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

May 2010

Det matematisk-naturvitenskapelige fakultet
Universitetet i Oslo

Contents

Chapter 1

Introduction

I have had trouble creating a title which captures the essentials of the project in few words. The one I landed on mentions the method used and the systems investigated, hopefully without being too general. The idea behind the project is investigating how solid structures behave, and how the type of chemical bonds between atoms influences the behaviour. I will address two types of daily occurring processes, fractures and friction. Being the more complex and less understood process, friction has the biggest emphasis in the thesis, mostly through numerical surface contact experiments.

Of the four fundamental forces, electromagnetism is the only relevant one at the nanoscale. Gravitational forces are neglected because its characteristic strength is as low as approximately 10^{-36} times the strength of electromagnetism. Objects the size of a car (about 10^{28} atoms) are needed for gravity to become significant. Furthermore, the strong nuclear force acting between quarks is important in nuclear physics, but quark degrees of freedom are eliminated in effective theories for atoms and molecules. An atomic nucleus is normally considered as a point particle, only interacting electromagnetically with electrons and other nuclei.

We are left with the theory of quantum electrodynamics, a theory developed in the 1940s to describe the interactions between electrically charged particles. Photons are the force carriers for these interactions. On the lowest level, the reason for electrostatic repulsion and attraction between particles is discrete cases of scattering induced by the transmission and absorption of virtual photons. The degrees of freedom of the photons are eliminated by averaging over these interactions and creating a continuous theory. The result is classical electrodynamics, formulated by Maxwell's equations.

In low-energy physics, which constitutes every process that occurs on the nanoscale, particles move at speeds much lower than the speed of light c. For example, medium-sized atoms vibrate with a speed of around $10^{-6}c$ at room temperature. We therefore do not have to take into account Einstein's theory of special relativity, and can approximate quantum field theory by quantum mechanics. Together with classical electrodynamics, this theory has been used for atomic energy calculations since the 1920s.

The field of thermodynamics and statistical physics also has an important place in this project. The systems to be studied include a great number of degrees of freedom. Even more of these, constituting the surroundings, are neglected and

modelled only by their average effect on the system. In such big systems, meaning big in comparison with a single atom, the effect of the second law of thermodynamics can be accepted as a fact. The entropy of the system will always decrease, meaning that all forms of abundant energy are eventually converted to heat. As this heat is dissipated and conducted out of the system, it does not return in other forms. This process is irreversible.

In the macroscopic world, irreversible processes with a lot of energy dissipation dominate everything we perceive. A car engine is said to be efficient at transforming the energy in fossil fuel to mechanical power. Yet a lot of heat (and negligibly, sound energy) is dissipated in the engine. Some of the energy is used to heat the drive shaft and connected parts, the catalyser, the exhaust pipe and the tires. Finally, friction in the ground and air dissipate energy into heat in the surroundings. Only about 250 kJ, the energy contained in 10 ml gasoline, is used to accelerate the car from 0 to 80 km/h.

My main investigative method is computational programming. Traditionally, physics has been divided into theoretical and experimental branches. Recently in the history of physics, in the second part of the 20th century, computational physics became established as a third branch. In a sense, it is methodologically positioned between the other two. Theoretical physics underlies a computational physics simulation, and the results from the simulation are compared with experimentally obtained results. The method is often called numerical experimentation, since theory and approximations are tested and verified. In most cases, this can not be done analytically by the hand of a theoretical physicist.

Mathematical problems with roots in real life, specifically linear algebra problems, differential equations and integrals, often require an automated problem solver. Pretty, continuous functions are discretized (sampled at a finite set of points) and stored in a computer as a large amount of numbers. These numbers go through a programmed procedure to produce results. As computers are logical in nature, their natural language is mathematics. Programming intricate problems is still a big challenge, and requires knowing a programming language and general programming techniques well. Computational science has contributed with important advances to molecular biology, material science, geophysics, solar astronomy, and fluid mechanics, to mention some fields.

Fracture mechanics is the study of why and how material failure occurs. The processes involved in fractures are very different in nature for e.g. ceramics, metals and polymers. The simplest form of analysis in this field is linear elastic fracture mechanics (LEFM), where Hooke's law is assumed an actual law. More advanced treatment is possible by examining dislocations in big systems, as lattice dislocation interactions are important in fractures of many materials. My systems are too small for investigating dislocations, but the used interatomic potentials will describe the mechanics better than LEFM, making the complexity of this project close to the average in fracture simulation studies. I study tensile (mode I) and shear (mode II) fractures, characterizing the elastic and plastic processes in fractures of the two materials Si and NaCl.

Tribology is the science of how material surfaces interact. It is a big field, and the one most relevant to the phenomena of friction. The interactions between two surfaces in contact determine the outcome of rubbing them together. Rougher

surfaces give a higher friction force. Lubricants and other contaminants can reduce this force. Adhesion is important in stick-slip motion, that is, the surfaces sticking together and slipping again, periodically [1]. The effect is easily seen and heard when sliding one rubber sheet on another with a low velocity. I attempt to model surface roughness by a single asperity interacting with a plane surface. I investigate the compressive and tensile stress at the interface of the two surfaces, and the energy involved in adhesion.

The next chapter presents what is to be studied, nanoscale mechanical processes. What the systems consist of and what will happen to them is discussed. Details concerning the two materials, silicon and sodium chloride, are also listed. Chapter 3 presents the method for the numerical 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. I explain how the system is represented inside my computer program, and some numerical methods used to treat the systems in physically realistic ways. Chapter 6 contains technicalities of the program and shows a flowchart of the executed procedures.

The results of the numerical simulations are presented in chapters 7 and 8, corresponding to the two main classes of systems mentioned in chapter 2. Finally, I put the results into perspective and make some concluding remarks regarding the outcome of this project and possible improvements and continued work.

I want to thank my advisor, Anders Malthe-Sørenssen, for advice, fruitful discussions and insight into physical results. Additionally, my co-advisor Morten Hjorth-Jensen has provided constructive critique of the thesis. I want to acknowledge the support and motivation from fellow students in the Computational Physics research group. Weekly seminars and a good atmosphere helped making my studies enjoyable.

Chapter 2

Systems of interest

In this project, I will investigate 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 parallel 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 the 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 more 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 interesting processes occuring 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. The

first chosen material is sodium chloride, better known as rock salt, which has ionic bonds. The second is a semimetal with covalent bonds, silicon, used in modern-day electronics and solar cells.

2.1 Fracture mechanics

It is a well-known fact that the usual reason for fracture in a solid state material is lattice 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. Griffiths relation predicts a proportionality between the tensile stress at which a fracture occurs, σ_f, to the flaw size a [2]:

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

The expression for the proportionality constant was found by Griffith and improved by G. R. Irwin. The more general form used by Irwin is [2]

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

Here, γ_s is the surface energy density in the material. When a surface is created inside the material, some potential energy corresponding to γ_s is lost. γ_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 occurs. 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. The relation is intended for a linear crack extending through a plate. For other flaw shapes, the constant factor is slightly different and can be calculated, but I will not go into more details on the exact value of the factor.

Figure 2.1 depicts tensile and shear forces deforming a cube. The lengths and forces shown can be used to define stress and strain. These quantities are actually cartesian tensors, but I assume an isotropic solid. The formulae below can be used in the general case when written in differential tensor forms. Tensile strain ϵ and stress σ are

$$\epsilon = \frac{\Delta L_x}{L_x}, \qquad \sigma = \frac{F_\sigma}{A_{yz}}, \tag{2.3}$$

where A_{yz} is the cross-sectional area normal to the tensile force. Shear strain γ and stress τ are

$$\gamma = \frac{\Delta x}{L_y}, \qquad \tau = \frac{F_\tau}{A_{yz}}. \tag{2.4}$$

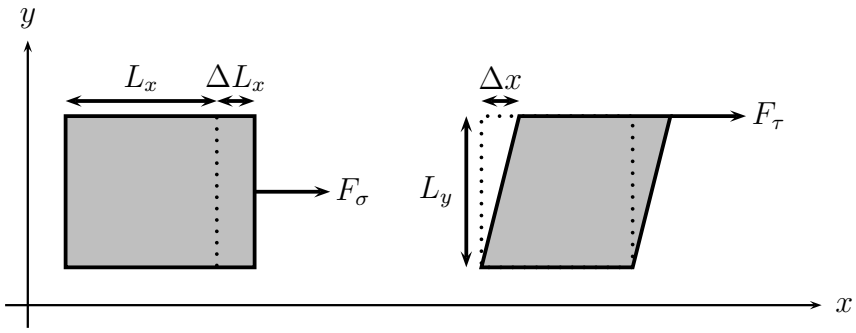


Figure 2.1: Illustration of how tensile and shear forces (respectively) deform a solid, and definitions of the lengths used in the stress and strain formulae.

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

$$\sigma = E\epsilon. \tag{2.5}$$

Using this relation, it is an easy task to estimate E. Although it is not a part of this project, estimation of γ_s and γ_p is also possible, so that Eq. (2.2), or its modification for a specific flaw shape, may be verified. In the case of a harmonic solid, Hooke's law is exact (but no fracture occurs). The harmonic spring potentials ($U = \frac{1}{2}kr^2$) between atoms will act as coupled springs when the solid is stretched or compressed.

In a real solid, the stress is not linear, but tends to be monotonically increasing with strain. This applies especially to brittle materials, which store a lot of elastic energy before a fracturing suddenly. Ductile materials have a broad plastic region, where deformation processes occur long before a fracture. These are irreversible because energy is dissipated. Upon a purely elastic deformation, the solid can always be brought back to its initial state with no energy loss.

Previous molecular dynamics studies of tensile stress with atoms interacting through the simple Lennard-Jones potential report a linear stress-strain relationship in the entire elastic region [3]. A newer study using the same potential, but an huge number of atoms (500 million) gives more complicated results due to the formations of dislocation patterns [4]. On the extreme side of large-scale MD computations, we find stress corrosion cracking simulations with 200 billion atoms simulated on 200 thousand CPUs [5]. Large-scale simulations with coupling to continuum mechanics are popular because the length scales become closer to the macroscopically observed ones. However, some studies are also devoted to the microscopic structure and behaviour of fracturing materials of more complex and realistic potentials, like crack propagation in silicon [6].

2.2 Material interfaces and friction

Friction between materials is a phenomenon which have been observed and measured for hundreds of years, but which has not yet been fully understood from basic principles. The problem is that it is quite a complex process, and different questions 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 independently by DaVinci, Amontons and Coulomb. 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:

$$F = \mu L. \tag{2.6}$$

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 phenomena that appears when sliding two rough materials against each other are fractures and slips near the material interface [7].

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 an entire surface. As surfaces can have a roughness on all scales, only very few asperities make contact to counteract the load L (see Fig. 2.2). Ringlein and Robbins [8] provide a refined friction formula based on their research on friction,

$$F = \mu L + cA_C. \tag{2.7}$$

They also point out the reason why Eq. (2.6) often works well for macroscopic materials. The real area of contact A_C is often proportional to the load L, causing one to believe that there is only a linear load dependence.

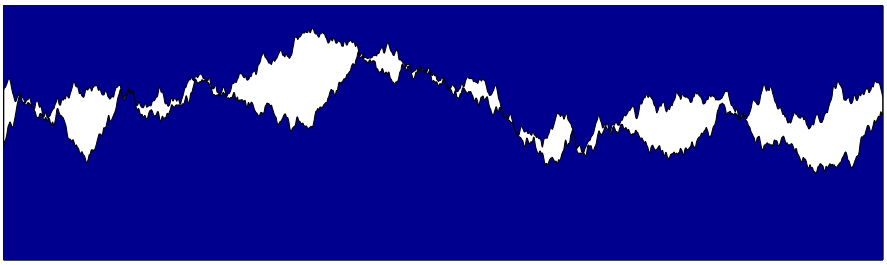


Figure 2.2: How the interface between apparently flat material surfaces can look like on a small scale. Modelled in Matlab using smoothed random walks.

Digging deeper into the origins of friction requires understanding the forces between microscopic asperities, which I attempt to model in this thesis. 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 in material interfaces. The distribution of stress between two spheres and between a sphere and a plane 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.

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 approximated 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 [9]. The method I use is based on this idea (see section 5.2.4).

Forces between asperities have been investigated by measuring the stress between a sphere and a plane surface or between two spheres composed of generic atoms [10, 9]. The stress distributions are heavily dependent on how the atoms inside the spheres are arranged. Results are shown in Fig. 2.3. Moving on to bigger scales, the paper [11] investigates the effect of pushing a rough and a smooth surface together using both molecular dynamics and the finite element method. The contact area between the surfaces varies linearly with the applied load, while the interfacial separation is reduced in a logarithmic manner. This can be seen as results averaged over thousands of asperities.

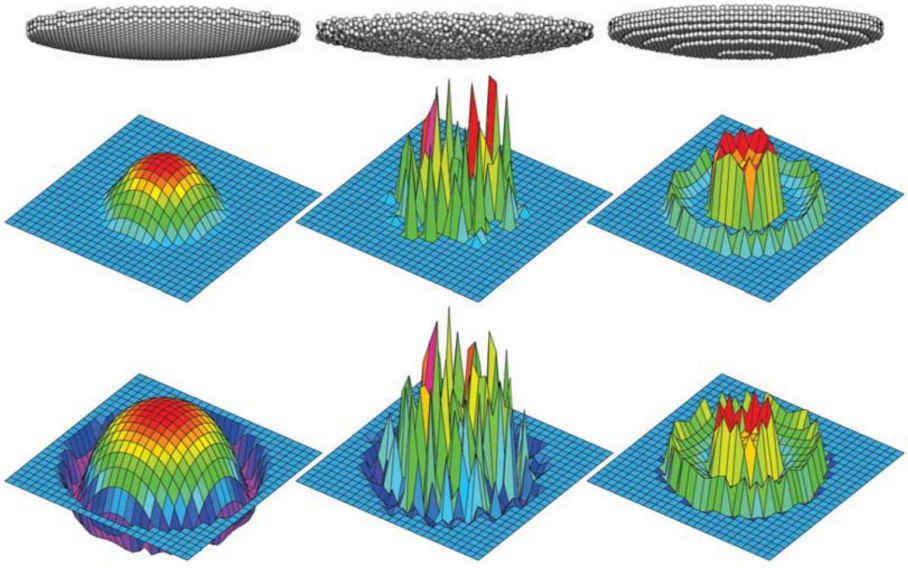


Figure 2.3: Stress distributions in the interface between plane surfaces and sphere segments of bent, random and stepped organization. The first row is with purely repulsive interactions while the second row also includes adhesive interactions. Borrowed from Luan and Robbins [10].

Friction simulations with realistic potentials are very rare. The paper [7] de-

scribes molecular dynamics studies of the processes occurring in the interface of large, atomically smooth blocks in relative motion. This includes stick-slip motion, fractures and bulging of surfaces. Such smooth surfaces have also been studied by experiment, with the fascinating technique of friction force microscopy. A microscopic tip of a hard material, such as tungsten or diamond is slid across a plane surface, and a pattern with the lattice constant a as periodicity emerges in the measured friction force [1]. The large-scale mechanisms of friction between rough surfaces, which cause F_{friction} to vary linearly with the normal force, are explained quite well in [8].

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 again affected by more fundamental properties, as the number of elementary particles of each type inside the atoms. The simulations will always be homogenous with respect to material types. Interfaces between two different types of materials are outside the scope of this thesis. Some details characterizing the two materials under study are given below.

2.3.1 Sodium chloride

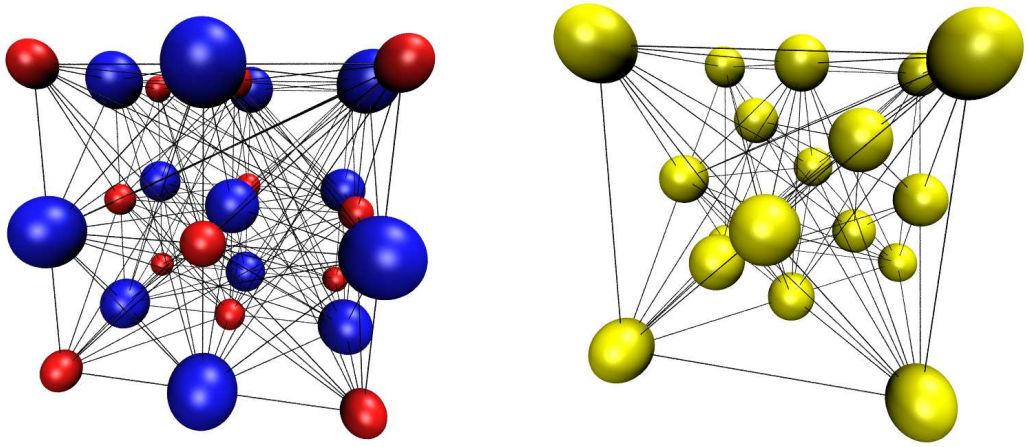
Sodium chloride is a solid composed of Na^+ and Cl^- ions forming an ionic lattice. The mean mass of sodium atoms and chlorine atoms are 22.9898 amu and 35.4525 amu, respectively [12]. These numbers are achieved by averaging over the different isotopes of the chemical elements, weighted by their relative occurrence in the universe. Only stable isotopes of chlorine are considered. I neglect the small effect of an abundant or deficient electron on the ion masses. Sodium chloride has a normal 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 parallel to the Cartesian axes will be charge neutral (visible in Fig. 5.6). The lattice constant has the value $a = 5.63$ Å [13]. The conventional unit cell consists of four atoms of each type. One filled unit cell of a NaCl crystal is shown in Fig. 2.4(a).

Sodium chloride is a ductile (soft) material. Rather than breaking violently, like glass, salts will exert only small amounts energy to its surroundings upon fracture. In most cases, atoms will float freely and rearrange slowly to when deformations are induced.

2.3.2 Silicon

Silicon is a semi-metal and has covalent bonds between atoms. The mean mass of a silicon atom is 28.0854 amu [12]. Its solid phase density is 2329 kg/m^3, corresponding to a particle density of 0.050 Å^{-3}.



(a) Na^+ and Cl^- ions in a double fcc cell.

(b) Si atoms in a diamond cell.

Figure 2.4: Conventional unit cells for the two crystalline materials.

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 [13]. Figure 2.4(b) shows one filled unit cell of a diamond crystal consisting of silicon atoms.

The angular dependence in the electronic density around silicon atoms 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. It is a brittle material, which means that a fracture will exert a lot of stored-up elastic energy, while there are no significant plastic deformations before the fracture itself. This behaviour can in some ways be seen as the opposite of that of a salt.

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 exerted 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 nucleus which is also consists of more elementary particles. This is, even when viewed classically, a complicated system. And we know that quantum mechanics must be applied to systems of this small size. All elementary particles are quantum fields with an associated wave function and a quantized energy. In this project, I study systems containing thousands of atoms. *Ab initio* calculations, containing only strictly controlled approximations, a minimum of empirical 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 an 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 wave functions, with e.g. density functional theory (DFT). This enables calculation of forces on nuclei, and the movement of nuclei can be simulated classically. 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 describes the motion of the atoms, but the potential between the atoms tries to take into account their electronic structure (more of this in chapter 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 homogeneous 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 are of great interest. Properties of phase transitions and pair correlation functions can be used to learn much about 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.

