short range forces and potentials, including LJ and short range Ewald terms.
  - (pvt.) `fpot_coloumb`(): Adds up LJ and direct Coloumb forces and potentials, for systems with no PBC.
  - (pvt.) `fpot_long`(): Adds up 3D long range Ewald forces and potentials.
  - (pvt.) `fpot_long_2d_1`(): Adds up $\mathbf{k} \neq 0$ 2D long range Ewald forces and potentials.
  - (pvt.) `fpot_long_2d_2`(): Adds up $\mathbf{k} = 0$ 2D long range Ewald forces and potentials.
  - (pvt.) `potential_constant`(): Calculates the constant correction term in the Ewald summation. Only called in the `initialize`() function.
  - (pvt.) `fpot_sw_2b`(): Adds up SW potential two-body forces and potentials.
  - (pvt.) `fpot_sw_3b`(): Adds up SW potential three-body forces and potentials.

- `phase`: The `main`() function reads phase-specific parameters like the number of time steps, thermostat type and temperature, output level and parameters concerning dynamic processes. These parameters are stored in objects of this class. Also contains timer objects for timing the execution of each phase.

- `atom_type`: Contains information about chemical elements, like mass, charge and LJ parameters.

- `atom`: Contains the position, velocity, and force on an individual atom. The methods are used very often, so the flowchart positions are not shown.

  - `dist2`(): Calculates the squared distance between two atoms, taking PBC into account.
  - `dist_vector`(): Calculates the distance vector between two atoms, especially useful in three-body force calculations.
  - `dotted_dists`(): Calculates the scalar product $\mathbf{r}_{ij} \cdot \mathbf{r}_{ik}$ for particles i, j and k, for use in three-body force calculations.
  - `update_recmot`(): Updates the recent motion quantity of an atom.
  - `flush_neighbours`(): Clears the neighbour list of an atom.
  - `add_neighbour_si`(): Adds a neighbour which has a lower index than this atom.
  - `add_neighbour_li`(): Adds a neighbour which has a higher index than this atom. Required only for three-body force calculations.

- `lattice`: Generates a lattice with the chosen geometry and chemical elements.
  - `generate_positions`() (**A**): Fills a two-dimensional array with position values for all atoms in a lattice for later use.
  - `spherical_void`() (**A**): Removes particles inside a sphere with a specific position and radius.
  - `cylindrical_void`() (**A**): Removes particles inside a cylinder with a specific position, radius and height.
  - `sphere`() (**A**): The "opposite" of `spherical_void`(). Cuts the crystal to a sphere, can leave atomic layers of simulated particles on top of the fixed ones. Preserves charge neutrality.
  - (pvt.) `remove_particle`(): Used by the particle removal functions to remove pointers to a particle and fix the particle indices.

- `ewald`: Generates the **k**-vectors and some functions of them which are needed in Ewald summations. Has two subclasses, ewald3d and ewald2d.
  - `find_vectors`() (**A**): Finds the required **k**-vectors by using a spherical or circular cutoff in reciprocal space. Tabulates some quantities depending on only **k**-vectors.
  - `calculate_h`() (**A**): Finds the required values of the integration variable h and tabulates some values depending on only **k** and h. Only for ewald2d objects.
  - `find_structure_factor`() (**A**,**D**): Calculates the structure factor $S(\mathbf{k})$ or $S(\mathbf{k}, h)$ of the entire simulation box with the current configuration.

Utilities

Some extra classes and collections of functions are required for the program to work. In addition to the tools provided by the C++ standard library, these are the utilities I have written:

- `complex`: Custom class for complex number arithmetic. Works in the same fashion as the C++ standard complex type.
- `parameters`: Class for reading parameters from a file. Reads both integers and floating-point numbers. The parameters must be sorted in a predefined order.
- `random_generator`: Generation of uniformly and normally distributed random numbers.
- `vtkwriter`: Writing positions, field values, etc. to `.vtk` files.
- `array`: Functions for easy allocation and deallocation of multidimensional arrays.
- `qots`: Function for displaying physics quotes while the program is running.

For automatic plotting the evolution of macroscopic quantities, a Matlab script is used. While working with the project, I have also created a couple of makefiles, and less useful sh scripts for converting `.vtk` files to `.xyz` and for automatically counting the total number of written C++ code lines.

6.5 Parameters

A parameters file defines a setup for a numerical experiment, and is taken into the program as the first of two input arguments. The second argument is the folder where all output files are produced. Parameters are divided into two types, global and phase-specific ones. The global parameters contain both initial values and parameters used in calculation during the whole simulation. Some of the parameters will be irrelevant to certain types of simulations, but they are all always included in the file. Here is a brief list of what the parameters file contains:

Global: System ID specifying material, the number of conventional crystal unit cells in each direction, a PBC flag, a flag turning on/off fixed particles on each x-side, the number of fixed particle layers, initial density, initial temperature, the thermostat parameter τ, time step Δt, number of time steps per visialized frame, temperature rescaling and neighbour list update, cutoff for the LJ potential, constants C and ϵ from Eq. (6.5), y-z-plane gap position and width, size of spherical flaw, crystal sphere flag, the number of simulation phases.

Space specific: Number of time steps, heat bath temperature, thermostat selection flag, dissipation coeficcient, volume expand rate in each direction, fixed particle

displacement rate for each of the x-sides, constant force vector, collective velocity adjustment vector, output level flag.

In the Ewald summation, the required parameters are calculated internally in the same way as in the Protomol MD package [15]:

$$\alpha = C\left(\frac{N}{V^2}\right)^{1/6}, \tag{6.3}$$

$$r_c = \frac{\sqrt{-\ln\epsilon}}{\alpha}, \tag{6.4}$$

$$k_c = 2\alpha\sqrt{-\ln\epsilon}. \tag{6.5}$$

In this way we change from three to two input parameters and gain a $O(N^{3/2})$ scaling. C is a constant adjusting the ratio between the time usage of the real-space and reciprocal-space sums, and is mostly set to 2.0. ϵ is the relative accuracy of the calculations. I put this to 10^{-5} most of the time, as this is more than enough for studying the phenomena the program is made for.

To prove the scaling hypothesis, I note that $V \propto N$, such that $\alpha \propto N^{-1/6}$. A particle has a number of neighbours proportional to $r_c^3 \propto \alpha^{-3} \propto N^{1/2}$. This must be computed for all N particles, giving a $N^{3/2}$ scaling for the real-space sum. The number of $\mathbf{k}$-vectors inlcuded in the reciprocal-space sum is proportional to $k_c^3 V \propto N^{1/2}$, because the density of $\mathbf{k}$-vectors is $\frac{V}{2\pi}$. This sum also happens for all particles, giving it a $N^{3/2}$ scaling.

I have presented the programming language and additional external programs I use for simulations and data analysis. The programming itself has obviously been demanding the most effort in this project. From the start, when the program only contained a crystal arranger and an Euler-Cromer integrator, it has gone through many severe structural changes before finding my preferred way of arrangement with respect to flexibility and efficiency. The majority of the results in the following sections are found with simulations differing only by their input parameters.

Chapter 7

Fracture results

The energy required to break a perfect crystal is very high, and more comparable to the energies of atomic bonds than the energy required to break a crystal with defects. I have attempted to model both these cases to study how much say defects have in the fracture process for the two materials under consideration. I will state some observations from my fracture simulations, starting with equilibrium properties of bulk NaCl and Si.

As a test of the Ewald summation technique implementation, I tried to reproduce the lattice energy of NaCl. The energy required to sublimate a NaCl crystal into a gas of Na^+ ions and Cl^- ions is -787 kJ/mol $= -4.079$ eV per particle [5]. This is, by a good approximation, the total potential energy of the lattice. With 4096 ions, the potential energy of the perfect lattice measures -4.076 eV per particle. This number is very similar for smaller system sizes. At a temperature of 300 K, the potential energy fluctuates around -4.036 eV per particle with a standard deviation of 0.001 eV per particle. More specifically, the energies from the Lennard-Jones interaction, the short ranged Coloumb interaction and the long range Coloumb interaction including self-energy correction is:

$$U^{LJ} = +0.432 \text{ eV} \qquad U^{SC} = -3.024 \text{ eV} \qquad U^{LC} = -1.445 \text{ eV} \tag{7.1}$$

All the numbers are stated per atom.

The experimentally measured cohesive energy of the silicon lattice is -4.638 eV [21]. With a temperature of 300 K, my simulation using the Stillinger-Weber potential gives the slightly higher number -4.2967. The energy is distributed among the force terms in this way:

$$U^{(1)} = -0.6938 \text{ eV} \qquad U^{(2)} = -3.6238 \text{ eV} \qquad U^{(3)} = 0.0209 \text{ eV} \tag{7.2}$$

Again, these numbers denote potential energy per atom. In the diamond structure, all atoms are able to have an angle of $70.53°$ between their neighbours simultaneously, giving a low energy contribution from the three-body interaction term.

The BDP thermostat generally does a good job cooling down edges of the simulation box when a fracture is occuring. As a lot of energy is dissipated in the process,

it takes time to conduct all the energy out of the system. Most of the kinetic energy arising is in the form of elastic waves which slowly loses energy to heat inside the crystal. Such collective motions make temperature measurements inaccurate, which they in any case will be in all non-equilibrium situations.

7.1 Tensile stress

I have performed several simulation runs of the system described in section 5.2.1. The rate at which the simulation box is stretched is normally set to 0.002 Å/fs. I will first present observations from fractures in defect-free crystals.

Figure 7.1 shows a simulation of a silicon crystal under tensile stress and at a starting temperature of 260 K. Two fracture spots are visible. The first fracture occures at about 43% strain, which is extremely high. After the fractures, elastic waves of great energy hit the sides of the simulation box and are reflected back. This is physically unrealistic, as the fixed atoms should be in just the same situation as the simulated ones. The reason I have not gone into energy transfer by such collective phenomena is that the dynamics I am interested in here happens before and during a fracture.

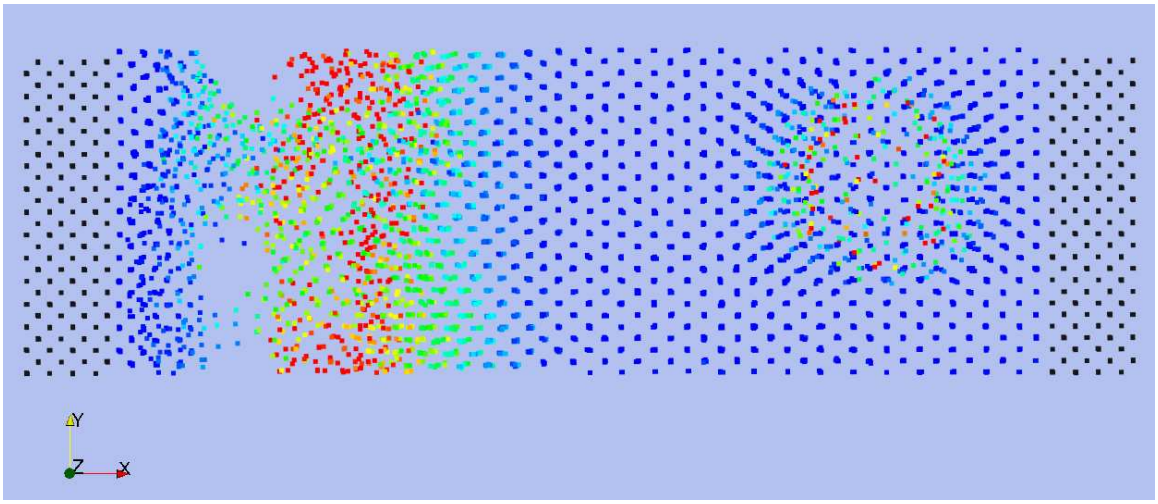


Figure 7.1: Tensile stress fracture of a crystal with 5488 Si atoms at 260 K. The colouring shows the values of the recent motion quantity. The fixed particles are shown in black.

If you neglect the thermal movement, we are dealing with a perfect crystal of atoms being streched. In my simulations of this system, fractures always start out by the breaking of single bonds. Normally, an atom in a diamond structure will have four neighbours. This means four two-body terms for each particle. At the moment this number decreases to three for an atom, this and nearby atoms start to show rapid motion. What happens is that the thermal motion kicks two atoms further

apart than the SW interaction range. A sphere of vacuum with its center where the bond breakage occured appears inside the material. This is how a fracture starts in a silicon crystal in all the cases I have observed. An example of this is visible at the right side of the crystal in Fig. 7.1.

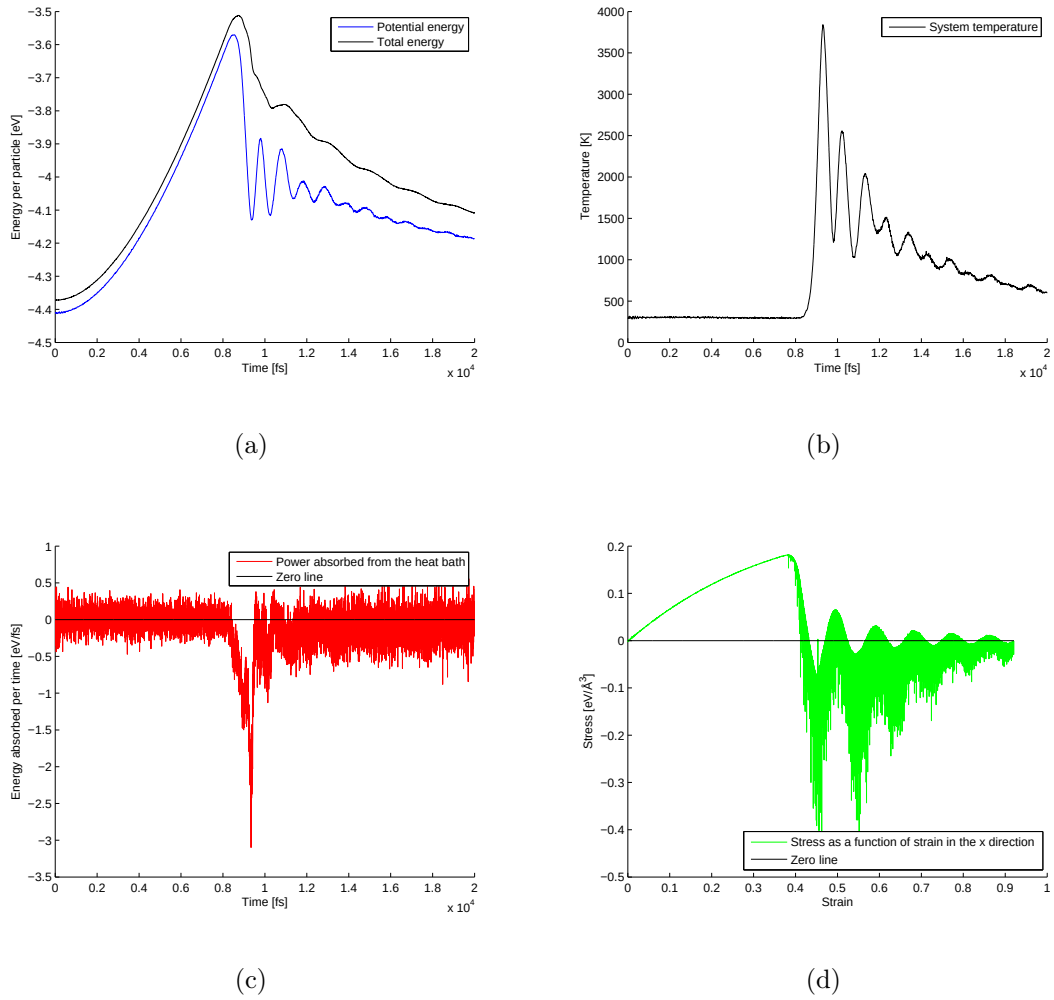


Figure 7.2: How macroscopic quantities behave when a silicon crystal fractures. The temperature is estimated with $T = 2U_k/3k_\mathrm{B}$.