3.1 Equations of motion

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

$$m_i \frac{\mathrm{d}^2 \mathbf{r}_i}{\mathrm{d}t^2} = \mathbf{F}_i, \tag{3.1}$$

where m_i is the mass of atom i, $\mathbf{F}_i$ is the force vector exerted on atom i by other atoms 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 atom with index i is the negative gradient of the potential energy of the entire system, U_p. In the usual case of interaction pairs, this equals the sum of the negative gradient of interaction potentials U_{ij} from other atoms j,

$$\mathbf{F}_i = -\nabla_i U_\mathrm{p}(\mathbf{r}_i) = -\sum_{j \neq i} \nabla_i U_{ij}(\mathbf{r}_{ij}). \tag{3.2}$$

The meaning of ∇_i is the gradient with respect to the position of atom 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 atoms. In a system of N atoms, this means $\frac{1}{2}N(N-1)$ force terms must be calculated when a straightforward approach is used.

When put together, Eqs. (3.1) and (3.2) form $3N$ time-dependent, coupled, nonlinear partial differential equations. The nonlinearities arise from the forms of the potential energy function, unless we deal with e.g. an ideal (non-interacting) gas of atoms in a homogeneous gravitational field. Because of the nonlinearities, even the exact solutions of the equations will display sensitivity to initial conditions. Perturbations will then grow exponentially with time [14]. I'll come back to this point briefly in section 3.2.

3.1.1 Time integration

In most cases, to integrate Newton's second law, Eq. (3.1), molecular dynamics programs use *symplectic integrators* like the Verlet and Leapfrog methods [15]. 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 $\mathbf{r}(t)$, velocities $\mathbf{v}(t)$ and forces $\mathbf{F}(t)$, the values of

these quantities at the next time step is calculated this way:

$$\mathbf{v}(t + \Delta t/2) = \mathbf{v}(t) + \frac{\mathbf{F}(t)}{2m}\Delta t, \tag{3.3}$$

$$\mathbf{r}(t + \Delta t) = \mathbf{r}(t) + \mathbf{v}(t + \Delta t/2)\Delta t, \tag{3.4}$$

$$\mathbf{F}(t + \Delta t) = -\nabla U_{\mathrm{p}}(\mathbf{r}(t + \Delta t)), \tag{3.5}$$

$$\mathbf{v}(t + \Delta t) = \mathbf{v}(t + \Delta t/2) + \frac{\mathbf{F}(t + \Delta t)}{2m}\Delta t. \tag{3.6}$$

The atom index i is dropped for brevity. The method can be derived from Newton's second law, or directly from Hamiltons principle of least action using a forward Euler discretization [14].

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 (1 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. The Verlet method is of order 4 in position and order 2 in velocity. This means that the errors are proportional to Δt^4 and Δt^2, respectively. Since the energy is a function of both position and velocity, the largest error term in the energy will be proportional to Δt^2.

Energy accuracy

The total energy of a system is the sum of kinetic and potential energy, most generally written as

$$U(\{\mathbf{r}_i\}, \{\mathbf{p}_i\}) = \frac{1}{2}\sum_{i=1}^{N}\frac{\mathbf{p}_i^2}{m_i} + U_{\mathrm{p}}(\{\mathbf{r}_i\}), \tag{3.7}$$

where $\mathbf{p}_i = m_i\mathbf{v}_i$ is the momentum of particle i. This is, in the case of a time-independent potential, equivalent to the Hamiltonian of the system.

In a simulation of 1728 NaCl ions with a time step $\Delta t = 2.0$, the standard deviation in total energy is $\sigma_U = 4.9 \cdot 10^{-6}$ eV. For comparison, the standard deviation in potential energy is $\sigma_{U_{\mathrm{p}}} 1.1 \cdot 10^{-3}$ eV when the ions are in equilibrium in their crystal structure. Figure 3.1 shows how the energy varies over the course of the simulation. The deviations are the biggest when conversions between kinetic and potential energy happen, that is, when the derivatives of these energies are great in absolute value. Increasing the time step gives an increase in the error over time, but not as drastic as the order 2 property suggests. With $\Delta t = 4.0$ fs, $\sigma_U = 7.2 \cdot 10^{-6}$ eV and with $\Delta t = 8.0$ fs, $\sigma_U = 19.1 \cdot 10^{-6}$ eV.

No dynamical equation integration scheme conserves energy exactly. It is possible to create such a scheme, but it might come at the price of momentum exactness. One possibility for ad hoc velocity estimation is to demand that lost or gained potential energy is converted directly to kinetic energy. This gives the following

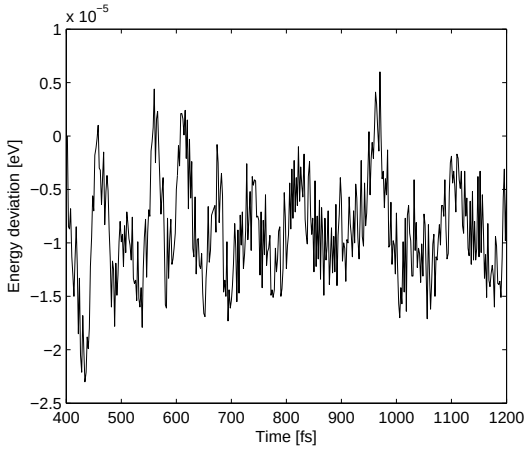


Figure 3.1: Deviation in energy from the starting value for 1728 NaCl ions simulated over 400 time steps.

explicit formula for the velocity:

$$\mathbf{v}(t+\Delta t) = \mathbf{v}(t) + \left[\sqrt{(\mathbf{v}(t)\cdot\mathbf{n})^2 + \frac{2}{m}\left(U(t) - U(t+\Delta t)\right)} - (\mathbf{v}(t)\cdot\mathbf{n})\right]\mathbf{n}. \quad (3.8)$$

The important thing with regard to energy is that the magnitude of the new velocity is correct. The unit vector $\mathbf{n}$ points in the direction of velocity change, which can be determined by force evaluation. Although this velocity estimation results in an exact conservation of energy, it should not be used when updating the position, because it would introduce a low-order approximation. Either way, this velocity estimator is not used in my calculations due to its troublesome nonlinearity.

3.2 Thermodynamics

Thermodynamics is the science of energy balance in systems of many degrees of freedom. Its essence is captured in four laws, which can be formulated in the following way:

0. *If two thermodynamic systems are each in thermal equilibrium with a third, then they are in thermal equilibrium with each other.* Two objects will exchange energy until they are in thermal equilibrium with each other. This is equivalent to having the same temperature. For objects A, B and C with temperatures T_A, T_B and T_C, the zeroth law says that $T_A = T_C$ if both $T_A = T_B$ and $T_B = T_C$. This may seem self-explanatory, but physicists saw it necessary to state the property as a fundamental law defined by the abstract concept of equilibrium.

1. *In any process in an isolated system, the total energy remains the same.* The law of energy conservation is the most fundamental in all of physics, and it is yet to be disproven. In thermodynamics, it is often stated as $\Delta U = Q + W$:

The change in energy U equals the energy input from heat Q and work W. I use this relation several times in the thesis, and always operate with quantities which are positive when the energy of the system is increased.

2. *Any process in an isolated system has a tendency to increase the entropy of that system.* This law does not exclude a decrease in entropy, but this is so improbable that it will only happen in systems of a couple of atoms. The law of large numbers is responsible for this. In a bigger system, there are many more composite states of high entropy than low entropy, so a high entropy is most probable. As a result, matter will diffuse and energy will spread out across the available degrees of freedom.

3. *As temperature approaches absolute zero, the entropy of a system approaches a constant minimum.* The same is true for the heat capacity, $C_V = T\frac{\partial S}{\partial T}$, where S is the entropy. I do not utilize the third law in this thesis, but it has important consequences in the field of low-energy quantum mechanics (e.g. for Bose-Einstein condensates).

Some words are needed to clarify what is meant by *thermodynamic equilibrium*. There are in principle three types of thermodynamic processes: heat exchange, compression/expansion, and particle exchange. These have their associated pairs of conjugate thermodynamic variables and types of equilibria, listed in Tbl. 3.1. The processes will occur between two systems until they are in the equilibrium specific for the process. This does not imply that any of the systems are totally static. Particles continue to move, and heat is still exchanged between the two systems, but in equal amounts both ways. Total equilibrium in a single system can be loosely defined by all the variables in Tbl. 3.1 being constant on average. Section 3.3 describes how equilibrium can be achieved in a simulation.

Process	Equilibrium	Extensive variable	Intensive variable
Heat exchange	Thermal	Entropy S	Temperature T
Compression/expansion	Mechanical	Volume V	Pressure P
Particle exchange	Diffusive	Particle number N	Chemical potential μ

Table 3.1: Thermodynamical processes and equilibria [16].

Table 3.1 leaves some loose ends in the form of intensive and extensive variables. Extensive variables have a linear dependence on amounts of matter, whereas intensive variables stay constant in this respect. For example, if you double the system size, the number of particles will of course double, but the chemical potential stays constant. The product of a pair of conjugate variables (one extensive and one intensive) equals the energy transferred in the associated process. For example, the work done in transferring N particles is μN. Furthermore, the table suggests trivial definitions for the intensive variables. Temperature is the tendency to transfer heat and loose entropy, pressure is the tendency to do mechanical work and increase volume, and chemical potential is the tendency to transfer, and thereby loose, particles. Note that a higher volume with the same pressure means a lower energy. This causes the minus sign in Eq. (3.15).

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,

$$\langle A \rangle = \int \rho(\{\mathbf{r}_i\}, \{\mathbf{p}_i\}) A \mathrm{d}\omega, \tag{3.9}$$

$$\bar{A} = \lim_{t\to\infty} \frac{1}{t} \int_0^t A \mathrm{d}t, \tag{3.10}$$

where $\mathrm{d}\omega$ is an infinitesimal phase space element. Since the distribution of the phase space variables will be equal in the ensemble and in time, these averages must be equal, $\langle A \rangle = \bar{A}$. 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 infeasible.

Due to nonlinearities in the dynamical equations, two nearly identical configurations in phase space will separate exponentially with time and involve into entirely different trajectories. For systems in equilibrium, this is no big issue, as we are interested in averages over atoms and time. Averages are not influenced by small fluctuations in the microscopic phase space variables. The importance is greater in systems far from equilibrium. Small fluctuations can e.g. make a solid fracture at an earlier or later point in time. Especially in the small system sizes I consider, simulations with apparently negligible changes in parameters will produce different results. This limits the accuracy of the numerical calculations and sometimes forces me to average over several simulation runs.

3.2.1 Ensembles

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. (The variables in the parenthesis are kept constant.) Every system state with total energy E will contribute equally to computed averages, so the phase space density is

$$\rho(\{\mathbf{r}_i\}, \{\mathbf{p}_i\}) \propto \delta(E - U(\{\mathbf{r}_i\}, \{\mathbf{p}_i\})), \tag{3.11}$$

with $\delta(x)$ being the Dirac delta function.

In macroscopic experiments, it is generally easier to keep the temperature constant instead 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*. In this case, the phase space density is proportional to the Boltzmann factor of a configuration:

$$\rho(\{\mathbf{r}_i\},\{\mathbf{p}_i\}) \propto \exp\left(-U(\{\mathbf{r}_i\},\{\mathbf{p}_i\})/k_{\mathrm{B}}T\right). \tag{3.12}$$

There is also a possibility of simulating the pressure canonical (NPT) ensemble by keeping the pressure constant and changing the volume by so-called *barostats*.

The fourth important ensemble, the grand canonical (μVT) ensemble is not common in MD simulations. However, this is useful when performing multi-scale simulations with interfaces between atomic and continuous regimes, and would require keeping the chemical potential constant and varying the number of particles. Finding the triple points of phase boundaries efficiently also requires this ensemble to be simulated. The (μPT) ensemble does not exist, because the variables μ, P and T together contains redundant information, while we are missing other pieces of information which can only be determined by keeping at least one extensive variable constant [14].

In computing most of the results in this thesis, the canonical ensemble is used. Thus the temperature is an important quantity which must be estimated. According to the *equipartition principle*, every degree of freedom which is quadratic in position or speed will contribute with an average energy of

$$E_{\mathrm{dof}} = \frac{1}{2}k_B T \tag{3.13}$$

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 $U_k = \frac{1}{2}\sum_{i=1}^{N} m_i v_i^2 = \frac{3}{2}Nk_BT$, we get

$$T = \frac{1}{3Nk_B}\sum_{i=1}^{N} m_i v_i^2 \tag{3.14}$$

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, derived from an expanded first law of thermodynamics,

$$\mathrm{d}U = T\mathrm{d}S - P\mathrm{d}V + \mu\mathrm{d}N. \tag{3.15}$$

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. For each Cartesian direction, it corresponds to a normal distribution with 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.

3.2.2 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. (3.14) to be

$$\alpha = \sqrt{\frac{3Nk_B T_{\text{target}}}{\sum_{i=1}^{N} m_i v_i^2}}. \tag{3.16}$$

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 [15]. The Andersen thermostat replaces the velocities of random particles with velocities from the Maxwell-Boltzmann distribution. The Berendsen and Nosé-Hoover thermostats add a fictitious 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 strikingly 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 BDP thermostat

The thermostat I use during dynamical processes is one recently developed by Bussi, Donadio and Parrinello [17] (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 thermostat is

$$\mathrm{d}U_{\text{k}} = \frac{U_{\text{k}}^{\text{target}} - U_{\text{k}}}{\tau}\mathrm{d}t + 2\sqrt{\frac{U_{\text{k}} U_{\text{k}}^{\text{target}}}{3N\tau}}\mathrm{d}W. \tag{3.17}$$

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:

$$\alpha^2 = D + \frac{T_{\text{target}}}{3NT}(1-D)\sum_{i=1}^{3N} R_i^2 + 2\sqrt{\frac{T_{\text{target}}}{3NT}D(1-D)}R_1, \tag{3.18}$$

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 $\mathbf{v}_i \rightarrow \alpha\mathbf{v}_i$ for all dynamical 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 3.2 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 negligible in comparison to the calculation of forces between particles.

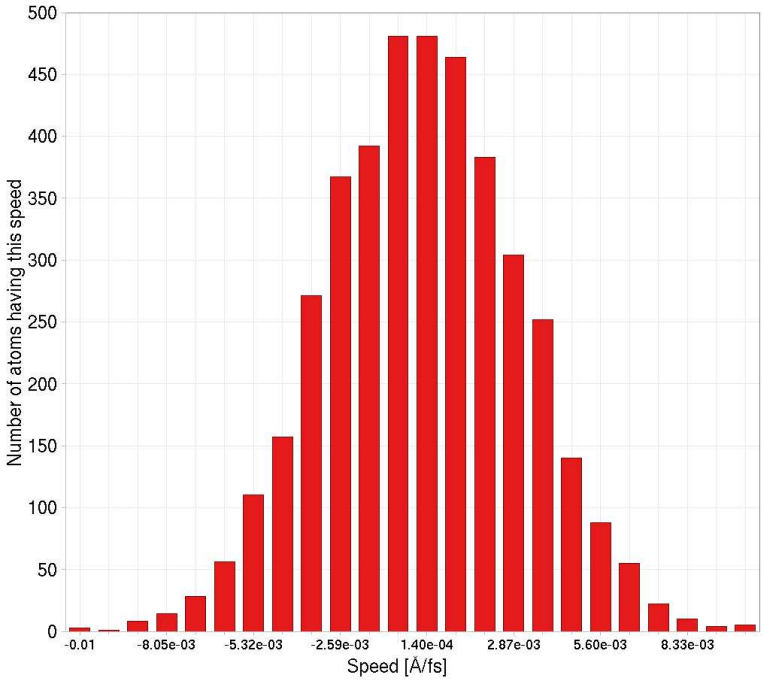


Figure 3.2: The distribution of one velocity component for 4096 silicon atoms in a simulation using the BDP thermostat.

For one velocity rescaling, the energy absorbed from the heat bath is $U_{\text{k}}(\alpha^2 - 1)$, where U_{k} is the total kinetic energy of the system. Assuming that the velocities are rescaled every time step, the power exerted on the system by the heat bath is

$$P = \frac{1}{\Delta t}U_{\text{k}}(\alpha^2 - 1). \tag{3.19}$$

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 equilibration epoch. An example of a method to equilibrate this system is the following phases:

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

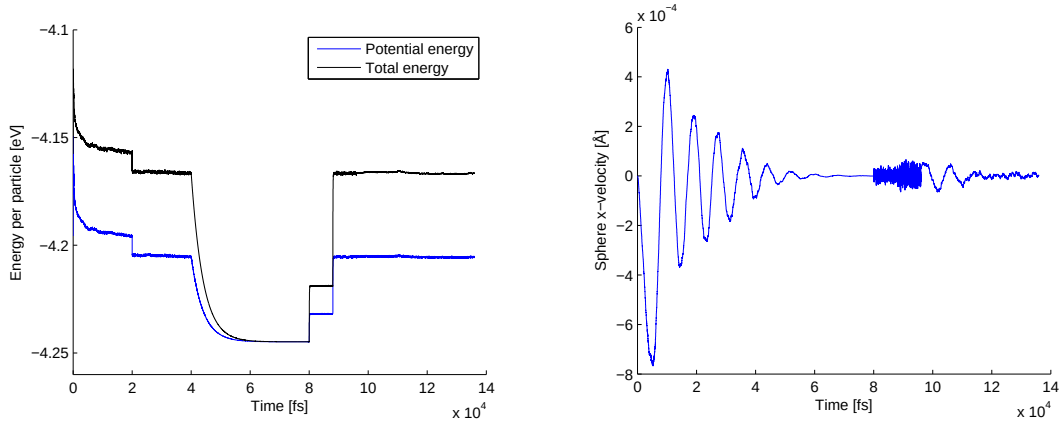
2. Energy dissipation phase, adding a force term $-\gamma \mathbf{v}_i$ to all particles to simulate viscosity. The friction parameter is chosen so that $\gamma \Delta t / m_i \ll 1$. To avoid requiring a "hard" implementation of this in the equation of motion integration, the velocity of the previous 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 just 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 abruptly increased from 50-100 K to 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 5.1.3). Some collective vibrations will reappear in the sphere, but these are naturally occurring phonons, and have an amplitude which is comparable to the thermal fluctuations of independent particle positions.

Figure 3.3 shows the energy and center of mass velocity in the x direction during a typical silicon surface contact 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



(a) (b)

Figure 3.3: Time development of energy and collective velocity of a big Si sphere pushed towards an Si surface during all simulation phases.