Figure 7.2 shows the set of output macroscopic quantities from a typical silicon tensile stress simulation with 4096 particles. The oscillations in the potential energy are caused by the elastic waves of compression and expansion occuring in the two crystal parts after fracture. The same goes for the kinetic energy and thereby temperature. The fact that silicon builds up a lot of energy before suddenly breaking makes it a very brittle material. However, the thermostat acting on the edges of the simulation box provides a damping of these oscillations due to outgoing heat. The

collective motion of the side particles is subtracted from the measured temperature when the it is adjusted in this way. Thus, the energy has to transform to actual heat, independent particle motion, before leaving the system. This is seen to happen in great amounts right after the fracture.

By comparing the trajectories of the atoms and the macroscopic quantities, it can be established that the fracture starts when the applied stress is at its greatest. I estimate this value of the stress as the crystal's breaking stress. Long before this point, it is clear that Hooke's law $\sigma = E\epsilon$ fails. For small strain, as the law is intended, we see a linear stress-strain relationship, and Young's modulus E is measurable. From Fig. 7.2(d), I read the value

$$E = (0.670 \pm 0.004)\ \text{eV/Å}^3. \tag{7.3}$$

Silicon is an anisotropic material, and sources mention differing experimental results for its Young's modulus. [22] states an experimentally measured [100] direction Young's modulus of 130 GPa (0.812 eV/Å^3) at a temperature of 300 K. The limiting factor for precision in this kind of measurement is the interatomic potential form, which could be improved by methods briefly mentioned in section 4.

The vertical lines beneath the stress graph are an artifact of the simulation. Neighbour lists have a certain cutoff range in space and is updated after a specified number of time steps. When a critical event like a fracture occurs, atoms are moving very fast and the forming of new atomic bonds are not discovered in time because of the limited neighbour list update frequency. This creates a small shift in potential energy. The reason this is not fixed is that CPU-time usage is very sensitive to these two parameters, and they are at sufficient values to simulate what happens until the fracture occurs.

The importance of defects was well illustrated by a bug in my program for the initial silicon simulations. One two-body force term for interaction with a fixed particle was not taken into account. This resulted in a significantly lower breaking stress, and that the fracture always started where this missing bond was located.

When defects of particle deficiency grow larger, the consequenses do not change as greatly as from zero to one missing particle. Figure 7.3 shows measurement of the constant in Griffith's relation (Eq. (2.1)). The initial number of particles (before introduction of the defect) is 4096. The relation is well reproduced, with a mean of 0.2657 and a standard deviation 0.0087.

When sodium chloride is stretched, it does not build up potential energy in the same violent fashion as the silicon crystal. As there are both strong attractive and repulsive forces which cancel each other, it does not take much energy to break a sodium chloride crystal. Regardless, I observed a fracture at about 63% strain in a simulation. Before a fracture occurs, the atoms float about lazily in a substance of low density. There is, however, evidence of crystal structure under the whole simulation.

The NaCl crystal breaks as softer material, often cut straight into two pieces separated by a y-z-plane. The two pieces stay charge neutral, which is energetically favourable. The velocities of the particles make a less considerable jump then in the

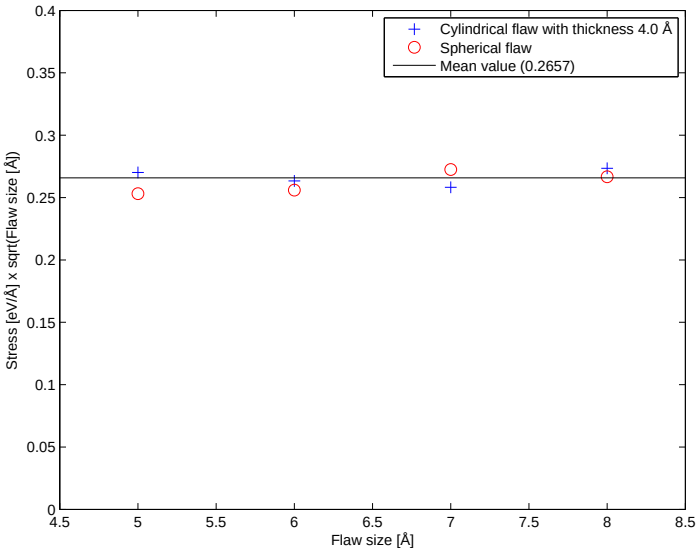


Figure 7.3: $\sigma_f\sqrt{a}$ as a function of a for silicon. The "flaw sizes" a are the radii of the spherical and cylindrical voids.

silicon crystal case when the pieces drift apart. In many cases, the fracture occurs close to the edge of the simulation box, probably because of the constant distance between the fixed atoms, making this a special place. In Fig. 7.4, a spherical void of radius 7 Å is inserted in the middle of the box. This causes the fracture to occur at this point, although this is not always the case with NaCl crystals containing defects. Threads of ions are often seen connecting the two pieces right after a fracture, but these disintegrate easier than corresponding silicon atom strings.

The small build-up of potential energy seems to happen because of the long-range part of the Coloumb interaction. In a simulation without the Ewald summation over **k**-vectors, the crystal breaks in the same fashion, but at a lower strain. The potential energy has a deviation of roughly 2.5% during the course of the simulation, as can be seen in Fig. 7.5(a). Figure 7.5(b) shows equally few signs of violent activity. The stress is kept at the same level for a long period, bearing signs of the stretching process being quasistatic. The defect size does not seem to influence the breaking stress of the NaCl crystal. The dynamics stays the same for different spherical void sizes, but a greater defect has a bigger probability of influencing where the fracture will occur.

Negative stress, as for the Si crystal, is caused by particles of high kinetic energy compressing both parts of the crystal right after the fracture. There are some oscillations caused by elastic waves, though not as strong ones as in a newly fractured Si crystal.

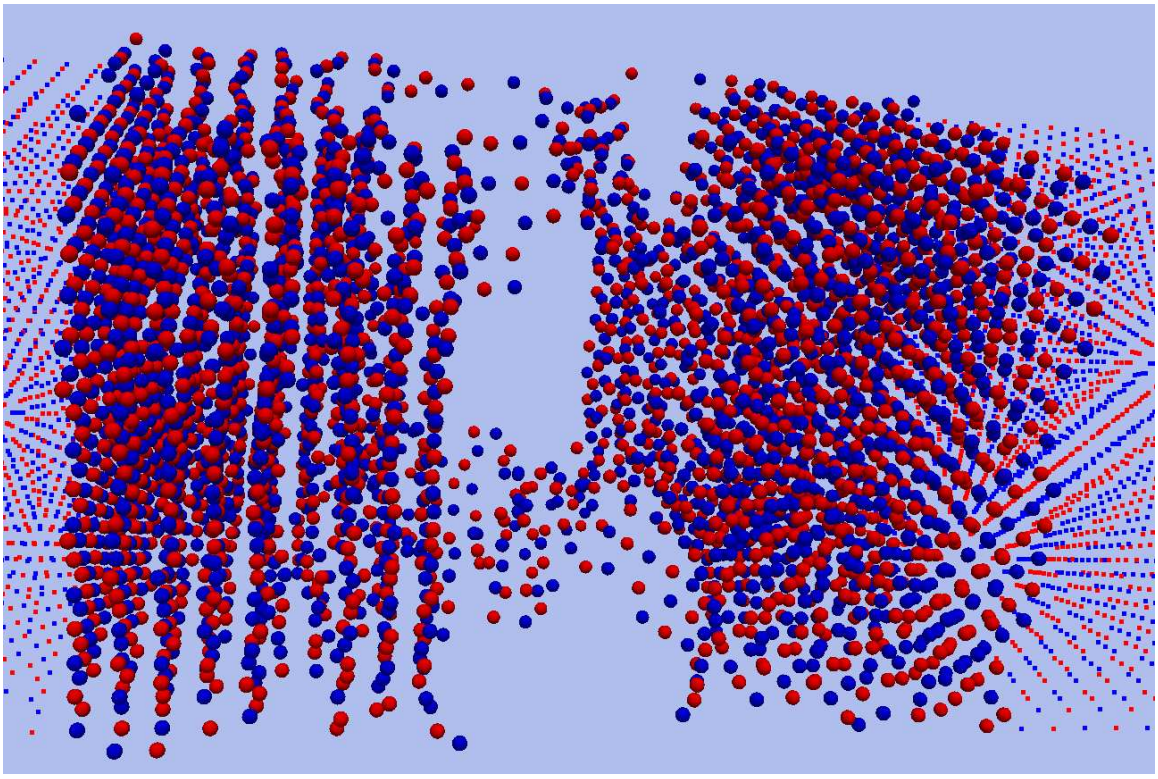


Figure 7.4: Tensile stress fracture of a NaCl crystal with 4096 atoms at 300 K.

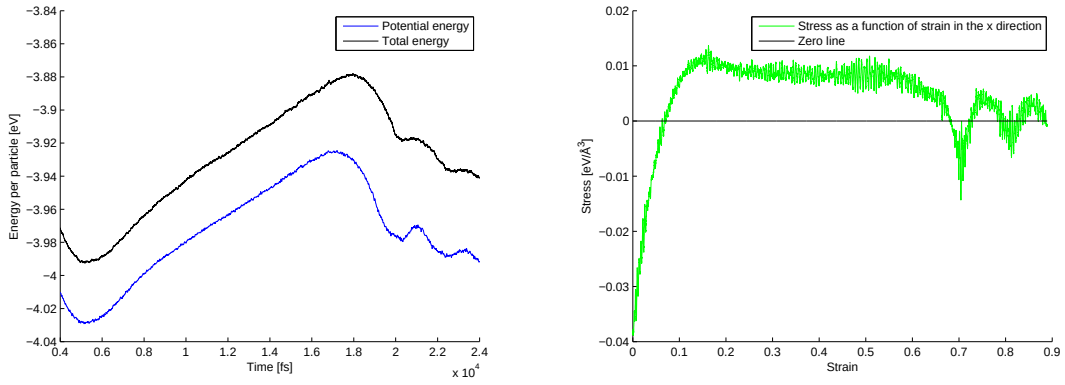
7.2 Shear stress

When shear stress is applied to a silicon crystal, the atomic layers will be skewed and the potential energy will build up. At some critical point, this energy will be converted into a great amount of kinetic energy, causing a phase transition at some points in the simulation box.

In Fig. 7.6, these areas emerge at the side of the box, making the energy distribution symmetric along the x axis. Sometimes, the phase transition happens in the middle of the box. The areas that do not become liquid keep their crystal structure. Silicon compresses as it melts, so liquid atoms have more neighbours. Therefore, their potential energy is higher, which is visible in Fig. 7.6(a).

Sodium chloride shows a quite different behaviour when deformed with shear stress. The atomic layers get more and more lopsided before the layers slip vertically at an y-z-plane in the middle of the box. The system then looks almost like a crystal in equilibrium, but some heat is generated. The cycle of tilting and atomic layer slipping repeats itself as long as the fixed particles are in motion. There are big temperature fluctuations, but a mean temperature of about 650 K is established as all the heat generated from the deformation is in average conducted out. This applies to a fixed particle speed of 0.002 Å/fs.

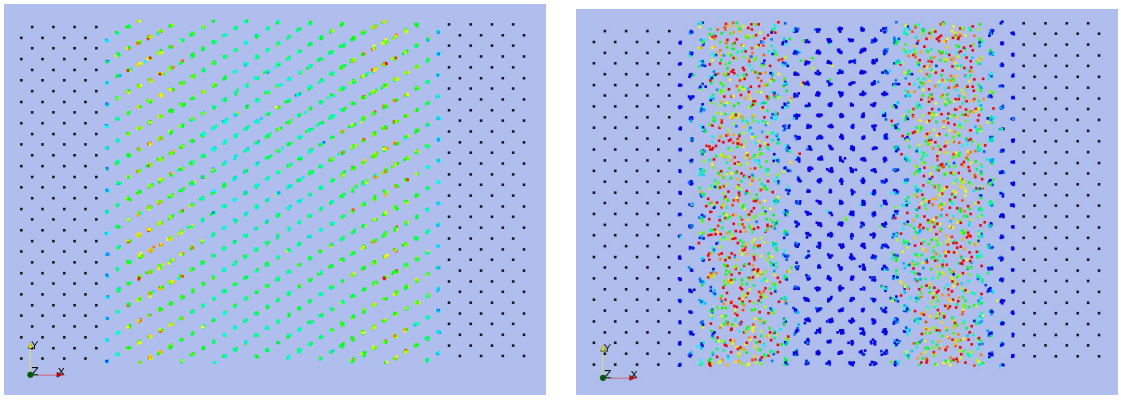
There is little difference in the dynamics for a simulation without long-range Col-



(a) (b)

Figure 7.5: How the energy and stress varies when a NaCl crystal fractures.

oumb interactions. The potential and total energy graphs look rugged in both cases. There is a slight spring-like effect when the atomic planes slip with full interactions turned on. This is a process which happens simultaneously everywhere in the middle y-z-planes. The long-range forces are expected to contribute significantly in the case of a large-scale displacement from equilibrium. In this case, a counteracting force arises.



(a) Right before fracture

(b) Right after fracture

Figure 7.6: Shear stress on a crystal with 4096 Si atoms at 260 K. The colouring shows the potential energy per particle. The fixed particles are shown in black.

Chapter 8

Contact and adhesion results

8.1 Sphere-surface contact

I have performed computational contact experiments for the systems described in section 5.2.3. I will first present the results from the sphere-surface systems, which I have used the most time on. For the calculations to be executed in a reasonable time, I have limited the system size to 4232 simulated and 1600 fixed atoms for silicon, and 2106 simulated and 2048 fixed atoms for sodium chloride. The pseudo-gravitational force has been varied logarithmically in the interval from 0.01 eV/Å to 30.0 eV/Å. An example simulation is visualized in Fig. 8.1.