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 a 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 wave function description of a particle, Ehrenfest's theorem says that [18]

$$\frac{\mathrm{d}\langle\hat{p}\rangle}{\mathrm{d}t} = \langle -\nabla U_\mathrm{p} \rangle \,. \tag{3.20}$$

This is equivalent to Newton's second law for average dynamical quantities. Of course, atoms are small enough that averages do 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 approximation 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 are also taken into account. This is done in subtle ways, as small amounts of energy transferred into or out of the system (read more in chapter 5). 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 using effective interaction potentials for 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. Common practices are to use quantum mechanical energy calculations or measurements from scattering experiments to construct the effective potentials. Approximately correct atom dynamics are recovered even though the atoms are treated as classical point particles. The effective potentials usually contain parameters which are determined by trial and error. The parameters are chosen such that numerical simulations best reproduce the experimentally observed behaviour for a specific system.

The interaction potential between atoms are determined by functions which have a continuous dependence only on the positions of these atoms. The first molecular dynamics simulations used so-called hard-sphere potentials, which are described mathematically by step functions. These require special algorithms with a non-constant time step. Although they had educating purposes in their time, such potentials are now seldom in use.

The total potential energy in a system is normally the sum of interaction potentials between all pairs of atoms. In the case of periodic boundary conditions (see section 5.1.1), this sum also goes over infinitely many copies of the atoms in the system:

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

Here U_{ij} is the two-atom interaction potential and $r_{ij} = \sqrt{\mathbf{r}_{ij} \cdot \mathbf{r}_{ij}}$ is the distance between atoms i and j. $\mathbf{R}$ represents all translation vectors to copies of the system ($\mathbf{R} = n_x L_x \mathbf{e}_x + n_y L_y \mathbf{e}_y + n_z L_z \mathbf{e}_z$ for $n_x, n_y, n_z \in \mathbb{Z}$). Atoms do not interact with themselves, so the $i = j$ term is skipped if $\mathbf{R} = 0$.

The force exerted on atom i by atom 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. Below, I present this and other potentials I use in my simulations, and give some details regarding how they are computed efficiently.

4.1 The Lennard-Jones potential

One of the simplest and most successful 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 electronic wave functions 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 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 can be fired at atomic nuclei to find their 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 phases of these waves have changed. The total phase change will tell how strong 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 a potential depending on interatomic distance.

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. It was the first continuous potential used in a MD simulation, by Rahman [19], who studied correlations in liquid argon. It has become a standard to use this potential for modelling generic short-range repulsion between 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 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 LJ parameters for interactions between pairs of equal atoms. These are found by varying them and comparing the resulting simulation data

Atom	ϵ_{ii} [eV]	σ_{ii} [Å]	Source
Ar	$1.0318 \cdot 10^{-2}$	3.405	[19]
Na^+	$5.6478 \cdot 10^{-3}$	2.35	[20]
K^+	$4.3501 \cdot 10^{-3}$	3.33	[21]
Cl^-	$4.3501 \cdot 10^{-3}$	4.40	[20]

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

with experimental data. The parameters for atoms of two different elements are calculated by two combination rules. Given parameters for interactions between atoms of the same element, homogeneous interactions, geometric and arithmetic means give reasonable values for the parameters of heterogeneous interactions [22]. The combination rules are

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

Ions are very close to being noble gases and having a spherically symmetric electronic density. Therefore the LJ potential is used to model the interactions between Na^+ and Cl^- ions in this project. The short-range part of the Coulomb interaction is strong enough that the attractive force of the LJ interaction is almost negligible in the dynamics. The most important role of the LJ potential is stopping equally charged particles from crashing into each other.

4.1.1 Implementation details

The simplest form of real space cutoff possible with periodic boundary conditions (see section 5.1.1) is the minimum image convention. Define x_{ij} as the distance in one direction between atoms i and j. With PBC, the distance is calculated as the smallest number of x_{ij}, $x_{ij}+L_x$ and $x_{ij}-L_x$. In words, the atoms are only interacting with the nearest of their copies. This is used in my program, as LJ interactions do not exceed half the system size in any direction.

Neighbour lists, or Verlet lists, are lists of all atoms that are closer to each other than a specified cutoff length. In my implementation, all atoms i have their own lists of atoms $j < i$ that are close to them. The force evaluations happen only between neighbouring pairs of atoms, and the lists are updated every 10th time step or so. The cutoff length must be set so that short range potentials like the LJ potential are negligible beyond it. I normally set the cutoff for this potential to $r_c = 12$ Å. The maximum number of neighbours is set to three times the expected number based on the average particle density ρ,

$$\text{max neighbours} = 3 \cdot \frac{4\pi}{3} r_c^3 \rho. \tag{4.6}$$

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 as mentioned, only the very short-range part of the LJ potential will make an important contribution to the dynamics.

4.2 The Coulomb potential

Electrically charged atoms, ions, interact through the Coulomb potential in addition to short-range quantum mechanically derived potentials like the LJ potential. The mathematical form of the Coulomb potential is exact, if the quantization of electromagnetic radiation is neglected. A more critical approximation is that of describing the abundant or deficient electron as a point charge. This is done in order to avoid quantum mechanical calculations and to complete the calculations in a reasonable amount of time. Thus ions are modelled as point particles with an associated unit elementary charge, $q = \pm e$. Still working classically, it is also possible to use partial charges for ions and atoms to model inhomogeneous concentrations of electrons in a discrete manner (see e.g. [23]).

In reduced units (see section 6.1), the Coulomb potential between two particles with charge q_i and q_j is

$$U_{ij}(r_{ij}) = \frac{q_i q_j}{r_{ij}}. \tag{4.7}$$

Regardless of how simple it may look, this very important potential causes problems with PBC (section 5.1.1) because of its long ranged nature. To see this, it is useful to write down the total electrostatic energy due to the Coulomb interactions,

$$U = \sum_{\mathbf{R}} \sum_{i=1}^{N} \sum_{j=1}^{i} \frac{q_i q_j}{\|\mathbf{r}_{ij} + \mathbf{R}\|}. \tag{4.8}$$

The comments to the more general Eq. (4.1) also apply here.

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 ion on a line being positively charged and the rest negatively charged, the interaction energy of one ion is $2q_1^2 \ln 2$. 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.

In all cases with periodic boundary conditions in one or more directions, the electrostatic energy of an ionic system can not be computed using a real space cutoff approach. Many methods have been devised for solving this problem. In this project, I use the simplest and most famous of these, the particle-particle Ewald summation technique. More advanced methods use multipole expansions or map the charge density to a mesh and solve Poisson's equation to obtain electrostatic forces [15].

4.2.1 3D Ewald summation

The Ewald summation technique is used to efficiently capture the long range interaction properties of the Coulomb interaction with a finite number of summation terms [15]. It can also be applied to other long range interactions, like the van der Waals potential.

First, we need to express the pair potential in Eq. (4.7) as a Fourier transform:

$$U_{ij}(\|\mathbf{r}_{ij} + \mathbf{R}\|) = \frac{q_i q_j}{(2\pi)^3} \int \frac{4\pi}{k'^2} e^{i\mathbf{k}'\cdot(\mathbf{r}_{ij}+\mathbf{R})} \mathrm{d}^3\mathbf{k}'. \tag{4.9}$$

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

$$\sum_{\mathbf{R}} e^{i\mathbf{k}'\cdot\mathbf{R}} = \frac{(2\pi)^3}{V}\sum_{\mathbf{k}} \delta^3(\mathbf{k}'-\mathbf{k}). \tag{4.10}$$

The $\mathbf{k}$-vectors are reciprocal to the translation vectors $\mathbf{R}$ for the entire system. V is the volume of the entire system. Using the identities in Eqs. (4.9) and (4.10) in Eq. (4.8), we will get rid of the $\mathbf{R}$-vector sum and replace it with a sum over $\mathbf{k}$-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:

$$U_{\mathrm{p}} = \frac{1}{V}\sum_{\mathbf{k}\neq 0}\sum_{i=1}^{N}\sum_{j=1}^{i} q_i q_j \frac{e^{i\mathbf{k}\cdot\mathbf{r}_{ij}}}{k^2}. \tag{4.11}$$

The particle index i is always used as an index and should not be confused with the imaginary unit i.

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

$$\rho_i(\mathbf{r}) = q_i\left(\frac{\alpha^2}{\pi}\right)^{3/2} e^{-\alpha^2|\mathbf{r}-\mathbf{r}_i|^2}. \tag{4.12}$$

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. (4.9). By doing this, the total energy will be given by three summation terms:

$$U_{\mathrm{p}} = \frac{2\pi}{V}\sum_{\mathbf{k}\neq 0}\frac{1}{k^2}\left|S(\mathbf{k})\right|^2 e^{-\frac{k^2}{4\alpha^2}} + \sum_{i=1}^{N}\sum_{j=1}^{i-1} q_i q_j \frac{\operatorname{erfc}(\alpha r_{ij})}{r_{ij}} - \frac{\alpha}{\sqrt{\pi}}\sum_{i=1}^{N} q_i^2. \tag{4.13}$$

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 an ion and itself, and is constant.

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

$$S(\mathbf{k}) = \sum_{j=1} q_j e^{i\mathbf{k}\cdot\mathbf{r}_j} \tag{4.14}$$

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

$$\operatorname{erf}(x) = \frac{2}{\pi}\int_0^x e^{-t^2}\mathrm{d}t. \tag{4.15}$$

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 vice versa for a small α [24].

The total electrostatic energy in Eq. (4.13) contains not only the distance between ions, but the actual position vectors. 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 ion i:

$$\begin{aligned} \mathbf{F}_i = &- \frac{4\pi q_i q_j}{V} \sum_{\mathbf{k}\neq 0} \frac{1}{k^2} \mathrm{Im}\{e^{-i\mathbf{k}\cdot\mathbf{r}_i} S(\mathbf{k})\} e^{-\frac{k^2}{4\alpha^2}} \mathbf{k} \\ &+ q_i \sum_{i\neq j}^{N} q_j \left(\frac{2\alpha}{\sqrt{\pi}} \frac{e^{-\alpha^2 r_{ij}^2}}{r_{ij}^2} + \frac{\mathrm{erfc}(\alpha r_{ij})}{r_{ij}^2} \right) \mathbf{r}_{ij}, \end{aligned} \tag{4.16}$$

where $\mathrm{Im}\{z\}$ is the imaginary part of a complex number z.

4.2.2 2D Ewald summation

In some cases, two-dimensional PBC have to be used even if the simulated system is three-dimensional. This actually makes the Ewald summation more difficult and time-consuming. I have looked at two methods, Kawata and Mikamis (KM method) [25] and the PHL method, presented in the same paper.

For the reciprocal space term of $\mathbf{k} = 0$, which for the 2D-PBC 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 analogue 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 Coulomb potential should have been used. The last term is the same self-interaction correction term as in the 3D case.

$$U^L_{\mathbf{k}\neq 0} = \frac{1}{L_y L_z} \sum_{\mathbf{k}\neq 0} \int_{-\infty}^{\infty} \frac{1}{k^2 + h^2} e^{-\frac{1}{4\alpha^2}\left(k^2+h^2\right)} \left|S(\mathbf{k}, h)\right|^2 \mathrm{d}h, \tag{4.17}$$

$$U^L_{\mathbf{k}=0} = -\frac{1}{L_y L_z} \sum_{i=1}^{N} \sum_{j=1}^{N} q_i q_j \left[\frac{\sqrt{\pi}}{\alpha} e^{-\alpha^2 x_{ij}^2} + \pi x_{ij} \mathrm{erf}(\alpha x_{ij}) \right], \tag{4.18}$$

$$U^L_{\mathrm{const}} = -\frac{\alpha}{\sqrt{\pi}} \sum_{i=1}^{N} q_i^2. \tag{4.19}$$

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

$$S(\mathbf{k}, h) = \sum_{j=1}^{N} q_j e^{i(\mathbf{k}\cdot\mathbf{r}_j + hx_j)}. \tag{4.20}$$

The trick of the KM method is that a double sum over ions is replaced by integration over this structure factor. The force on ion 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

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

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

$$\left(\mathbf{F}^L_{\mathbf{k}\neq 0,i}\right)_x = -\frac{2}{L_y L_z} \sum_{\mathbf{k}\neq 0} \int_{-\infty}^{\infty} \frac{h}{k^2+h^2} e^{-\frac{1}{4\alpha^2}\left(k^2+h^2\right)} \mathrm{Im}\{e^{-i(\mathbf{k}\cdot\mathbf{r}_i + hx_i)} S(\mathbf{k}, h)\} \mathrm{d}h, \tag{4.22}$$

$$\left(\mathbf{F}^L_{\mathbf{k}=0,i}\right)_x = \frac{2\pi q_i}{L_y L_z} \sum_{j=1}^{N} q_j \mathrm{erf}(\alpha x_{ij}). \tag{4.23}$$

4.2.3 Implementation details

The most demanding matter in this project, both mathematically and computationally, is the Ewald summation method for the Coulomb interaction. The short ranged parts of the sums, which are the least computationally expensive, are carried out with the same neighbour lists as with the LJ potential. The cutoff length r_c from Eq. (4.26) is used. This length is forced lower than half the system size, for compatibility with the minimum image convention.

The long ranged parts are summed over $\mathbf{k}$-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 $k_x^2 + k_y^2 + k_z^2 \leq k_{\mathrm{cutoff}}$ In the 2D case, k_x is simply put to zero. There will be roughly thousands of vectors in the sum for 3D periodicity, and hundreds for the 2D periodicity. The vectors are found and stored at program start, together with the terms in the summation formulas depending only on the vectors.

In both the 2D and 3D periodicity cases, the problem of calculating forces is divided into two parts. First, the structure factor in Eq. (4.14) or Eq. (4.20) is found. Only the particles which are repeated periodically are included in the structure factor sums. In some cases, the simulation will contain particles without PBC. These interact amongst each other using a direct Coulomb term, $U_{ij} = q_i q_j / r_{ij}$. The structure factor is updated whenever particles move between time steps, thus not for stationary fixed particles. An speed-optimizing improvement is also implemented for the case of a constant displacement of all the fixed particles. In this case, which

occurs during many of my applied deformations, the structure factor is multiplied with a single complex number for each value of $\mathbf{k}$ and h. For example, if 2D PBC is applied to all fixed particles, and they are moved a distance Δx in the x direction, the numbers to multiply with is $\exp(ih\Delta x)$. The structure factor is correctly updated with no new summations over particles required.

After the structure factor of the periodic interaction is found, the resulting forces on dynamical ions and total potential energy can be calculated. This takes somewhat shorter time than the structure factor calculation. In the 3D energy and force sums, the symmetry operation $\mathbf{k} \to -\mathbf{k}$ does not change the inner expressions. 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 half-sphere), a double sum with $k_x = 0$ (a semi-circle) and $k_y > 0$ and a single sum with $k_x = 0$, $k_y = 0$ and $k_z > 0$ (a line).

The integrals in the 2D $\mathbf{k} \neq 0$ sums are approximated by using Simpson's composite rule. By minimal recommendations in the paper [25], the interval for h is $[-1.5k_c, 1.5k_c]$, and the number of evaluation points is 65. The more sophisticated Gaussian quadrature would work very well here. Hermite polynomials use a Gaussian weighting function, which corresponds well to the $\exp(h^2/4\alpha^2)$ term in the energy and force sums.

A much more efficient way to model repeating two-dimensional patterns would be to keep 3D-PBC and insert lengths of vacuum in the x direction directly outside the system. The slabs would then lie in a layerwise fashion, interacting only slightly with each other. This introduces complications when applying deformations to some of my systems, so I have not investigated this method, although it has been used in several cases of numerical surface research. It is definitely a matter worth pursuing if bigger ionic lattice simulations are needed.

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. Complex number arithmetic is performed with a custom class, including basic algebraic operations, and exponential and trigonometric ones.

The required parameters for the Ewald summation technique are calculated internally in the same way as in the Protomol MD package [24]:

$$\alpha = C\left(\frac{N}{V^2}\right)^{1/6}, \tag{4.24}$$

$$r_c = \frac{\sqrt{-\ln\epsilon}}{\alpha}, \tag{4.25}$$

$$k_c = 2\alpha\sqrt{-\ln\epsilon}. \tag{4.26}$$

In this way we change from three to two input parameters and gain an $O(N^{3/2})$ scaling. C is a constant adjusting the ratio between the time usage of the real-space and reciprocal-space sums, and is is most cases 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 included 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 runs over all particles, giving it a $N^{3/2}$ scaling.

There are many ways of optimizing the Ewald summation method, some general and others specific to systems. The methods I have mentioned which is not implemented in my code would indeed increase the performance, but I choose to limit the time and effort used to attend this matter, as this project is more oriented towards physics than numerical methods.

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 an atomic nucleus. Higher energy states, starting with the so-called $2p$ states, includes a spherical harmonic function, which is not spherically symmetric. Thus the electron distributions in 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 that do not fulfill the chemical 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 potential energy sum

$$U_\mathrm{p} = \sum_{b=1}^{N} U^{(b)} \tag{4.27}$$

where $U^{(b)}$ is the b-body potential. Each term is a sum of all b-body energies 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, which is constructed especially for silicon [26]. This potential reproduces the cohesive energy and melting point of bulk 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 (sums over system replicae are dropped for convenience):

$$U_\mathrm{p} = \sum_{i=1}^{N} U_i^{(1)} + \sum_{i=1}^{N}\sum_{j=1}^{i-1} U_{ij}^{(2)} + \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{j-1} U_{ijk}^{(3)}. \tag{4.28}$$

In the last term, the sum does not include the terms where $j = i$ or $k = i$. The double counting of index j occurs because atom 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 the 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 natural cutoff of the potential, which encourages the use of neighbour lists.

The two-body part of the potential has the form

$$U_{ij}^{(2)} = \begin{cases} A\left(\frac{B}{r_{ij}^4} - 1\right)\exp\left(\frac{\sigma}{r_{ij}-r_c}\right) & \text{for} \quad r_{ij} < r_c, \\ 0 & \text{for} \quad r_{ij} \geq r_c. \end{cases} \tag{4.29}$$

All derivatives of the potential are continuous at $r_{ij} = r_c$. The potential acts repulsively at short distances and attractively at slightly larger distances, just like the LJ potential.

The SW three-body potential has the form

$$U_{ijk}^{(3)} = \begin{cases} \lambda \exp\left(\frac{\gamma}{r_{ij}-r_c} + \frac{\gamma}{r_{ik}-r_c}\right)\left(C_{ijk} + \frac{1}{3}\right)^2 & \text{for} \quad r_{ij}, r_{ik} < r_c, \\ 0 & \text{otherwise.} \end{cases} \tag{4.30}$$

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 $\mathbf{r}_{ik}$:

$$C_{ijk} = \cos\theta_{jik} = \frac{\mathbf{r}_{ij} \cdot \mathbf{r}_{ik}}{r_{ij} r_{ik}}. \tag{4.31}$$

The $\frac{1}{3}$ in Eq. (4.30) 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 [26]. 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 [Eqs. (4.29) and (4.30)].

4.3.1 Forces

The two-body force between atoms i and j can be derived straightforwardly from Eq. (4.29), producing the expression

$$\mathbf{F}_{ij}^{(2)} = \frac{A}{r_{ij}}\left[\frac{4B}{r_{ij}^5} + \left(\frac{B}{r_{ij}^4} - 1\right)\frac{\sigma}{(r_{ij}-r_c)^2}\right]\exp\left(\frac{\sigma}{r_{ij}-r_c}\right)\mathbf{r}_{ij}. \tag{4.32}$$

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 atoms: $(\nabla_i + \nabla_j + \nabla_k)U_{ijk}^{(3)} = 0$

[27]. Therefore, the most complicated term, $\nabla_i U_{ijk}^{(3)}$, does not have to be calculated. The forces arising from $U_{ijk}^{(3)}$ are

$$\begin{aligned}\mathbf{F}_i^{(3)} &= (\nabla_j + \nabla_k)U_{ijk}^{(3)},\\ \mathbf{F}_j^{(3)} &= -\nabla_j U_{ijk}^{(3)},\\ \mathbf{F}_k^{(3)} &= -\nabla_k U_{ijk}^{(3)}.\end{aligned} \tag{4.33}$$

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

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

$$\nabla(\mathbf{A}\cdot\mathbf{B}) = \sum_{ij}\left(A_j\frac{\partial^2 B_j}{\partial x_i^2} + B_j\frac{\partial^2 A_j}{\partial x_i^2}\right)\mathbf{e}_i, \tag{4.34}$$

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:

$$\nabla_j C_{ijk} = \frac{1}{r_{ij}}\left(\frac{C_{ijk}\mathbf{r}_{ij}}{r_{ij}} - \frac{\mathbf{r}_{ik}}{r_{ik}}\right) \tag{4.35}$$

With Eq. (4.35), the required potential derivative can be calculated, and we finally obtain

$$\begin{aligned}\nabla_j U_{ijk}^{(3)} = \lambda\exp\left(\frac{\gamma}{r_{ij}-r_c} + \frac{\gamma}{r_{ik}-r_c}\right)\left(C_{ijk}+\frac{1}{3}\right)\\ \cdot\left[\left(C_{ijk}+\frac{1}{3}\right)\frac{\gamma}{(r_{ij}-r_c)^2}\frac{\mathbf{r}_{ij}}{r_{ij}} + \frac{2}{r_{ij}}\left(\frac{C_{ijk}\mathbf{r}_{ij}}{r_{ij}} - \frac{\mathbf{r}_{ik}}{r_{ik}}\right)\right],\end{aligned} \tag{4.36}$$

which can be inserted into Eq. (4.33) 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 normally set to 110% of this. Each atom must now keep track of the neighbours with both lower and higher indices than their own. The atom checking for neighbours is always the middle one, with the first index in Eqs. (4.30) and (4.33). The third index will always be smaller than the second one, as when

calculating forces between pairs of particles. This way, all bonds between dynamical atoms are accounted for exactly once.

Sometimes, an open ends boundary condition with fixed particles on the sides are necessary (see section 5.1). Calculation of three-body forces between dynamical and fixed particles requires some extra thought. Four types of three body fixed particle terms (TBFPT) can occur, 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 dynamical particles 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.

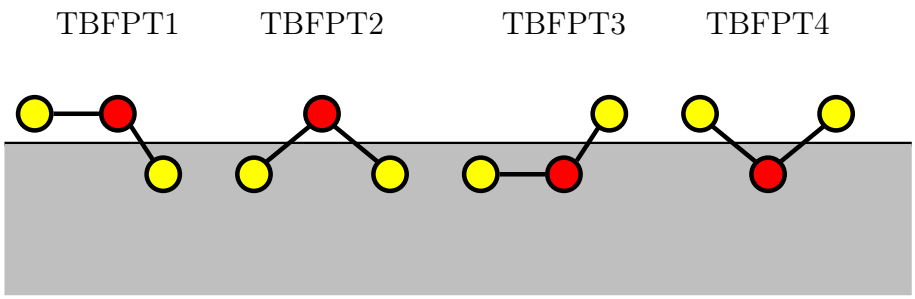


Figure 4.1: The four different three-body fixed particle terms in force and potential calculations. Particles in the gray area are fixed.

4.4 Comparison 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 Coulomb potentials are both spherically symmetric, in the sense that only the distance between the interacting pair of atoms affects the potential energy. The Coulomb potential has different signs when equally and oppositely charged particles interact. The range of the Coulomb 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 shows radial plots of the different two-body potentials presented. 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 is seen (-1.819 eV) while the LJ potential well is too small to be seen at all ($-\epsilon_{ii} = -5.6478 \cdot 10^{-6}$ eV).

This chapter concludes the theory part of the thesis. The physical and mathematical foundation is now laid, and what remains is to model the systems on a computer.

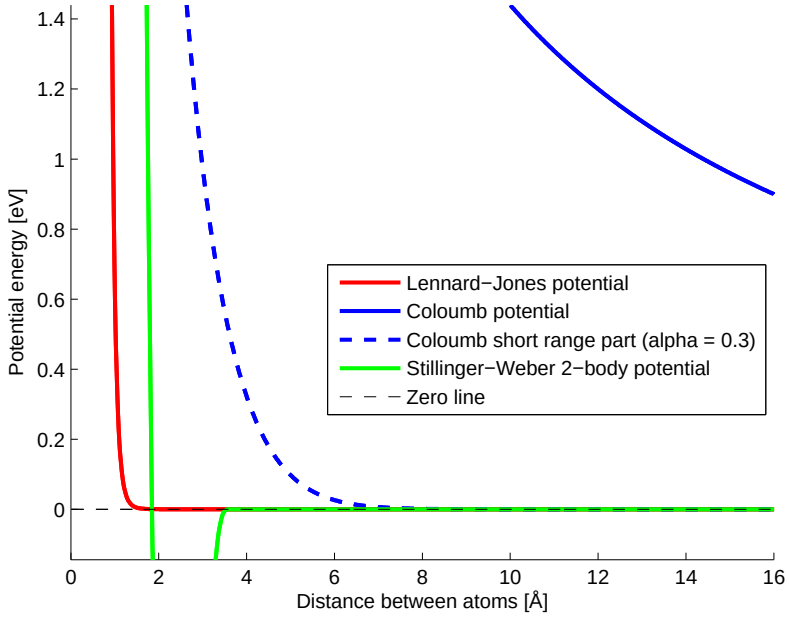


Figure 4.2: Comparison of interaction potentials as a function of the distance between two sodium ions and two silicon atoms.

This enables me to predict the response of a system to an applied deformation, which is the objective in this project. Bits of the numerical methods have already been presented, as it was natural to explain how the potentials are computed within this chapter.

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 computationally efficient methods and methods with a stronger physical foundation. 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 accurately predicting the mechanical strength of a macroscopic piece of material.

The systems I consider are atoms initialized to an energetically preferable 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. 2.4). This creates a cuboid lattice, which can be chopped, stretched, cut to a sphere, or otherwise deformed. 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 than they would be for e.g. a uniform speed distribution. The total linear momentum of the system is set to zero on simulation start, to prevent drift when periodic boundary conditions are applied. No corresponding correction is done for the rotational momentum, because periodic boundary conditions assure this to be zero.

5.1 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. What I am mainly interested in is the interactions between the atoms inside the box, but the effect of the surrounding atoms is also important. This is incorporated approximately through different boundary conditions.

5.1.1 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 $\mathbf{R} = n_x L_x \mathbf{e}_x + n_y L_y \mathbf{e}_y + n_z L_z \mathbf{e}_z$, for integers n_x, n_y and n_z. There will not be any surfaces, only an infinite amount of bulk particles. PBC in non-cuboidal geometries is also possible, as long as the shape of the region can be stacked periodically to fill space. Irregular prisms and the truncated octahedron are usual examples.

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 is taking place, but PBC is still used in this case in the directions parallel 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 simulations. The dynamical atoms will behave like atoms in the middle of a macroscopic piece of material.

In the program, all particles have flags which tell in which directions they experience PBC. This is useful in surface contact experiments, where a group of particles with no PBC is in contact with one or two plane surfaces. The surface particles, either if fixed or dynamical, will always have PBC in the directions tangent 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, particles will not fall of the edge of such a plane surface, but continue to move as if the surface continued to infinity in its PBC-directions.

5.1.2 Fixed particles

The simplest kind of boundary condition is open ends, that is, nothing is done to prevent particles from escaping. In most of the experiments in this project, I use this boundary condition in one direction in tandem with fixed particle forces. A perfect lattice of fixed particles is 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 dynamical particles are in a crystalline state, the difference between the dynamics with PBC and open ends with fixed particles in one direction is almost negligible.

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 5.1 shows the two-dimensional analogue of the setup schematically.

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

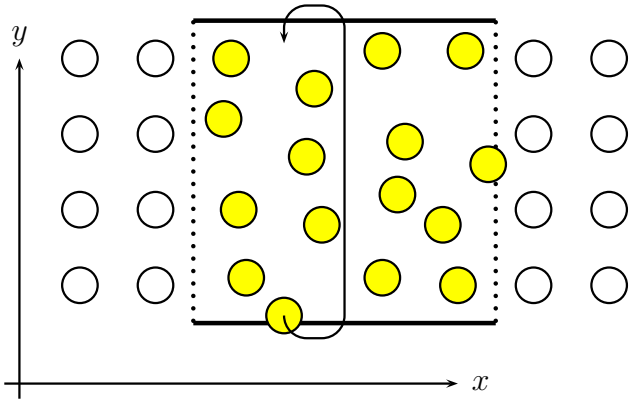


Figure 5.1: The boundary conditions for a system with fixed particles on the x-sides and PBC in the y direction. The thermal motion is greatly exaggerated.

been attempts, in e.g. [28]. 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 monolayer 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 6 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 7.1). For this test, layers of fixed particles were put on both x-sides, and PBC was applied in the y-z-plane.

When the lattice is cut to a sphere, the positive x direction layers of fixed particles are removed and replaced by vacuum. In both the sphere and sphere segment systems, only fixed particles are experiencing PBC. For NaCl, this gives an electrostatic structure factor of very low magnitude, because of the translational symmetry of the lattice. Thus the long-ranged Coulomb forces will have little say for the dynamics.

5.1.3 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 dynamical atoms. There must be a thermal coupling between fixed and dynamical atoms, though the fixed atoms can be said to have zero heat conductance because they do not gain any kinetic energy. An artificial thermal coupling to the outside can be accomplished by performing a temperature rescaling on only the dynamical atoms near the box sides. For this purpose, I use the BDP thermostat introduced in section 3.2.2.

To avoid an unreasonably sudden cutoff, the relaxation time τ_i of the thermostat

is chosen differently for each atom. An atom directly on the side wall will have a base relaxation time τ read in as a parameter. Diverging at the cutoff point, the relaxation time decays from this point towards the edge 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. (3.17)] multiplicatively as $1/\tau$. So the energy coupling to the outside heat bath decreases linearly inwards from the box sides.

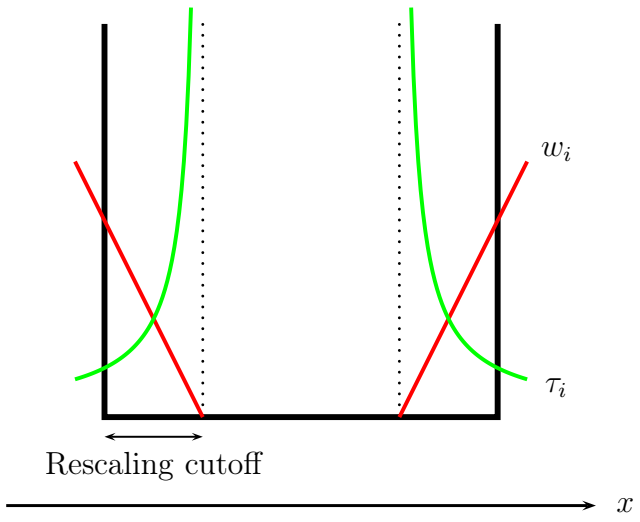


Figure 5.2: The variation of the temperature measurement weight w_i and thermostat relaxation time τ_i for an atom i at different x-positions in the simulation box.

Since the relaxation time varies with the distance from the sides, how temperature is measured before a rescaling must be changed. Fourier's (macroscopic) 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 (3.17) 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

$$T_{\text{side}} = \frac{1}{3k_{\text{B}}} \sum_{i=1}^{N_{\text{incl}}} w_i m_i (\mathbf{v}_i - \mathbf{v}_{\text{coll}})^2 \quad \text{with} \quad \sum_{i=1}^{N_{\text{incl}}} w_i = 1, \tag{5.1}$$

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. This is not equally important in global temperature

estimation, as the average motion of all particles often cancels out. Figure 5.2 shows the values of 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.

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. This will induce a mode I fracture.

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 dynamical particles at the edges in place. The distances between dynamical 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. Applying a constant stress is also possible, by using a thermodynamical ensemble similar to the pressure canonical ensemble [3]. Figure 5.3 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.

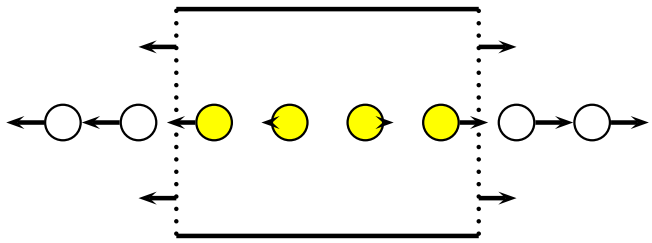


Figure 5.3: Displacements caused by tensile strain.

Let us consider stress and strain in 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 tensile strain ϵ is the easiest to calculate:

$$\epsilon = \frac{\Delta L_x}{L_x}. \tag{5.2}$$

Before calculating the tensile stress σ, the energy balance between two consecutive time steps t and $t + \Delta t$ must be examined. I assume that the only processes

changing the system energy are heat conduction and applied stress. For each time step, the energy transferred 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

$$U(t + \Delta t) - U(t) = \Delta U_T + \Delta U_\sigma. \tag{5.3}$$

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 derivative $F_\sigma \approx \frac{\Delta U_\sigma}{\Delta L_x}$, where the minus sign is discarded because the force from the dynamical 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

$$\sigma = \frac{\Delta U_\sigma}{L_y L_z \Delta L_x}. \tag{5.4}$$

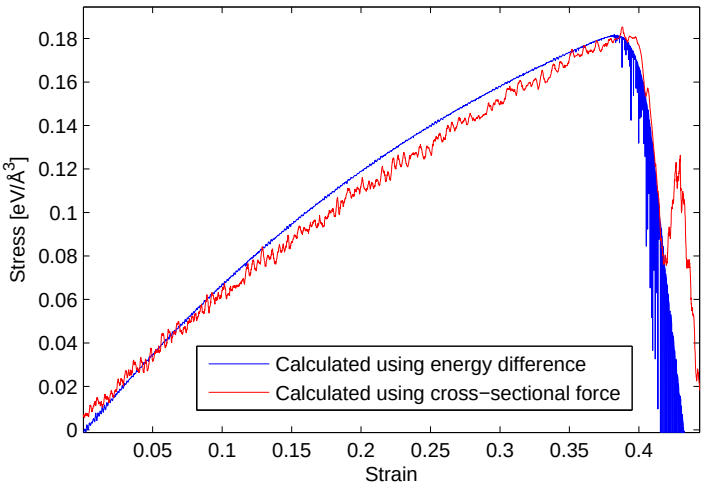


Figure 5.4: Comparison between the two tested methods of stress calculation, showing a good correspondence for a silicon crystal.

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. Long-ranged Coulomb forces are too troublesome to be included in these calculations, so the cross-sectional force is estimated by only including short-range forces. The cross-sectional stress is

$$\sigma_{\text{cs}} = \frac{F_{\text{cs}}}{L_y L_z}. \tag{5.5}$$

The stresses σ and σ_{cs} are compared for a 4096-Si-atom system undergoing a fracture in Fig. 5.4. The energy balance-based 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}. The methods give the same maximal stress, though the slope is slightly different over the whole curve. The results after the fracture are irrelevant. The two stress graphs for a corresponding NaCl simulation are similar to each other, but have much larger deviations, both positive and negative. These are presumably only due to the exclusion of long-ranged forces in σ_{cs}.

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 comparison with ordinary bulk matter, giving a reduced cross-sectional force. Additionally, this method of measurement is too complicated when long-ranged Coulomb forces are included.

The time of fracture can be pinpointed by examining when a few particles start to move rapidly in area that is about to fail. The recent motion quantity (section 6.4) is ideal for this. Generally, the time of fracture coincides with the time of highest achieved stress. The value of this stress is used as the breaking stress σ_f when testing Griffith's relation [Eq. (2.1)].

5.2.2 Shear stress

A shear deformation exerts inhomogeneous stress on different parts of a material, as when scissors cut through paper. With fixed particles outside the x direction edges, I move these particles in opposite directions in the y direction, as shown in Fig. 5.5. This alone will move the dynamical 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 is constant. A plastic deformation must eventually occur, in the form of a mode II fracture.

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

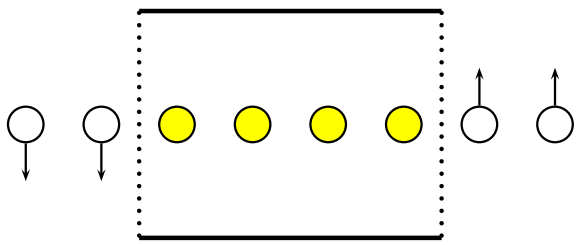


Figure 5.5: Displacements caused by shear strain.

When applying shear stress in the y direction, the atom layers will become skewed. Shear strain γ is defined by the tangent of the skew angle, which here

can be measured visually as

$$\gamma = \frac{\Delta y}{L_x}, \tag{5.6}$$

where Δy is the relative displacement of the fixed atoms from the left and right sides. The length of the simulation box in the x direction L_x enters the equation because this is the direction where the applied stress is inhomogeneous. Be aware that the x and y axes have switched places from Fig. 2.1.

Shear stress τ can be calculated in a way similar to tensile stress, by dividing the force required to apply the deformation by the area normal to this force. Again, the force is approximated by a discretized derivative of the energy, this time with respect to the distance the left and right sides have moved relative to each other, Δy. We get the expression

$$\tau = \frac{\Delta U_\tau}{L_x L_z \Delta y}, \tag{5.7}$$

with ΔU_τ calculated exactly as ΔU_σ in Eq. (5.3).

5.2.3 Bulk material flaws

In fracture experiments, an interesting question is how a flaw (crack) inside the material influences the breaking stress. Theoretical considerations were presented in section 2.1. Such a flaw is modelled as a void (vacuum area where atoms are removed) shaped as a sphere or cylindrical disk. As the simulations contain discrete atoms, the flaw must have a thickness larger than the interactomic distance. The position of the flaw is not important due to translational symmetry (PBC), so it is 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 radius of the void.

5.2.4 Contact, adhesion and friction

In numerical experiments regarding surface contact, I look at the interaction between a sphere or sphere segment of particles and a plane surface of the same type of particles. The attraction between such surfaces with the same atomic species is called cohesion, but I will stick to using the more general word adhesion. The spherical form is meant to model an asperity on a rough material surface, such as depicted in Fig. 2.2. It corresponds to the stepped crystal surface in a similar model by Luan and Robbins, seen in [10] and Fig. 2.3. The plane surface consists of some atomic layers of fixed particles, as mentioned in section 5.1.2. Additional layers of dynamical particles can be added for the surface not to be completely rigid. The surface particles have PBC in the directions parallel 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 homogeneous gravitational field, although it is set 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 another surface of the same material.