Results for the measured contact area using Eq. (2.6) are presented in Fig. 8.2. The spheres do not seem to compress much because of the force affecting their top halves. In a large force interval, They are pushed towards the surface and the bottom atoms rearrange slightly, but the energy seems mostly stored in an elastic deformation. When the force reaches a certain threshold, the sphere collapses under its own "weight". The center of mass x coordinate and the measured contact area tells the same story. The point at which this collapse happens varies slightly as there are great fluctuations in these kind of simulations. Nevertheless, pushing an asperity towards a surface creates a binary situation where the material either has undergone a plastic deformation or not. The two materials act similar in this respect, but the microscopic processes responsible for the effect are different.

In the case of silicon, the lowermost atoms display a liquid-like behaviour, as expected from the property of higher liquid density than solid density. The sodium chloride sphere compacts itself vertically and extends horizontally, not mainly by compression. The slip mechanism also occuring in the shear strain experiment is responsible for reducing the number of atomic layers in the vertical direction. The configurations preceding and following the slip motion are shown in Fig. 8.3.

When measuring the stress between a ball and a surface of similar materials, the system size has to be big to get a clear image of what the stress distribution looks like. This is especially the case for silicon, where the particles close to the surface arrange quite arbitrarily. Therefore, I have not only averaged the stress measurements

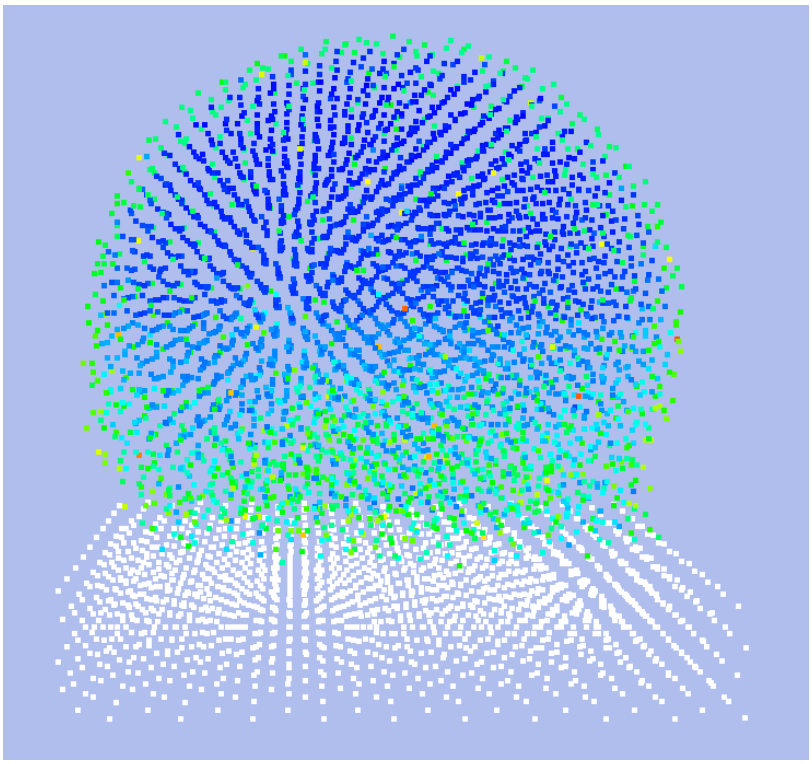
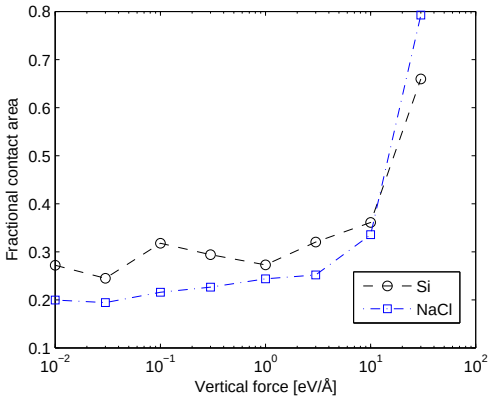


Figure 8.1: A ball of silicon pushed against a silicon surface with a force of 30.0 eV/Å. The colours show potential energy. The discontinuity in this colouring is caused by the fact that only the upper half of the particles are subject to the ficticious gravitational force. Perspective projection.

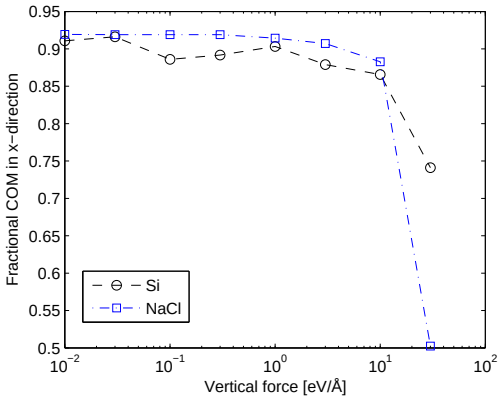
over time, but over 8 different initial configurations (random seeds). This makes the structure of the stress graph (Fig. 8.4(a)) more apparent. The graphs for the different runs contain trends similar to the result for the bent crystal in [3], where a harmonic potential is used. They also contain a great deal of noise, like they should, seeing that the bottom sphere particles arrange randomly. The outer moat of attraction is caused by adhesive forces between atoms not pushed close enough together to repel. Around 20000 dynamic particles were used in the simulations.

The stress graph for the NaCl system is very different. The signs of the stresses have the opposite character: The middle particles attract each other slightly while the outer particles repel each other (see Fig. 8.4(b)). The graph looks the same for different initial conditions, with the exception that the ball may be displaced one atomic distance in some direction on the surface. The sphere particles arrange very regularly. Small, regular zigzag patterns in the stress graph are caused by oppositely directed forces between the ions. Around 7000 dynamical particles were used in the simulation. The reason for the odd surface stress graph is investigated in the next section.

In both the Si and NaCl cases, the sum of forces between the sphere and surface

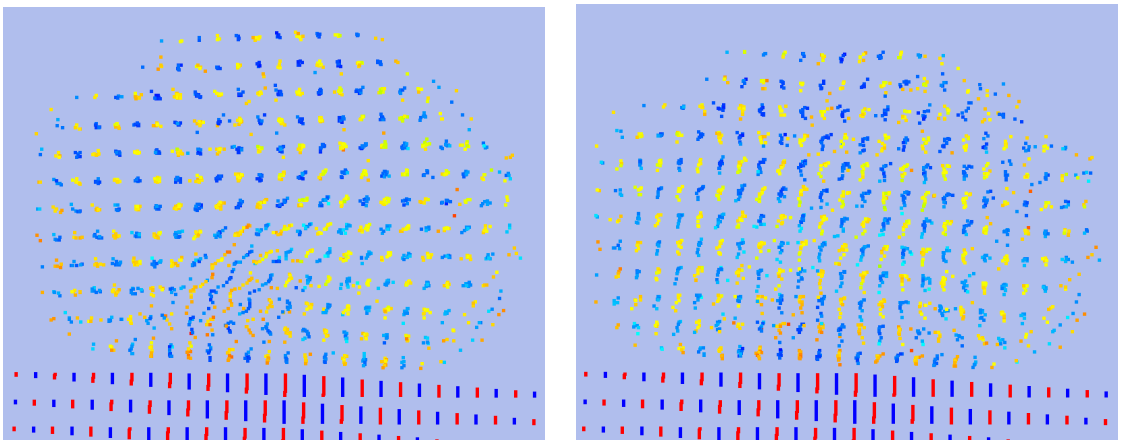


(a) Contact area in units of one sphere cross-section. Average over two runs per point.