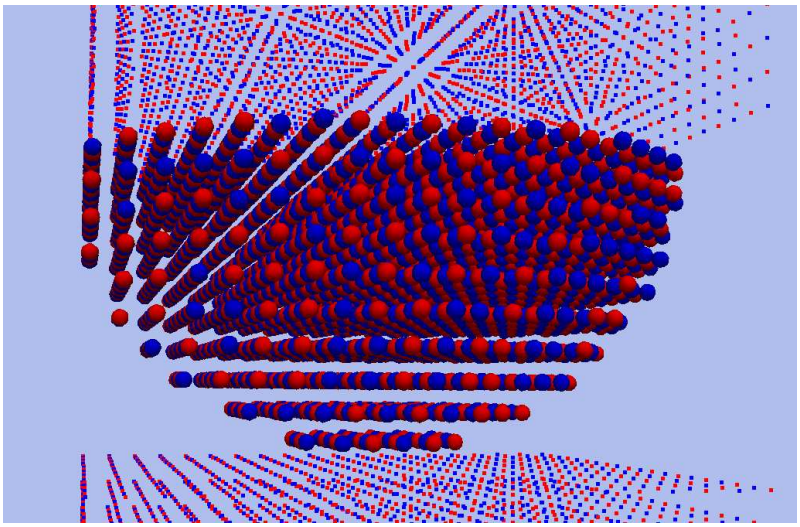


Figure 5.6: A microscopic sphere segment (here a half-sphere) and surface of NaCl. The initial positions in a contact force simulation.

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 the simulation 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:

$$\mathbf{F}_i = \frac{m_i}{\sum_j m_j}\mathbf{F}, \tag{5.8}$$

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 middle 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 therefore 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 stop after a given time for measurements to be performed. The applied force is difficult to control when this approach is used. Though, the contact forces and stress can be measured, giving results very similar to those obtained with the sphere method.

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

$$\bar{U}_k = \frac{1}{2}m\bar{v^2} = \frac{3}{2}k_\mathrm{B}T \quad \Rightarrow \quad v_\mathrm{RMS} \equiv \sqrt{\bar{v^2}} = \sqrt{\frac{3k_\mathrm{B}T}{m}} \tag{5.9}$$

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 effectively quasistatic. The pushing of the upper layer will not create unwanted elastic waves or other dynamics which is not already happening in equilibrium due to thermal motion. The method I use for the indentation is the same as the expansion (and compression) method introduced in section 5.2.1, so dynamical particles will also be slightly moved. I have tried turning this effect on and off with no noticeable difference in dynamics, energies or forces.

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 x components of forces on sphere particles are summed up and divided with the area of the bin to find the stress,

$$\sigma_b = \frac{N_b}{L_y L_z} \sum_{i \in b} \sum_{j \in \text{surface}} F_{ij}^x, \tag{5.10}$$

where N_b is the number of bins and $L_y L_z$ is the area of the plane 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, $\sigma(r)$, but the results were too unstable. Long range Ewald forces between dynamical 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 are studied using the sphere segment method. The materials are brought in close contact and then separated. Some particles stick to the lower surface of fixed particles, and some 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 5.2.1):

$$\Delta U_{\text{push}} = U(t + \Delta t) - U(t) - \Delta U_T. \tag{5.11}$$

The energy gained by pushing the upper fixed layer of particles will be used to pull particles out of potential fields, to deform the crystal structure, and to induce movement which is transferred out of the system as heat.

Contact area

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

$$A_C = \frac{U}{\Sigma}. \tag{5.12}$$

The constant Σ is material dependent and must be measured in a computational experiment. In order to to this, I simulated bulk crystals with PBC in the y and z directions and walls of fixed atoms 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 as the calculated potential energy between dynamical atoms and lower fixed atoms, divided by the area $L_y L_z$.

My results are $\Sigma = -0.2471$ eV/Å^2 for silicon and $\Sigma = -0.02522$ eV/Å^2 for sodium chloride. The NaCl calculations were done with four layers of unit cells with fixed atoms 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 atoms.

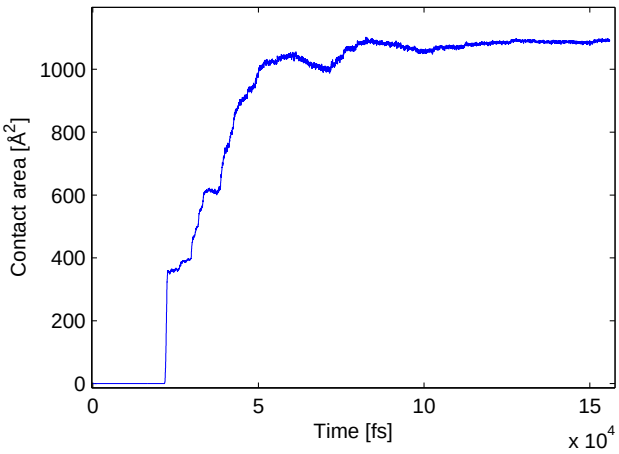


Figure 5.7: The time evolution of the area of contact for a Si sphere segment upon indentation and separation (see Fig. 8.8).

When pushing a sphere and a surface of equal materials together, the A_C vs t graph (Fig. 5.7) is continuous 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 this kind of experiments, a contact area measurement will have a low precision and is only used to extract qualitative information about trends for different cases.

Friction

Using the sphere method, 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 traditionally measured for two surfaces with 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

$$F_{\text{friction}} = \frac{\Delta U_{\text{drive}}}{\Delta r_{\text{COM}}} = \frac{\Delta U_{\text{drive}}}{v_{\text{coll}} \Delta t}. \tag{5.13}$$

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

Two infinite surfaces where one has a single asperity is a better setup for a friction experiment. Therefore I have also investigated the friction phenomenon using the sphere segment method. An input applied load is necessary, which is kept by moving the upper layer of fixed particles up and down until the contact force reaches a desired value. By abstraction, this is equivalent to the problem of solving the equation $F(x) = L$, where F is the current contact force, x is the relative position of the surfaces of fixed atoms and L is the desired load. There are many well-developed ways of solving such an equation, but nearly all of them involves calculating the derivative $\frac{\mathrm{d}F}{\mathrm{d}x}$. In my case, this would have to be a discretized derivative, given the computational complexity of storing the x-derivative of all forces between the sphere and the bottom layers of fixed atoms. I emphasize simpler methods not requiring such derivatives.

I have tested some methods of obtaining a desired load, all with one parameter β which has to be adjusted. One alternative is to simply move the fixed particles with a speed $\pm\beta$, upwards if the load is too high and downwards if it is too low. This gives a displacement

$$\Delta x = \pm \beta \Delta t. \tag{5.14}$$

This method is reminiscent of step methods in equation solvers for many variables. Preferably, a time damping term is included to prevent the fixed layers from moving constantly back and forth, creating elastic waves which propagate through the sphere segment. Such a damping can also be introduced by bringing the difference $F - L$ into the equation. The obvious choice is to let the speed vary linearly with this force difference, creating a displacement

$$\Delta x = (F(x) - L)\beta \Delta t. \tag{5.15}$$

This method has problems with stability, and will also induce oscillations in Δx because it creates weak elastic waves in the sphere segment. When a small β is used, it is still the preferred method.

5.2.5 Relative rotations

The numerical experiments described so far are all performed with materials aligned in a *commensurate* manner. This simply means that the atomic layers of the surfaces in contact are oriented equally with respect to each other, a condition rarely satisfied in real life. It is non-trivial to investigate the forces between incommensurate surfaces in MD simulations. In my cuboidal systems, the PBC directions always coincide with the three Cartesian directions. When rotating layers of fixed atoms, the Cartesian translational symmetry of the simulation box will in general be broken. In order to avoid this, all four corners of the intersection between the simulation box

and the plane of rotation must correspond to the same point in the conventional unit cell of the crystal structure. That is, for a set of Pythagorean triples $a, b, c \in \mathbb{N}$, the angle of rotation θ must obey

$$\cos\theta = a/c, \quad \sin\theta = b/c. \tag{5.16}$$

This condition even requires the intersection between the simulation box and the plane of rotation to be quadratic, with the number c determining the original number of unit cells in each Cartesian direction. A visual example for the lowest Pythagorean triples is given in Fig. 5.8.

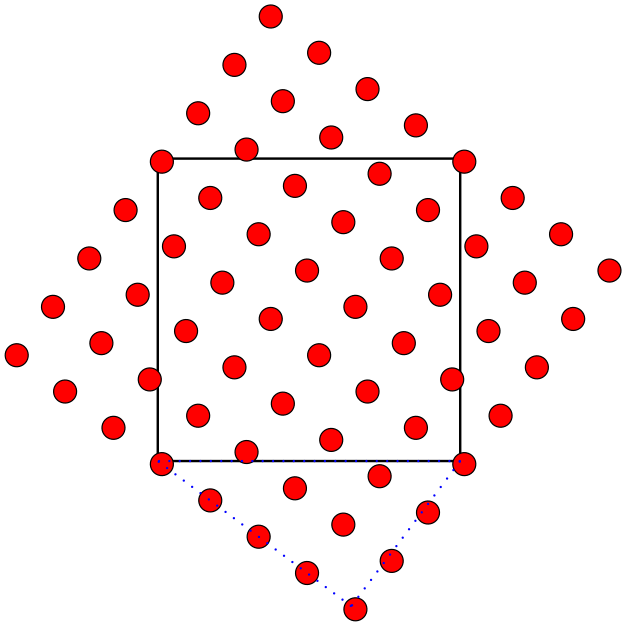


Figure 5.8: A simple cubic crystal plane rotated with an angle $\theta = 36.87°$. The dotted triangle is Pythagorean with $a = 4$, $b = 3$, $c = 5$. Correct translational symmetry is achieved if only the particles inside the box are included in a simulation.

The system in Fig. 5.8 is somewhat troublesome to create, owing to the required extension and cutting of the lattice. Furthermore, the rotation angle is special because of the joint repetitiveness of the lattices, which is required for translational symmetry. The surfaces are not as incommensurate as they could have been. I have therefore not gone through analysis of the systems in section 8.2 with a partial rotation. Instead, I have concentrated on the simpler system of a rotated sphere interacting with a plane surface. The interface force as a function of rotation angle gives interesting patterns, as seen in section 8.1.2.

With this and the previous chapters, I have elaborated on how all quantities in my simulations are calculated. Perhaps more importantly, I have explained how real-world systems are approximated and represented with discrete mathematics. The

most important approximations presented in this chapter are the ones eliminating the degrees of freedom of objects in the vicinity of the system. Other atoms are responsible for compressing and stretching the system and for conducting heat in and out. In addition, PBC neglects that atoms in system replicae have different thermal fluctuations in position and velocity, and can contribute in a slightly different way to fracture processes than atoms inside the system. But including millions of atoms which independently have very small effects on the system is computationally infeasible. Therefore these degrees of freedom are incorporated into the simple boundary conditions and applied deformations presented in this chapter. This is an attempt to capture the behaviour of the surroundings of the system in a computationally cheap fashion.

Chapter 6

Program details

My MD simulation program is written in C++, contains 10 classes and a total of about 5700 source code lines. 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.

Below, I will elaborate on some methods which are not part of the main numerical calculations. The mathematics and discretization of continuous functions is presented in earlier chapters, and the implementation of this consists mostly of `for`-loops including elementary mathematical operations. Appendix A contains descriptions of the classes and functions of the program. The adjustable input parameters defining the setup for a simulation are also listed there.

6.1 Physical units

The physical units used for input and output, external units, are chosen so that the magnitude of the values will be close to, and usually 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 for using a different set of units when calculations are done is that equations will be simpler and require fewer multiplications with constants. Also, it is possible to extract multiple data sets 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 have also decided to put the most important constant in force calculations, the Coulomb force constant, to 1. The charge of particles will be measured in elementary charges, $e = 1.602176487 \cdot 10^{-19}$C. The Coulomb force law will give rise

to a constraint in the following way:

$$F = k_e \frac{q_1 q_2}{r^2} \quad \Rightarrow \quad F_0 \bar{F} = k_e \frac{e^2 \bar{q}_1 \bar{q}_2}{r_0^2 \bar{r}^2}, \tag{6.1}$$

$$\bar{F} = \frac{\bar{q}_1 \bar{q}_2}{\bar{r}^2} \quad \text{and} \quad \frac{k_e e^2}{r_0^2 F_0} = \frac{k_e e^2 t_0^2}{m_0 r_0^3} = 1. \tag{6.2}$$

This determines the mass conversion factor, m_0. Other conversion factors are combinations of r_0, t_0, e and m_0. Table 6.1 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 6.1: Physical units for I/O and calculations in the program.

6.2 Random numbers

Randomly distributed numbers are required for generating velocity values from the canonical distribution and using 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* [29], 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,

$$n_1 = Su, \qquad n_2 = Sv, \qquad S = \sqrt{\frac{-\ln s}{s}}. \tag{6.3}$$

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.

6.3 Parallelization

As my program requires a considerable amount of CPU time for the systems I am simulating, parallel computations with several processor cores 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. Quantities which are calculated by double (triple) sums over particles are the most important ones to parallelize. 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 parallelized is the computation of forces. Especially with the very intensive Ewald summations, 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 $\mathbf{F}_{ij}$ to calculate. The exception is the last node, which gets "what is left". Figure 6.1 shows this way of splitting the workload schematically. The same procedure is used in calculating the total potential energy of the system.

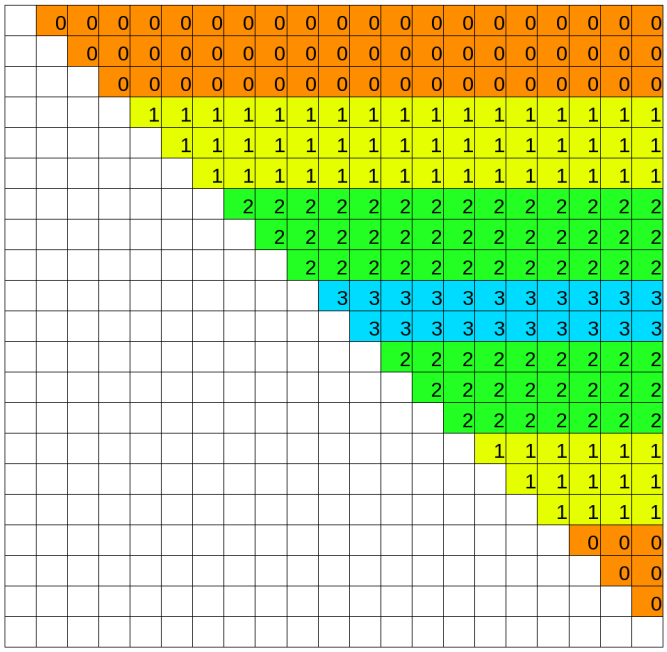


Figure 6.1: Matrix showing how the calculation of forces between 20 particles (matrix elements $\mathbf{F}_{ij}$) is divided amongst four nodes. The numbers specify the node numbers, or "ranks".

The BDP thermostat requires the sum of many squared random numbers. These numbers are found distributively, with different seeds for the random generators. The calculation of the random numbers for the initial velocities are not parallelized, but drawn from random generators with the same seed, producing the same numbers for all nodes.

Everything that is not parallelized, 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 heterogeneous

clusters, this could cause slight deviations between the nodes induced by unequal round-off errors. Output is done only from the master node (of rank 0).

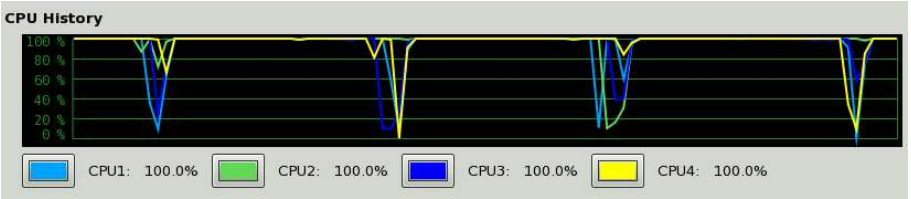


Figure 6.2: CPU usage for the simulation of a big silicon system on four CPU cores.

For small systems, parallelization will require too much time for communication between nodes in comparison with the actual computations. The force calculations must therefore be quite extensive for parallelization 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. Multiplying the calculation time by four gives a rough estimate of the time which would be used by a single node. One time step would be completed in about $4 \cdot 20 = 80$ seconds, while four nodes uses 25 seconds per time step. This gives a parallelization 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 parallelization. This would 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 parallelism.

The reason why I am not normally using parallelization in Si experiments is that I make these physically big (like the one just 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 cache, which enables the program to work about ten times faster than it would otherwise. The cores of the CPU have their individual memory registers, so a four-core parallel 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.

6.4 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. 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 (parallel) and perspective projections.

Recent motion is an ad hoc quantity for seeing "where things happen". For one particle, I define $E_{\text{over}} = E_{\text{k}} - n_\sigma E_{\text{thermal}}$, 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 E_{k} 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

$$\texttt{recmot} = \sum_{i=0}^{\infty} \lambda^i E_{\text{over}}(t - i\Delta t), \tag{6.4}$$

where λ is a decay parameter, set to e.g. $\lambda = 0.98$. 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` in the `.vtk` files. 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 the simulation plots in this thesis. On paper, 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, like energy, temperature, stress and strain, are output to a file for each time step for later plotting with Matlab.

6.5 Algorithm overview

Figure 6.3 shows a flowchart of procedures in the program. A simulation is divided into phases, for example an equilibration phase, a dynamics phase forcing the system into a desired state and a data gathering phase. These phases are again divided into thousands of time steps, which constitute the time loop. The letters next to the flowchart boxes are used for showing where the execution of the functions in appendix A takes place.

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 (appendix A.3).