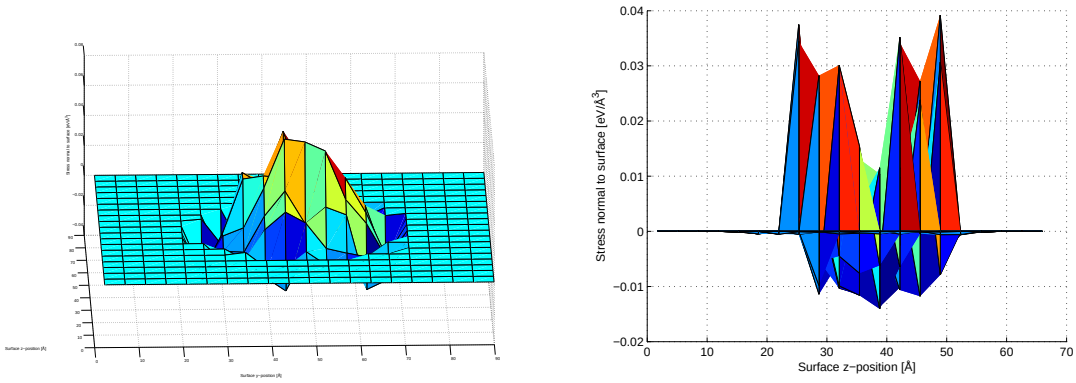
(b) Center of mass in the x direction in units of one sphere radius.

Figure 8.2: How quantities vary when an increasing vertical force is applied to a sphere on a surface.



(a) (b)

Figure 8.3: The deformed NaCl sphere before and after a slip motion. A height difference is evident. The colouring shows the atoms' potential energy, excluding long range (recprocal sum) electrostatic energy.



(a) Si system, averaged over 8 runs.

(b) NaCl system.

Figure 8.4: Contact stress between a ball and surface composed of the two different materials in study.

particles lie within 1% of the applied pseudo-gravitational force. This validates the results by assuring that Newtons second law is fulfilled.

8.2 Sphere segment-surface contact

When the forces behind the NaCl stress graph are examined more carefully, the Lennard-Jones potential appears to be the culprit behind the great repulsion at the sides of the contact area. The ionic interactions actually lessen the effect of this and cause the attraction in the middle of the contact area. The short-range part of the Coloumb forces do all the work, while the long ranged part do not contribute significantly to this process. The stress graph looks very different when the sphere segment system is used. The upper fixed layers and sphere segment propagates with a constant speed of $2 \cdot 10^{-4}$ Å/fs. The most important difference from the whole sphere system is that fixed particles keep the simulated particles strongly in place at the positive x side of the simulation box. Therefore, global oscillations are avoided. Natural oscillations are nearly certainly the reason behind the stress spikes. According to the particle trajectory animations, the sphere tip very slightly over to the left and right side of the contact point, which cause the contact particles at one side to come closer to the surface and experience a vastly bigger LJ force. This creates a peak which is significantly taller than the mean stress across the contact area. When averaged over time, this effect is the prominent feature of the stress graph all around the edges.

Without the oscillations, when the sphere segment system is slightly compressed between two surfaces, a more uniform stress is measured. Figure 8.5 shows such a

stress graph, with a total force between the dynamical particles and the lower fixed particles of 131 eV/Å. This number can seem very high, but the force is applied in a more gradual manner than before, and the sphere is still not collapsed. With a little more applied compressive strain, the sphere segment will deform in much the same way as before, creating a bigger contact area and lowering the total force. NaCl has a much stiffer crystal structure than Si, which in this case makes it more durable in the absence of a more sudden impact. The intact stepped crystal structure makes a uniform stress across the contact area seem likely.

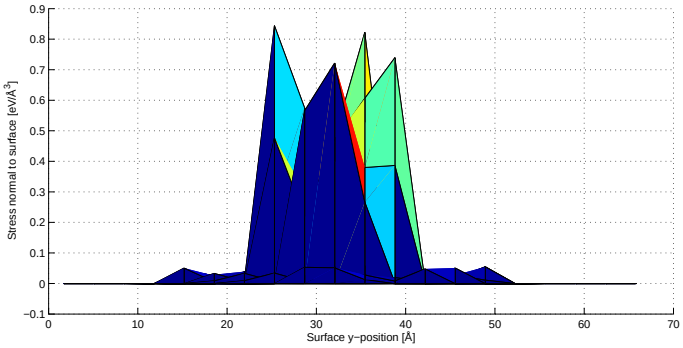


Figure 8.5: Contact stress between a NaCl sphere segment and a NaCl surface. The applied compressive strain is 4.1%.

In another attempt to explain the strange sphere-surface stress graph, I observed that the lowermost atomic layer creates a deformation in the rest of the sphere for large applied forces. However, this deformation is too small to create such a big effect, and it occurs also in the sphere segment system. I refer to the deformation, which is clearly visible in Fig. 8.6, as the bow phenomenon. The figure is from the same simulation run as Fig. 8.5. The effect of the bow phenomenon can be seen as a small ring outside the stress from the lowermost atomic layer. It is a peculiar phenomenon which most probably happens in real systems, but the effect is not significant.

The stress graph of the silicon system still has stability issues, but the results from sphere-surface interactions are reproduced.

In summary, the stresses I have found for the different materials seem to carry information on nothing else than the microscopic structure of the compressed asperity. The paper [3], which works only with LJ and harmonic spring potentials, describe similar results for the corresponding microscopic structures. Different potential symmetries will however induce differing reactions to an asperity-surface interaction, leading to either destruction or some form of conservation of the material's default bulk crystal structure.

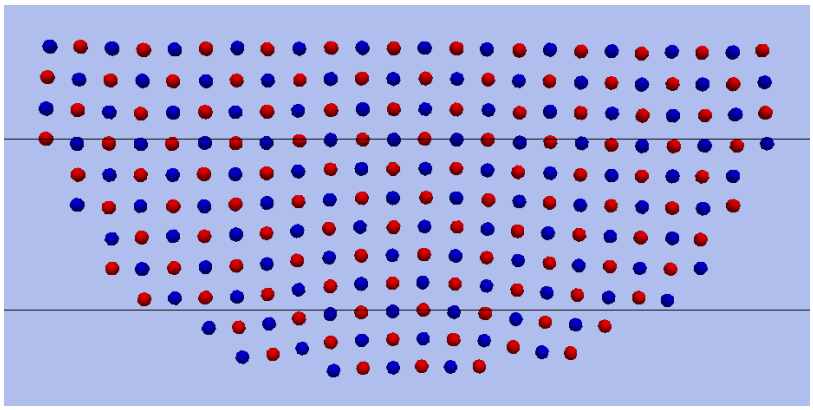


Figure 8.6: Arrangement of one x-y-layer of NaCl sphere segment ions. The applied compressive strain is 4.1%. The straight lines are just eye guides.

8.3 Adhesive energy

The adhesion experiments gives resulting atom trajectories which look realistic for their respective materials. The sphere segment setup is again used, using the same propagation speed of the upper surface, this time both for pushing the surfaces together and pulling them apart. Another difference is that the sphere segment will not start in slight contact with the lower fixed particles, as in Fig. 5.4. Instead, I place a gap of 4 Å under the sphere segment in order to measure the energy needed to stick the surfaces together. This also gives a clearer picture of how the adhesion process takes place. I will refer to how far the upper fixed layer is forced down or up from its default position as the *indentation distance*, denoted d. In the initial position, the indentation distance is $d = -4$ Å. Upon interactions between the surfaces, this will not be the same as the center of mass x coordinate, which is measured in section 8.1.