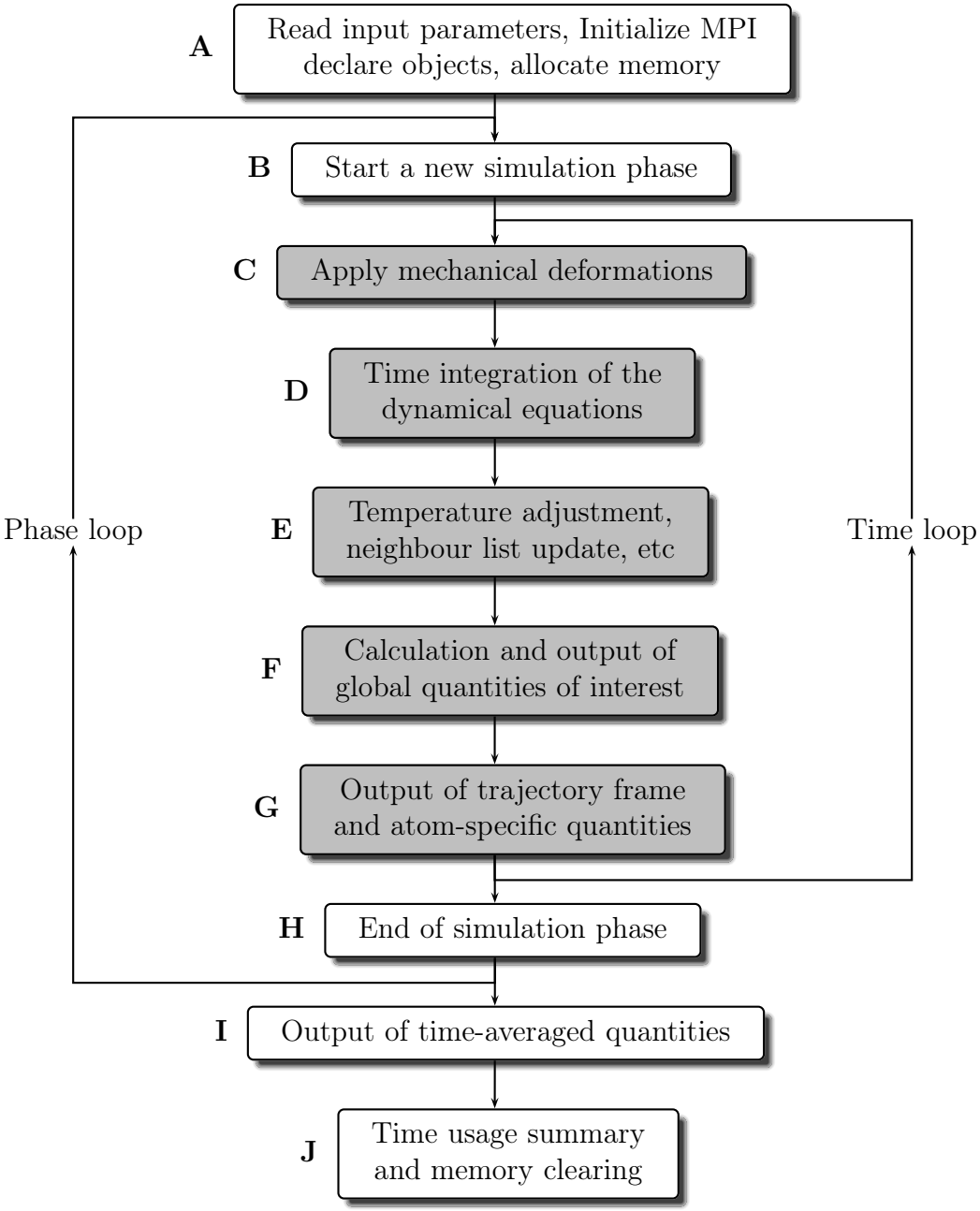


Figure 6.3: Flowchart for execution of the MD program.

Chapter 7

Bulk fracture results

The energy required to break a perfect crystal is very high, comparable to the energies associated with chemical bonds. A crystal with defects will have some atoms which are not saturated with electrons, that is, they have fewer chemical bonds than a stable bulk atom. I attempt to model bulk fracturing for cases with and without defects to study how much say they have in the fracture process for the two materials I consider. I will state some observations from my fracture simulations, starting with equilibrium properties of bulk NaCl and Si.

7.1 Equilibrium properties and general notes

As a test of my Ewald summation technique implementation, I have 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 [30]. This is, by a good approximation (for a low density gas), minus the total potential energy of the lattice. With an MD simulation with 4096 ions, the potential energy of the perfect lattice measures −4.076 eV per particle. The energy deviation is negligible upon a change in system size. At a temperature of 300 K, the total potential energy fluctuates around −4.036 eV per particle with a standard deviation of 0.001 eV per particle. The average result is 1% higher than the experimentally measured energy, which is an acceptable difference. More specifically, the energies from the Lennard-Jones interaction, the short ranged Coulomb interaction and the long range Coulomb interaction including the 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 per particle [31]. My simulation using the Stillinger-Weber potential gives the 6.5% higher number −4.337 eV. With a temperature of 300 K and 4096 silicon atoms, the average potential energy is −4.2967 eV. This is somewhat inaccurate, but the SW potential is still believed to give a qualitatively correct dynamical behaviour for silicon. The measured 300 K 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 equilibrium angle of 70.53° between their neighbours simultaneously, giving a low energy contribution from the three-body interaction term. For both NaCl and Si, the deviation in energy between a 3D PBC system and a 2D PBC system with fixed particles is insignificant.

The BDP thermostat does a good job cooling down edges of the simulation box when a fracture is occurring. 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 lose energy to heat inside the crystal. Such collective motion make temperature measurements inaccurate, but temperature is either way a quantity associated with thermal equilibrium and will generally be inaccurate in non-equilibrium situations. In the equilibration phase before any deformation is applied, the BDP thermostat is used with a coupling to all atoms.

7.2 Tensile stress

I have performed several simulation runs of a system with the tensile stress deformation 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.

7.2.1 Silicon

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 occurs when the strain ϵ is about 43%, 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 dynamical 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, and only in the fracture region.

If you neglect the thermal movement, we are dealing with a perfect crystal of atoms being stretched. In my simulations of this system, fractures always start out by the breaking of single bonds. Normally, an atom in a perfect diamond structure will have four neighbours. This means four two-body force and energy terms for each particle. At the moment this number decreases to three for some atom, this and nearby atoms start to show signs of rapid motion. What happens is that the thermal motion kicks two atoms further apart than the SW interaction range. A growing sphere of vacuum with its center at the point of the bond breakage 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 shows a set of output macroscopic quantities from a typical silicon tensile stress simulation. The oscillations in the potential energy are caused by the elastic waves of compression and expansion occurring in the two crystal parts after

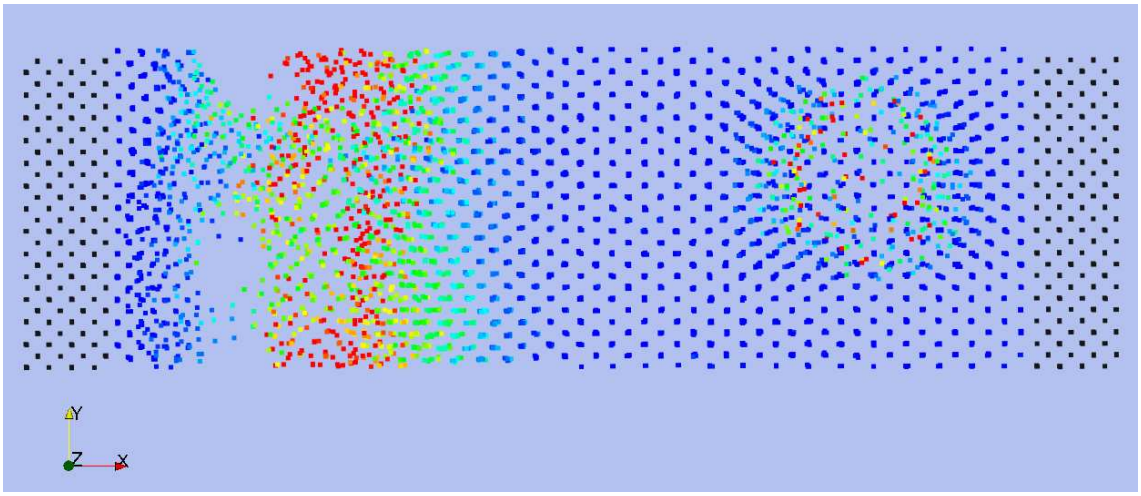


Figure 7.1: Tensile stress fracture of a crystal with 5488 Si atoms at 260 K. The colouring shows the values of the recent motion quantity. The fixed particles are shown in black.

fracture. The same goes for the kinetic energy and therefore also the temperature. The fact that silicon builds up a lot of elastic energy before suddenly breaking makes it a very brittle material. The thermostat acting on the edges of the simulation box provides a damping of the elastic waves due to outgoing heat. The collective motion of the side particles is subtracted from the speed vectors when side temperatures are measured and adjusted. 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, some of which is caused by the unphysical reflection of elastic waves.

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 breaking stress of the crystal structure. Long before this point, it is clear that Hooke's law of linear stress-strain relationships fails. For small strain, when the law is usually a good approximation, 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 different sources mention differing experimental results for its Young's modulus. The webpage [32] 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 numerical measurement is the interatomic potential form, which could be improved by methods briefly mentioned in chapter 4.

The vertical lines beneath the stress graph are artifacts 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 formations of new chemical bonds are not discovered quickly enough because of the limited neighbour list update frequency. This creates a small shift

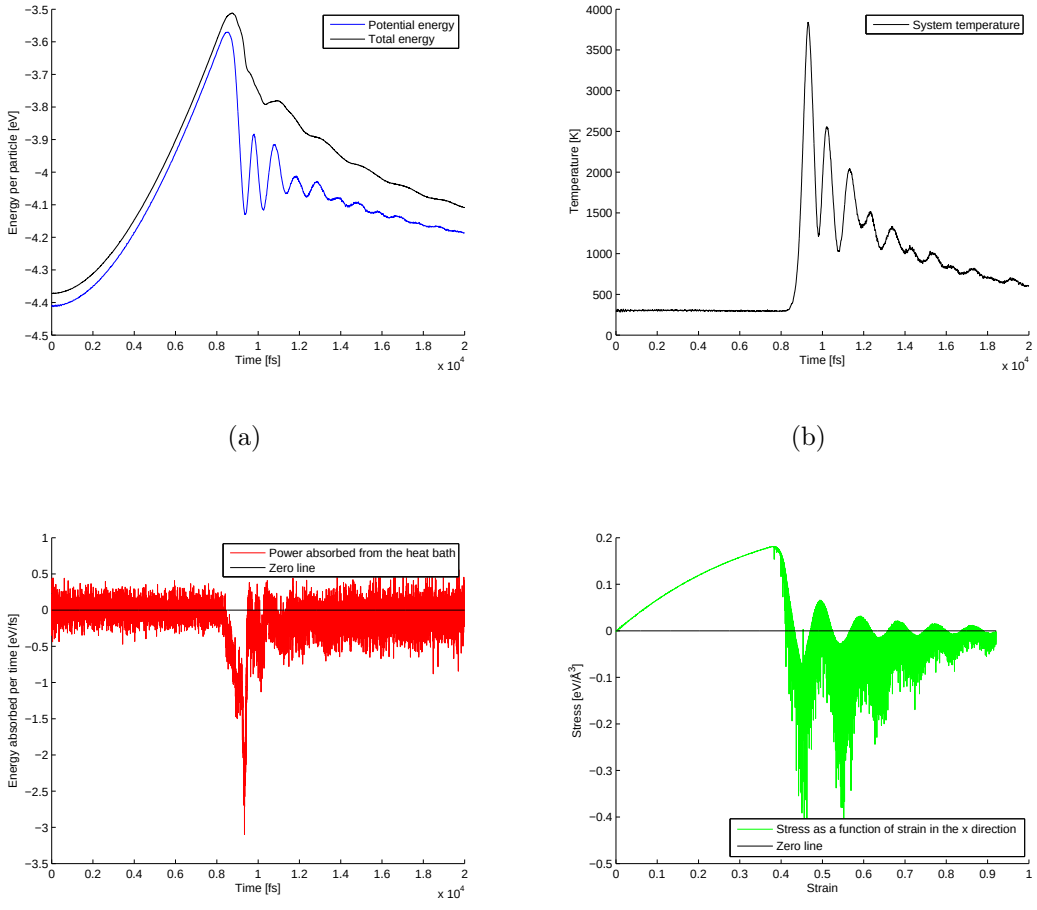


Figure 7.2: How macroscopic quantities evolve when a silicon crystal with 4096 atoms fractures. The temperature is estimated with $T = 2U_k/3k_\mathrm{B}$.

in potential energy. The reason why I have not fixed is that the CPU time usage increase greatly with cutoff range and update frequency, and these parameters are at sufficient values to simulate what happens until the fracture occurs. For the particular case of tensile stress, the measurements are in any case unreliable after this point.

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 breaking stress lowers. 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.

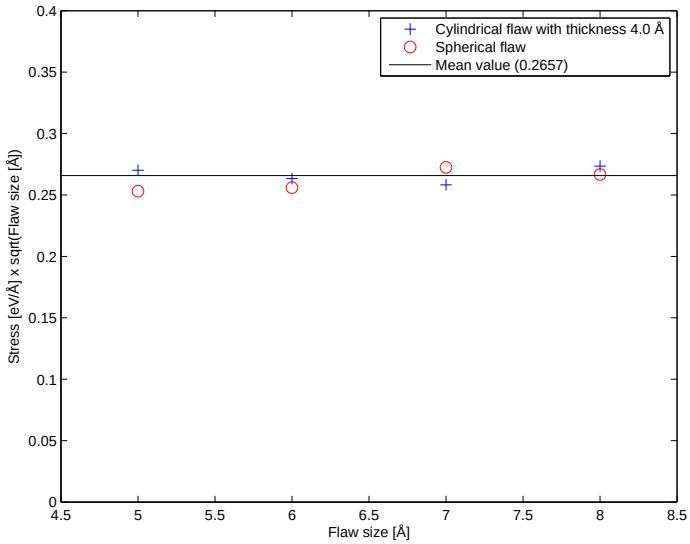


Figure 7.3: $\sigma_f \sqrt{a}$ as a function of a for silicon. The "flaw sizes" a are the radii of the spherical and cylindrical voids.

In these simulations, the upper bound for the flaw size is the size of the simulation box. When this size is approached, Griffith's relation is expected to break down because the void will touch its own edges. The smallest possible void removes just one atom, and this void can correspond to many different small flaw sizes. The range of flaw sizes in Fig. 7.3 avoids these upper and lower limits with safe margins.

7.2.2 Sodium chloride

When a crystal of sodium chloride is stretched, it does not build up potential energy and fracture 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 observe a fracture at $\epsilon \approx 63\%$ in simulations of perfect crystals. Before a fracture, the atoms float

about lazily in a NaCl substance of decreasing density. There is, however, evidence of crystal structure under the whole simulation, so the system does not seem to undergo a local phase transition.

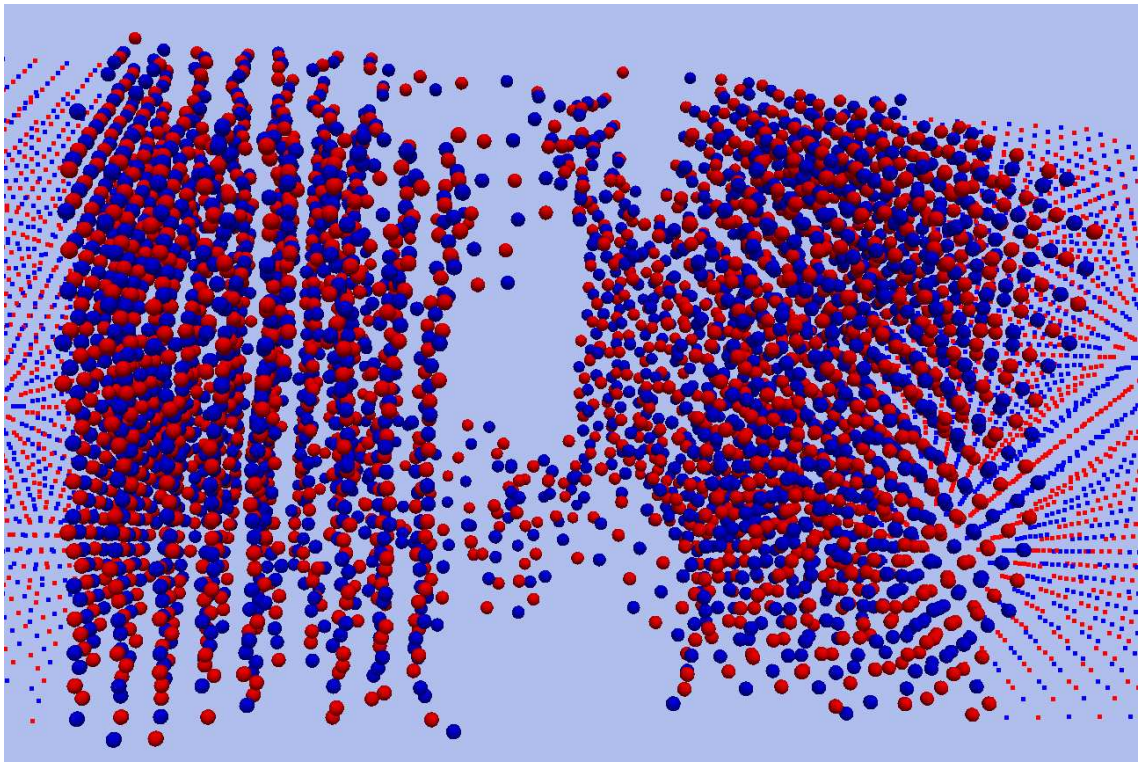
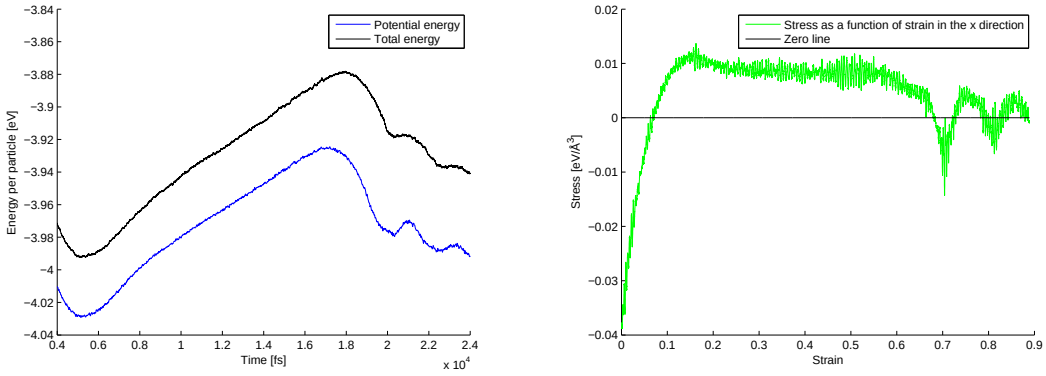


Figure 7.4: Tensile stress fracture of a NaCl crystal with 4096 atoms (minus atoms removed by a spherical defect) at 300 K.

The NaCl crystal breaks as soft 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 ions make a less considerable jump than the silicon atom velocities 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 location. 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 flawed NaCl crystals. 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 Coulomb interaction. In a simulation without the Ewald summation over $\mathbf{k}$-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 the stress-strain graph, which is characteristic for a ductile material. The stress is kept at the same level for a long period, bearing signs of the stretching process being quasistatic. Defect size does not seem to influence the breaking stress of the NaCl crystal. The dynamics stays

the same for differently sized spherical and cylindrical voids, but a greater defect has a bigger probability of influencing where the fracture will occur.



(a) (b)

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

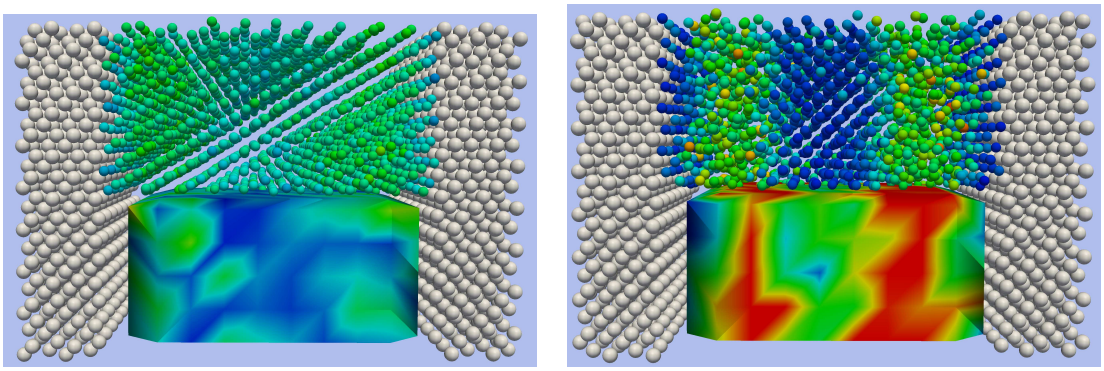
Right after fracture, negative stress values are caused by particles of high kinetic energy compressing both parts of the crystal. There are some oscillations caused by elastic waves, though not as strong as those in a newly fractured Si crystal. The negative initial stress is a small mystery. It seems the crystal is more stable at 10% strain than in its supposed stable state. The pressure caused by the finite temperature is one possible explanation.

7.3 Shear stress

When shear stress is applied to a crystal, in the way explained in section 5.2.2, 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.

Silicon crystals clearly experience phase transition at some points in the simulation box. In the simulation visualized in Fig. 7.6, these areas emerge at the side of the box, making the energy distribution symmetric along the x axis. This phase transition can also happen in only one area, in the middle of the box. The areas that do not become liquid keep their crystal structure. A collection of disordered (liquid) silicon atoms has more bonds per atom. Therefore, their potential energy is higher, which is visible in Fig. 7.6(b). The deformation normally occurs at a shear strain of $\gamma \approx 36\%$.

Sodium chloride shows a quite different behaviour when deformed with shear stress. The atomic layers get increasingly lopsided before the layers slip vertically at an y-z-plane in the middle of the box. The system again looks like a crystal in equilibrium, but the temperature is risen because of the internal friction created by the slip motion. The cycle of tilting and atomic layer slipping repeats itself as long as the fixed particles are kept in motion. There are big temperature fluctuations,



(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. In the lower half of the box, temperature values are shown.

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. The shear strain on deformation is quite similar to that of silicon, $\gamma \approx 34\%$.

There is little difference in the dynamics for a simulation without long-range Coulomb 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 all types of interactions turned on. This is a process which happens simultaneously everywhere in the middle y-z-planes. The long-range forces seem resistant to the slip motion, which is a large-scale displacement from equilibrium.

Again, defects lower the breaking strain, but the effect is less prominent for shear fractures. Little change is observed in the dynamics, but the energy barrier for the deformations is of course lowered, so that less heat is generated upon shear fracturing.

Shear stress graphs are shown in Fig. 7.7. Silicon reacts more strongly to the deformation at first, because of the angular dependence in its interatomic potential. The maximum shear stress reached is surprisingly close for the two materials, lying at about $\tau = 0.12$ eV/Å^3. In contrast to the tensile stress experiments, the volume of the simulation box is now constant, and stress measurements after fracture can provide more relevant information. The shear stress is fluctuating, oscillating for NaCl and varying more or less randomly for Si. This gives supplementary insight about the post-facture behaviour. NaCl needs to pass energy barriers for each slip motion, but after the temperature is increased in the first slip, successive slips require much less applied stress. The stress in a Si crystal becomes even lower, owing to the layers of atoms which is displaying liquid lubricant-like behaviour.

The results in this chapter give a feel for the qualitative differences between the materials under study. Sodium chloride is ductile and transforms via slip motions

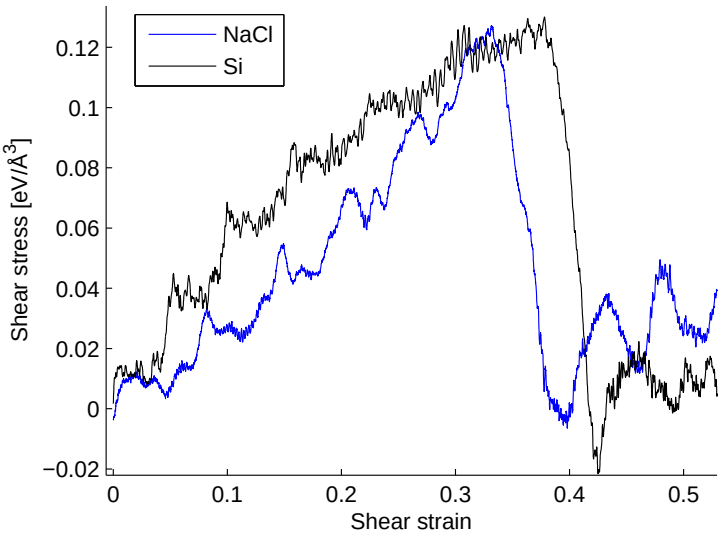


Figure 7.7: Shear stress during fracturing of perfect crystals composed of 2744 atoms at 300K.

when the crystal structure is forced alter. This is a sign of a ductile material, and a characteristic it shares with metals. The ionic potential is surprisingly effective at keeping the double fcc lattice structure intact, both upon stretching and fracturing. Silicon, on the other hand, breaks with a snap, like brittle glass. In both the investigated types of deformations, areas where the atoms display liquid-like behaviour appear. The transition to this phase does not require much energy. As also seen when looking ahead in the thesis, the effects of tensile stress can be summarized in this way: Silicon is easy to compress and hard to stretch, while sodium chloride is hard to compress and easy to stretch.

Chapter 8

Contact and adhesion results

8.1 Sphere-surface contact

I have performed computational contact experiments for the systems described in section 5.2.4. 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 amount of time, I have limited the system size to 4232 dynamical and 1600 fixed atoms for silicon, and 2106 dynamical 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. The spheres start one extra atomic layer above the plane surfaces of fixed particles, for the contact forces to be negligible at simulation start. The equilibration proceeds as described in section 3.3. An example simulation is visualized in Fig. 8.1.

8.1.1 Deformations and contact area

Results for the measured contact area [using Eq. (5.12)] are presented in Fig. 8.2. In addition, the center of mass x coordinate for the sphere atoms is measured in the same simulation runs and plotted in the same figure. The force affecting the top halves do not seem to compress the spheres significantly. In a rather large force interval, they are pushed towards the surface and the bottom atoms rearrange only slightly. This is a small plastic deformation, but the added energy from the external potential is mostly stored as elastic energy. When the force reaches a certain threshold, a sphere will collapse under its own "weight". The center of mass x coordinate and the measured contact area confirm that this happens quite abruptly. The force threshold varies slightly between simulations as their dynamical nature introduces great fluctuations. Nevertheless, pushing an asperity towards a surface creates a binary situation where the material either has undergone a greater plastic deformation or not. The two materials act similar in this respect, but the microscopic processes responsible for the effect are different.

In the silicon sphere, 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 occurring in the earlier mentioned shear stress

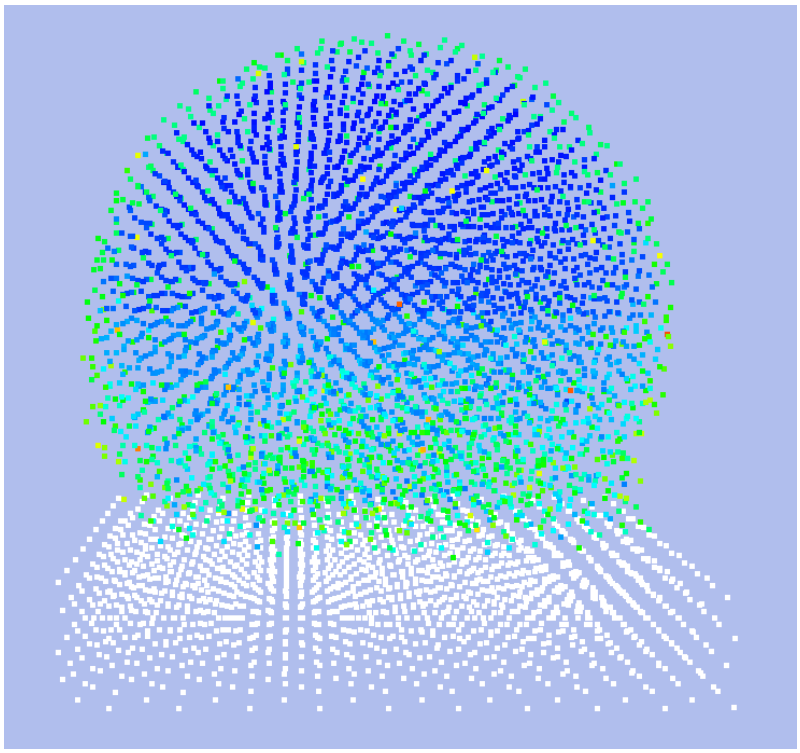
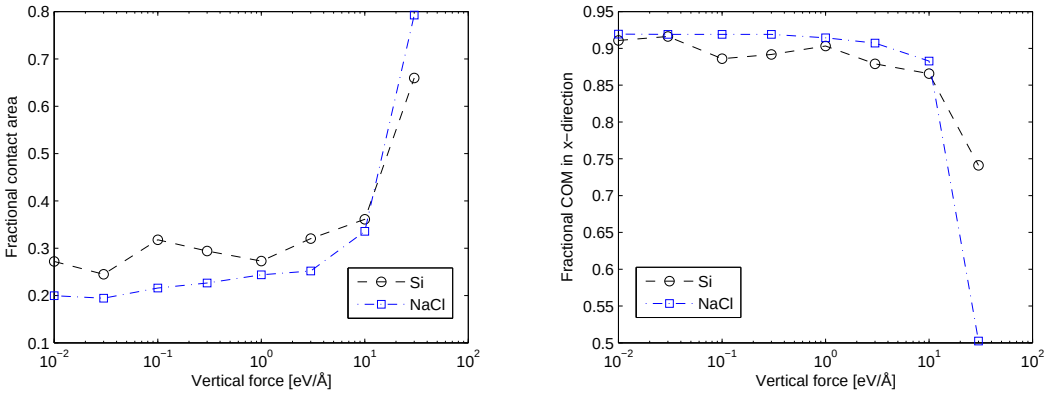


Figure 8.1: A ball of silicon pushed against a silicon surface with a force of 30.0 eV/Å. Atoms are coloured by potential energy. The discontinuity in this colouring is caused by the fact that only the upper half of the particles are subject to the fictitious gravitational force.

experiments is responsible for reducing the number of atomic layers in the vertical direction. The configurations preceding and following a slip 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 over time, but over 8 different initial configurations (random seeds) when producing the silicon results. This makes the structure of the stress graph in Fig. 8.4(a) more apparent. The graphs for the different runs contain trends similar to Luan and Robbins' result for the bent crystal in Fig. 2.3, where a harmonic potential is used. They also contain a great deal of noise, because the bottom sphere particles arrange randomly. The outer moat of attraction is caused by adhesive forces between atoms at the edge of the interface. These are not pushed close enough together to repel. Around 20000 dynamic particles were used in the simulations for creating the stress graph.

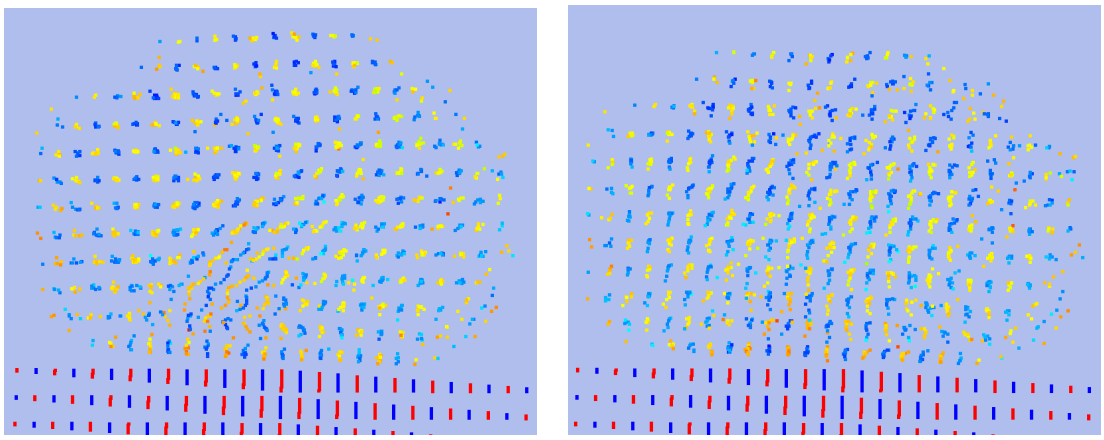
The stress graph for the NaCl system is very different. The sign of the stress has an opposite character: The middle particles attract each other slightly while the outer particles repel each other [see Fig. 8.4(b)]. The graph looks the same for different initial conditions, with the exception that the ball may be displaced one atomic distance in some direction on the surface. The sphere particles arrange



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

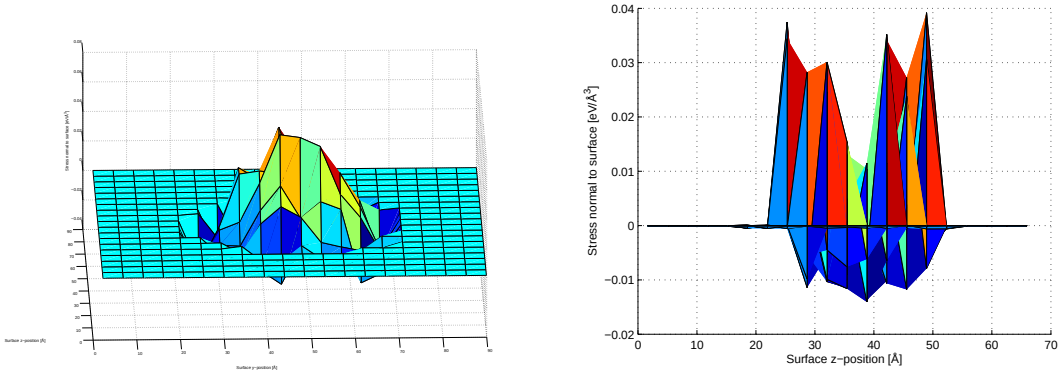
(b) Center of mass in the x direction in units of one sphere radius.

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



(a) (b)

Figure 8.3: The deformed NaCl sphere before and after a slip motion. A height difference is evident. The atoms are coloured by potential energy, excluding long range (reciprocal 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.

very regularly. Small, regular zigzag patterns in the stress graph are caused by the duality of attraction and repulsion in the Coulomb potential. Around 7000 dynamical particles were used in the simulation. The reason for the odd surface stress graph is investigated in section 8.2.

In both the Si and NaCl cases, the sum of forces between the sphere and surface particles lie within 1% of the applied pseudo-gravitational force. This validates the results by assuring that Newtons second law is fulfilled for the entire sphere.

8.1.2 Dependence on relative rotation

Up to this point, all experiments have been performed using commensurate surfaces. I am also interested in investigating the forces between relatively rotated surfaces. This has no effect for silicon, due to its liquifying behaviour, but sodium chloride crystals are strongly affected by rotations.

An arbitrary rotation angle of $\theta = 65.9°$ is used for an example simulation. As expected, the potential energy of the initial configuration is now higher when the sphere and surface start in contact. In fact, the interface forces are strong enough to send the sphere away, so it must be kept down with a pseudo-gravitational force, which I set to as much as 50 eV/Å. The sphere must start in contact with the plane surface, which assures that no deformation takes place. Strong vibrations occur, but the contact stress graph does not show regular plateaus as in Fig. 8.4(b). Instead, more irregular spikes of stress appear, as the charge distributions of the two surfaces now look relatively different at the same y-z-points.

An important question is how the contact force varies with the rotation angle. I investigate this only for the initial (non-equilibrated) state of the sphere-surface system. This is because the sphere would have to be pushed down during equilibration and measurement phases, making force measurements difficult. Figure 8.5 shows the resulting contact force for systems rotated with an angle of $\theta = n°$ for $n \in \{0, 90\}$.

As expected, the graph is symmetric about $\theta = 45°$. When the sphere is rotated further, the graph repeats itself with a period of 90°. I read the top angles of the two first peaks to $\theta \approx 14°$ and $\theta \approx 31°$. These peaks are most probably associated with the special angles 15° and 30°. The potential energy graph has a similar behaviour, but with much smaller deviations. It is unlikely that the form of a corresponding time-averaged contact force from an equilibrated system would look much different.

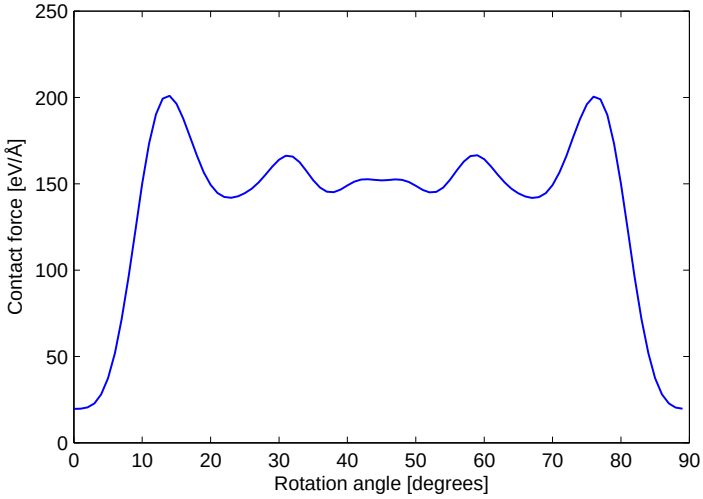


Figure 8.5: Force between a NaCl surface and a rotated NaCl sphere. The peaks are located at 14°, 31°, 59°, and 76°.

In a macroscopic solid, the crystal structure of asperities on two rough surfaces will be relatively rotated. It is a reasonable assumption that the rotation angles is uniformly distributed, since all asperities can not fall into the lowest energy state (the commensurate state with $\theta = 0°$) at the same time. Therefore, the mean of the graph in Fig. 8.5 is more representative than the contact forces measured earlier. In this case, the mean is about 7 times larger than the force in the commensurate state. Given the problems of rotating structures with PBC, I will not pursue the matter further, but these results are useful as a side note.

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 edges of the contact area. The ionic interactions are actually fully adhesive, and counteract the repulsion at the edges. The short-range part of the Coulomb force does all the work, while the long ranged part do not contribute significantly to this force. Natural oscillations are nearly certainly the reason behind the stress spikes at the edges of Fig. 8.4(b). As seen in particle trajectories and center of mass plots, the sphere tips back and forth in both directions tangential to the plane surface. This periodically causes the contact particles at the edges to come closer to the surface and experience a vastly bigger LJ force. A peak is created, significantly taller than

the mean stress across the contact area. At the same time, adhesive forces give a negative stress around the center. When averaged over time, this effect is the prominent feature of the stress graph.