Silicon is the material displaying the most interesting behaviour in this case. I use a sphere segment with a height of 5 unit cells and upper radius of 15 unit cells, constituting 3963 atoms. The adhesive forces start to pull the lowermost layer of the sphere segment in quite a violent fashion at $d = -2.1$ Å. Notice that this will contribute with a negative indentation energy. At $d = 0.4$ Å, the forces from the bottom fixed particles change sign so the interaction is repulsive. If pushed further, the sphere segment is deformed, as in the pure contact experiments, and this requires a lot of energy. When pulled apart again, the interaction does not turn repulsive before $d \approx -7$ Å, and the strings of atoms between the surfaces do not break completely until $d \approx -30$ Å. These numbers are for a maximum indentation distance of 3.2 Å. Figure 8.8 shows a snapshot of this simulation run. The energy required to pull the surfaces apart is much bigger than the energy required to join them.

The sodium chloride system is again made much smaller, with a sphere segment height of 3 unit cells and upper radius of 7 unit cells. Only 612 dynamical particles are in the simulation, but this seems to be enough to get a qualitative picture of what is

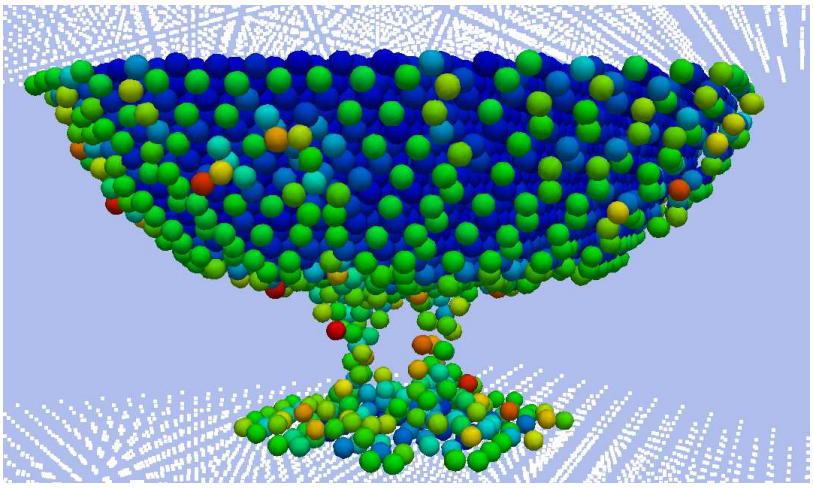
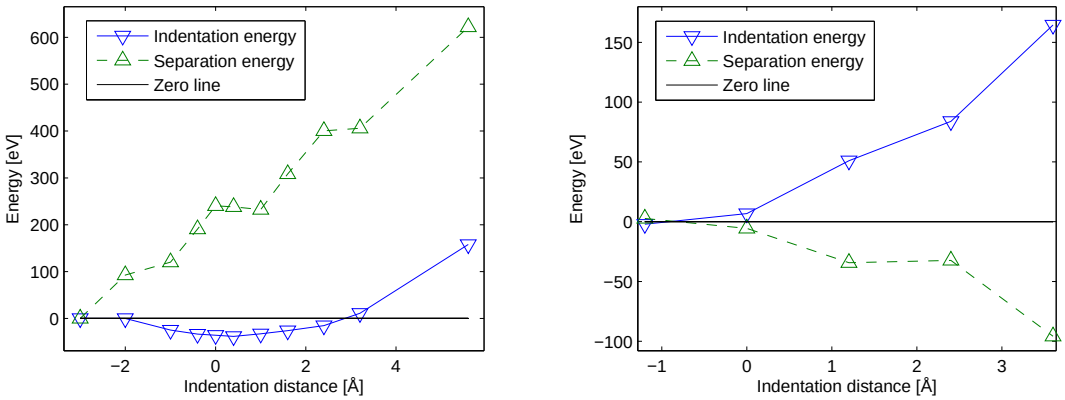


Figure 8.7: An Si asperity pushed into and separated from a Si surface. $d_{\max} = 3.2$ and $d_{\text{current}} = -12.0$ Å. The atoms are coloured from blue to red by their potential energy.

happening and approximate numerical results. The material reacts in a very different way to the indentation and separation from silicon, as expected. The earlier contact and fracture experiments both give clues to its behaviour. The adhesive forces in the interface are very small, not very noticable in an animation of the trajectories, and they change to a small repulsion already at $d = -2.1$ Å. If compressed enough, the sphere segment will deform by slip movements. In rare cases, this can form a multicrystal, a structure composed of lattices with different axes. I have only observed one such case, with a maximum indentation distance of $d = 2.4$. In the same simulation run, a horizontal monolayer of NaCl connected the two surfaces until $d = -9$ Å. It hung between them like a carpet, in much the same way as the torn flakes in NaCl fracture simulations (see the top of Fig. 7.4). This made the separation energy, which normally is negative for NaCl, much higher than expected. I have discarded the numerical data from this run and consider it a peculiarity. In short, it takes some effort to connect the two NaCl-interfaces when a deformation is needed, but it is very easy to pull them apart again.

Figure 8.8 shows the numerical data from adhesion runs with the two materials. In the Si case, a lot of energy is needed to pull the surfaces apart, while energy is actually gained in the NaCl case. It is a reasonable hypothesis that the magnitudes of both the indentation and separation energies increase with maximum indentation distance and maximum interface contact area achieved. The second of these quantities can explain the apparent "jumps" in separation energy for silicon. As the sphere segment is pushed downwards, the second layer, third layer and so on of Si atoms will start to feel interactions with the fixed particles and eventually adhere to the lower surface. The probable reason why this is not so apparent in the indentation energy is that the adhesion creates big velocities close to the fixed particles, which is transformed to



(a) Silicon.

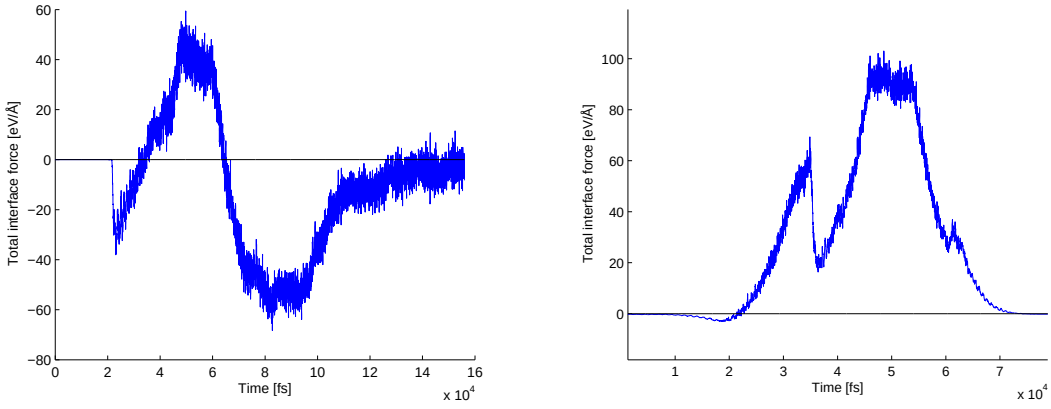
(b) Sodium chloride.

Figure 8.8: The energy required to put the surfaces in contact and pull them apart again, respectively. For various maximum values of d.

heat and conducted out. Such an effect does not exist for NaCl, though it can have several small stages of deformation.

Heat plays a big role in the adhesion process. I have not measured the amounts of heat systematically, but will give some estimates. Almost all the energy that is gained from adhesion is lost as heat. In addition to the energy needed for deforming the systems to higher potential energy states, this is what is responsible for the apparent hysterisis in the process. The temperature is kept at 300 K, as in most of my simulations. If this were not the case, the indentation and separation energies would have been closer to having the same value with different signs. As the figure shows, much more heat is generated in the Si system. Here, roughly half of the separation energy goes to heat the system. Meanwhile, the NaCl system goes into a higher potential energy state when compressed. When separated again, this energy is lowered, and it actually gains heat from the outside. The deformations themselves lower the potential energy, because a flatter NaCl surface is more stable, but this energy is not gained before the separation process happens.

The last figures I want to present from these experiments are two examples total interface force plots as functions of time (Fig. 8.9). On the silicon graph, it is visible how suddenly the adhesive forces are turned on. The force varies almost linearly (elastic Hook-spring-like) when the sphere segment is pushed further in, except for some irregularities having to do with plastic deformations. The plateu corresponds to an equilibration phase when d is at its maximum. The adhesive forces are very strong for a long period when the surfaces are separated again. On the sodium chloride graph, adhesive forces are barely visible in the entry phase. Again, the force varies roughly linearly with time (and displacement), only interrupted by a sudden grand deformation. The NaCl runs where such a deformation does not occur have



(a) Silicon.

(b) Sodium chloride.

Figure 8.9: The energy required to put the surfaces in contact and pull them apart again, respectively. For various maximum values of d.

quite a symmetric force graph. This is also visible in Fig. 8.8(b), where only heat dissipation stops the two energies from cancelling out.

8.4 Friction experiments

Measuring quantities related to friction from atomic simulations is a very ambitious task. I have only attempted to simulate friction using the sphere-surface contact system, and the sphere, which unevitably begins to roll, may do a poor job at describing a rough macroscopic surface asperity in a time-dependent situation. While the spheres rolled/slipped across the infinite layers of fixed surface atoms, more sphere particles became attached to the surface, tearing off more and more of the sphere. Data for the coupling between contact area and frictional force is therefore only reliable in the start of the simulation.

The data contains great fluctuations, but large-scale shapes of the A_c-F_{friction}-distribution are visible. In the NaCl case, the frictional force seems to be distributed equally around zero at all times, which I have made no effort to explain. Interestingly, the silicon data seem to indicate a linear relationship between A_c and F_{friction}, though the fluctuations are of the same size as the increased force from small to large contact area (see Fig. 8.10). For different values of the driving velocity and the pseudo-gravitational force, I see no general trend of change in the frictional force. This may or may not be accurate, but is either way only valid for a single asperity. The larger-scale mechanisms of friction, which cause F_{friction} to vary linearly with the normal force, are explained in [2].

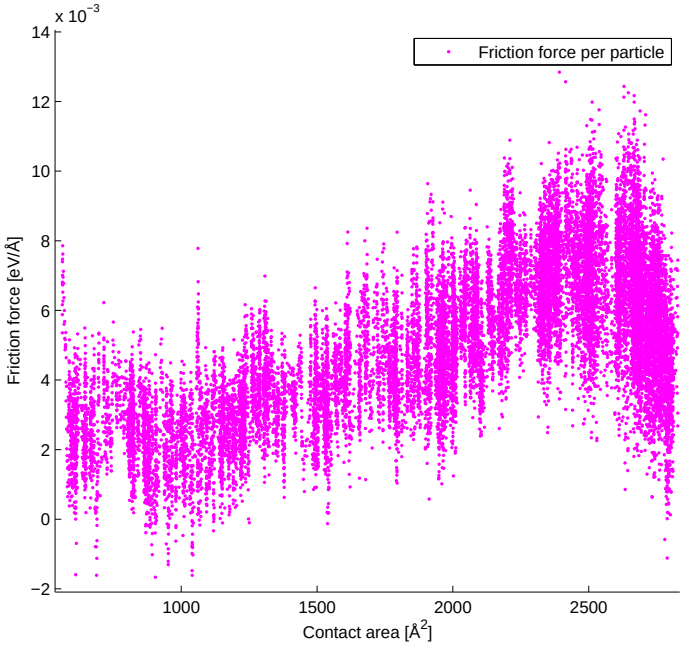


Figure 8.10: F_{friction} vs A_c for a silicon friction experiment with about 4000 particles. One data point per time step.

Chapter 9

Concluding remarks

Nacl is easy to pull apart, but hard to compress. Si is hard to pull apart, but easy to compress.

Bibliography

[1] E. Gerde and M. Marder, "Friction and fracture," *Nature*, vol. 413, pp. 285–288, Sep 2001.

[2] J. Ringlein and M. O. Robbins, "Understanding and illustrating the atomic origins of friction," *American Journal of Physics*, vol. 72, no. 7, pp. 884–891, 2004.

[3] B. Luan and M. O. Robbins, "The breakdown of continuum models for mechanical contacts," *Nature*, vol. 435, 2005.

[4] F. Gilabert, A. Krivtsov, and A. Castellanos, "A molecular dynamics model for single adhesive contact," *Meccanica*, vol. 41, pp. 341–349, Jun 2006.

[5] "Wikipedia, the free encyclopedia." http://www.wikipedia.org.

[6] M. A. Omar, *Elememtary Solid State Physics*. revised printing ed., 1993.

[7] J. M. Thijssen, *Computational physics*. New York, NY, USA: Cambridge University Press, 1999.

[8] G. Bussi, D. Donadio, and M. Parrinello, "Canonical sampling through velocity rescaling," *The Journal of Chemical Physics*, vol. 126, no. 1, p. 014101, 2007.

[9] D. Griffiths, *Introduction to Quantum Mechanics*. 2nd edition ed., 2005.

[10] A. Rahman, "Correlations in the motion of atoms in liquid argon," *Phys. Rev.*, vol. 136, pp. A405–A411, Oct 1964.

[11] D. E. Smith and L. X. Dang, "Computer simulations of nacl association in polarizable water," *The Journal of Chemical Physics*, vol. 100, no. 5, pp. 3757–3766, 1994.

[12] L. X. Dang and P. A. Kollman, "Free energy of association of the k+:18-crown-6 complex in water: A new molecular dynamics study," *The Journal of Physical Chemistry*, vol. 99, pp. 55–58, Jan 1995.

[13] J.-P. Hansen and I. R. Mcdonald, *Theory of Simple Liquids*. Academic Press, 3 ed., April 2006.

[14] P. Vashishta, R. K. Kalia, J. P. Rino, and I. Ebbsjö, "Interaction potential for *sio*2: A molecular-dynamics study of structural correlations," *Phys. Rev. B*, vol. 41, pp. 12197–12209, Jun 1990.

[15] T. Matthey, "Plain Ewald and PME." May 2005.

[16] M. Kawata and M. Mikami, "Rapid calculation of two-dimensional ewald summation," *Chemical Physics Letters*, vol. 340, no. 1-2, pp. 157 – 164, 2001.

[17] F. H. Stillinger and T. A. Weber, "Computer simulation of local order in condensed phases of silicon," *Phys. Rev. B*, vol. 31, pp. 5262–5271, Apr 1985.

[18] D. C. Rapaport, *The Art of Molecular Dynamics Simulation*. Cambridge University Press, 2 ed., April 2004.

[19] V. K. W. Cheng and E. R. Smith, "The electrostatic potential energy at a plane surface of a point ionic crystal. ii. numerical results for an ion near the (100) kcl surface," *Journal of Physics A: Mathematical and General*, vol. 20, no. 10, pp. 2733–2742, 1987.

[20] W. T. Vetterling and B. P. Flannery, *Numerical Recipes in C++: The Art of Scientific Computing*. Cambridge University Press, February 2002.

[21] X.-P. Li, D. M. Ceperley, and R. M. Martin, "Cohesive energy of silicon by the green's-function monte carlo method," *Phys. Rev. B*, vol. 44, pp. 10929–10932, Nov 1991.

[22] Ioffe Institute, "New semiconductor materials. characteristics and properties." http://www.ioffe.ru/SVA/.