When the sphere is truncated and instead pushed down by fixed particles, the most important difference is that the dynamical atoms are held strongly in place at the positive x side of the simulation box (the top). Therefore, global oscillations are avoided. The sphere segment is slightly compressed between two plane surfaces, and a more uniform stress than before is measured. Figure 8.6 shows such a stress graph, with a total contact force of 131 eV/Å. This number may seem very high in comparison with previous results, but the force is applied in a more gradual manner than before, and the sphere is still not collapsed. The upper fixed layers move downwards with a constant speed of $2 \cdot 10^{-4}$ Å/fs. With a little more applied compressive strain (longer indentation time), the sphere segment will deform in a slip motion as before, creating a bigger contact area and lowering the total force. NaCl has a much stiffer crystal structure than Si, which in the absence of a sudden impact makes it more durable.

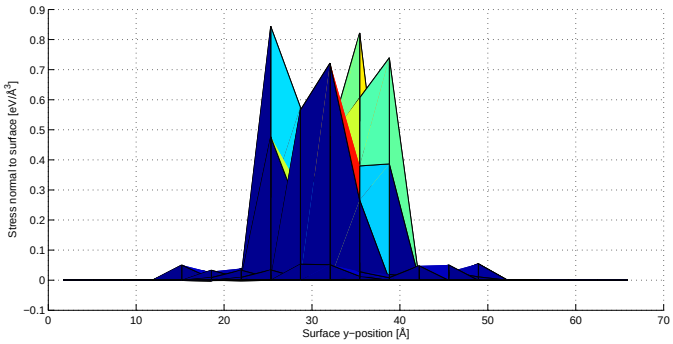


Figure 8.6: Contact stress between a NaCl sphere segment and a NaCl surface. The applied compressive strain is 4.1%.

In another attempt to explain the strange sphere-surface stress graph in Fig. 8.4(b), I observed a deformation in the lower part of the sphere for large applied forces. However, this deformation is too small to create such a big effect, and it also occurs in the sphere segment system. I refer to the deformation, which is clearly visible in Fig. 8.7, as the bow phenomenon. The figure is from the same simulation run as Fig. 8.6. The effect of can be seen here as a small ring outside the main stress plateau. The bow phenomenon is bound to happen in real systems, but the effect should not be very noticeable. Earlier studies show a similar phenomenon occurring, visible in the rightmost part of Fig. 2.3.

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 [10], which works only with LJ and harmonic spring potentials, describe similar results for the corresponding microscopic structures. Different potential symmetries will induce differing reactions to an asperity-surface interaction,

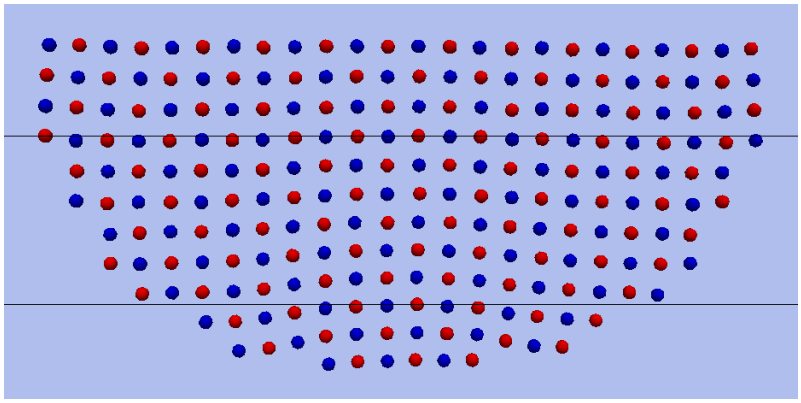


Figure 8.7: Arrangement of one x-y-layer of NaCl sphere segment ions. The applied compressive strain is 4.1%. Straight horizontal lines are added as eye guides.

leading to either destruction or some form of conservation of the material's default bulk crystal structure.

8.2.1 Adhesive energy

Contact forces between a plane surface and a model asperity have been investigated. The energy involved in the squeezing process is also of interest. 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. The sphere segment will not start in slight contact with the lower fixed particles, as in Fig. 5.6. Instead, I place a gap of 4 Å under the sphere segment, so the surfaces are approximately out of interaction range. This 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 from its default position as the *indentation distance*, denoted d. This should not be confused with the center of mass x coordinate, which is measured in section 8.1. When the sphere segment undergoes an elastic deformation, d and x_{COM} will differ. When the surfaces are pulled apart, d becomes negative. In the initial position, the indentation distance is defined as $d = -4$ Å.

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.

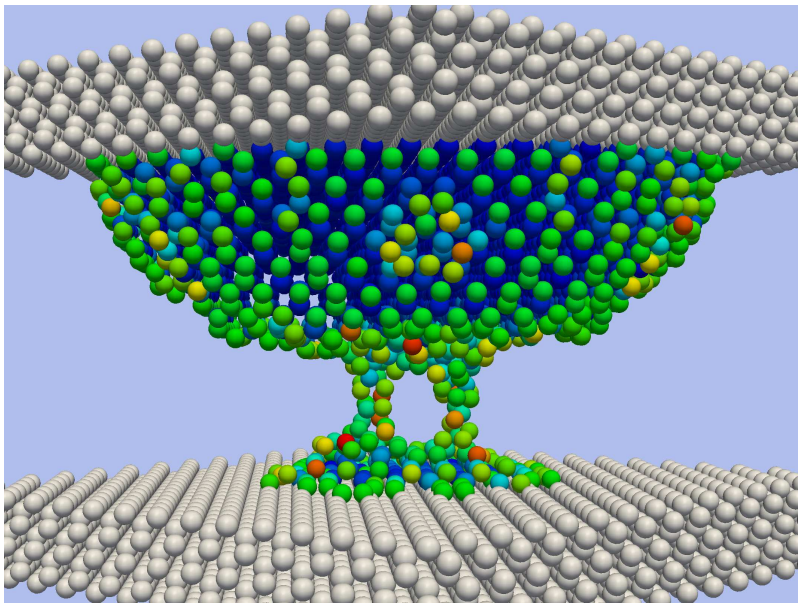


Figure 8.8: An Si asperity pushed into and separated from a Si surface. $d_{\max} = 3.2$ and $d_{\text{current}} = -12.0$ Å. The atoms are coloured from blue to red by their potential energy.

The sodium chloride system is again made much smaller than the silicon system, with a sphere segment height of 3 unit cells and upper diameter of 9 unit cells. Only 840 dynamical particles are simulated, but this seems to be enough to get a qualitative picture of what is happening and approximate numerical results. The material reacts in a very different way to 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 small, not very noticeable in an animation of the trajectories, and they change to a small repulsion already at $d = -1.3$ Å. If compressed enough, the sphere segment will deform by a slip motion. 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 $d = 2.4$. This run was discarded because this effect alters the energy state.

In all my simulations with $d_{\max} \geq 1.2$ Å, a plastic deformation occurs. When the surfaces are separated, one horizontal monolayer of NaCl connects the two surfaces until about $d = -9$ Å. This layer hangs between the surfaces like a carpet, in much the same way as the torn flakes in NaCl fracture simulations (see the top of Fig. 7.4). This makes the separation energy much higher. Without this effect, the energy loss in the process is in fact very small, as found out from simulations where the Coulomb interaction strength between dynamical ions is increased.

Figure 8.10 shows the numerical data from adhesion runs with the two materials. For Si, a lot of energy is needed to pull the surfaces apart, and not as much for pushing them together. It is reasonable 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

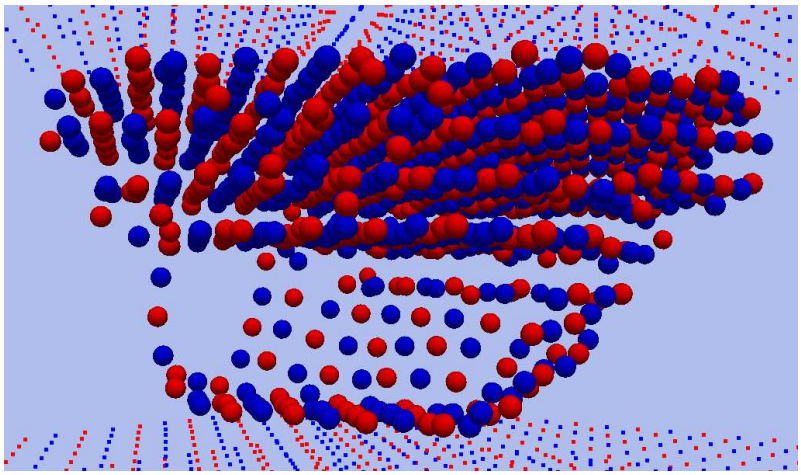
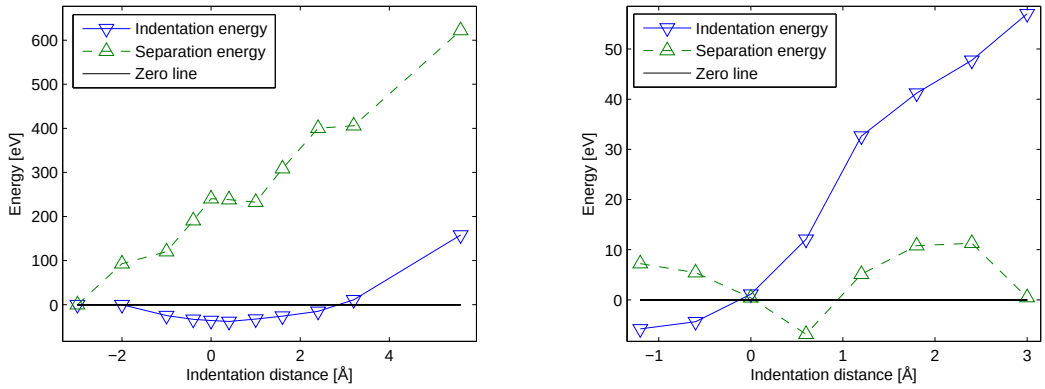


Figure 8.9: A NaCl asperity pushed into and separated from a NaCl surface. $d_{\max} = 2.4$ Å and $d_{\text{current}} = -4.0$ Å. A "carpet" of ions stick to the lower surface.

apparent plateaus 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. This is not so apparent in the indentation energy. The probable reason is that adhesive forces drag atoms down and induce great velocities, which is transformed to heat and conducted out. The indentation itself does not provide much energy in the process.

In the far left of Fig. 8.10(b), the NaCl system experiences only a small adhesion effect. It gains energy by attaching the two surfaces and input energy is required to break them apart again, as in the Si system. A beautiful symmetry between adhesion and repulsion reduces these energies to approximately zero when $d_{\max} = 0$, that is, the surfaces are brought to one regular atom distance apart. When pushed further, the sphere segment stores elastic energy before a plastic deformation occurs at $d \approx 1$ Å. This deformation brings the system to a preferable energy state. The deformation itself requires a higher indentation energy, and more energy is required to pull the system out of this low energy state. At $d_{\max} > 2$, an increasing amount of elastic energy is stored. In short, an elastic deformation lowers the separation energy, while a plastic one increases it. In bigger systems, several stages of deformation will be possible. I then expect the concave form of the separation energy to repeat itself for higher indentation distances.

Heat plays a big role in the adhesion process. In addition to the energy needed for deforming the systems to higher potential energy states, heat is what is responsible for the apparent hysteresis in the process. I will not present the amounts of heat dissipation systematically, but will give some estimates. In the silicon system, almost all the energy that is gained from adhesion and roughly half the applied separation energy is lost as heat. Meanwhile, the NaCl system goes into a higher potential energy state when compressed elastically. When separated again, this energy is forced lower, and the system actually gains extra heat from the outside. The temperature is kept at 300 K, as in most of my simulations. If heat was not



(a) Silicon.

(b) Sodium chloride.

Figure 8.10: The energy required to bring the surfaces in contact and pull them apart again, respectively. For various maximum values of the indentation distance d. The system sizes are different, so the absolute values cannot be compared between Si and NaCl.

conducted out of the systems, the indentation and separation energies would have been closer to having equal values with opposite signs.

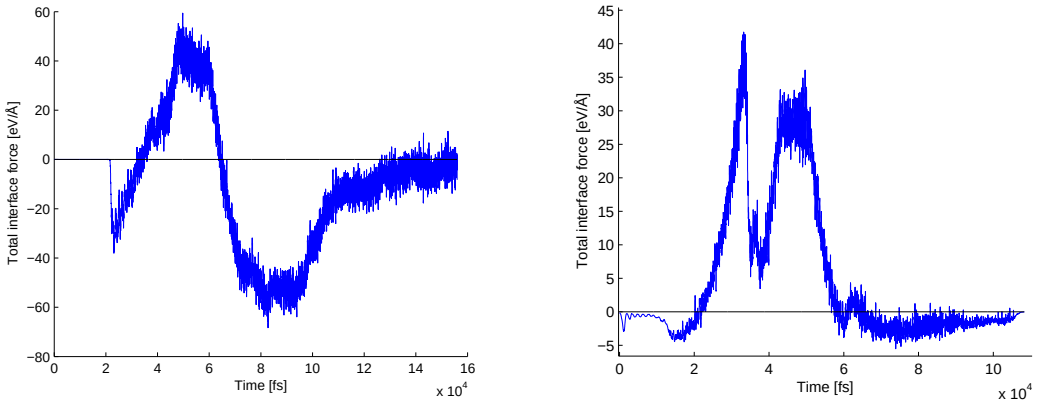
The last figures I want to present from these experiments are two example plots of total interface force as a function of time (Fig. 8.11). On the silicon graph, it is visible how suddenly the adhesive forces are turned on. The force varies almost linearly (elastic Hook-spring-like) when the sphere segment is pushed further in, except for some irregularities bearing signs of plastic deformations. The skewed plateau 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, small adhesive forces are visible in the entry phase. The force varies roughly linearly with time (and displacement), only interrupted by a sudden, grand deformation. In the NaCl simulations where such a deformation does not occur, the force graph is quite symmetric. The process is then close to reversible, as seen in Fig. 8.10(b) for $d_{\max} < 1.2$.

Dependence on parameters

Some simulations have been performed to see how much the adhesion and separation energies vary when different conditions are changed. I have tried changing three different parameters from a reference Si simulation: the radius of curvature R for the sphere segment (creating a system with more or fewer atoms), the base thermostat relaxation time τ and the speed of indentation and separation v. The parameters of the reference simulation are labelled by a 0 subscript. Table 8.1 shows the energies measured in simulations with differing parameters.

The separation energy increases with the radius of curvature, as expected. A



(a) Silicon, $d_{\max} = 3.2$ Å. (b) Sodium chloride, $d_{\max} = 3.0$ Å.

Figure 8.11: The total force between the lower surface and the sphere segment during adhesion simulations.

Changes	Indentation energy [eV]	Separation energy [eV]
Reference (R_0, τ_0, v_0)	**11.0**	**405.8**
$R = 1.2R_0$	2.6	529.3
$R = 1.4R_0$	7.5	540.7
$\tau = 4\tau_0$	10.3	397.5
$\tau = 40\tau_0$	17.1	374.5
$v = 2v_0$	18.81	407.3
$v = 4v_0$	23.8	410.4

Table 8.1: Energies measured in Si adhesion simulations using singly different parameters than the set used for previous results.

flatter sphere will give an increased area of contact and more atoms stick to the lower surface. The energy will also depend on the exact structure of the tip, so it is difficult to find a general relationship between radius of curvature and separation energy in systems of this size. This is probably also the cause of the fluctuations in indentation energy. Reduced heat conduction causes less heat dissipation during the separation, while the cause of the increased indentation energy is less clear.

The driving speed of the fixed particles is almost negligible in comparison with thermal speeds, as mentioned before. Altering this speed therefore makes no big differences in the energies. Though, the sphere is forced faster into a lower energy state on indentation. This causes a lower need for transformations from potential to kinetic energy by the dynamical particles, leading to reduced heat loss. The snap when the surfaces are separated are not as violent, so the effect is barely visible here.

As an additional test, I have run ionic lattice adhesion simulations where the Na^+ ions are switched with K^+ ions. The potassium atom has a mean mass of 39.0983 amu [12], where the mass and binding energy of a missing electron is neglected,

as before. The mass of the K atom is close to that of the Cl atom, which might give differing microscopic dynamics from that of NaCl. The LJ parameters for potassium can be found in Tbl. 4.1. The lattice has greater oscillation amplitudes, caused in part by weaker LJ interactions. However, I see no difference in the adhesion dynamics, and all measured macroscopic data, including contact stress and adhesion energy, do not differ in behaviour from the NaCl data.

8.3 Friction experiments

Measuring quantities related to friction from simulations on the atomic scale is a very ambitious task. The length and time scales which are required for accurate measurements is difficult to achieve, and the external forcing of the system must be carefully adjusted in order to reproduce a realistic situation. My numerical framework and available computing resources is lacking, but I have made some attempts to produce stable dynamics where two surfaces slide in contact.

In initial attempts, I used the sphere-surface contact system. The sphere, which inevitably begins to roll, may do a poor job at describing a rough macroscopic surface asperity in a time-dependent situation. While the sphere rolls/slips across the infinite layers of fixed plane surface atoms, more sphere particles become attached to the surface, tearing off more and more atoms from the sphere. Data for the relation between contact area and frictional force is therefore not reliable after a certain period of time.

The data contains great fluctuations, but large-scale shapes of the A_c-F_{friction}-distribution are visible. For NaCl, the frictional force seems to be distributed equally around zero at all times and have no distinguished pattern. As in the low-$d_{\max}$ adhesion simulations, there seems to be little energy loss. Interestingly, the silicon data indicates a linear relationship between A_c and F_{friction}, though the fluctuations are of the same size as the increased friction force from small to large contact area (see Fig. 8.12). The proportionality factor c introduced in Eq. (2.7) take a value around 0.007 eV/Å^3. For different values of the driving velocity and the pseudo-gravitational force, I see no trend of change in the frictional force (fluctuations dominate).

Using the sphere-segment system with a stepwise load balancing method [Eq. (5.14)], I also got inconsistent results for F_{friction} as a function of L. In the Si system, the time developement of frictional force is very similar to the one earlier measured and plotted in Fig. 8.12. The frictional force decays to zero once the sphere segment has disintegrated into two parts sticking to each fixed particle surface. (The load balancing stops after sideways movements are induced. If I continue the balancing, the sphere segment is eventually flattened and the system from section 5.2.2 is reproduced.) The result from a big simulation with 6790 dynamical and 10368 fixed particles is a friction force $F_{\text{friction}} \approx 35$ eV/Å for a load $L = 7.0$ eV/Å. Thus the friction coefficient, which is meant to lie in the range $[0, 1]$, is measured to $\mu = 5$. This with a driving speed of 0.002 Å/fs.

The results from a simulation of the NaCl sphere segment system is similar in the respect that a deformation occurs. When held in place, the system is stable. When moved, the lower layer of the sphere segment hangs on to the lower fixed ions.

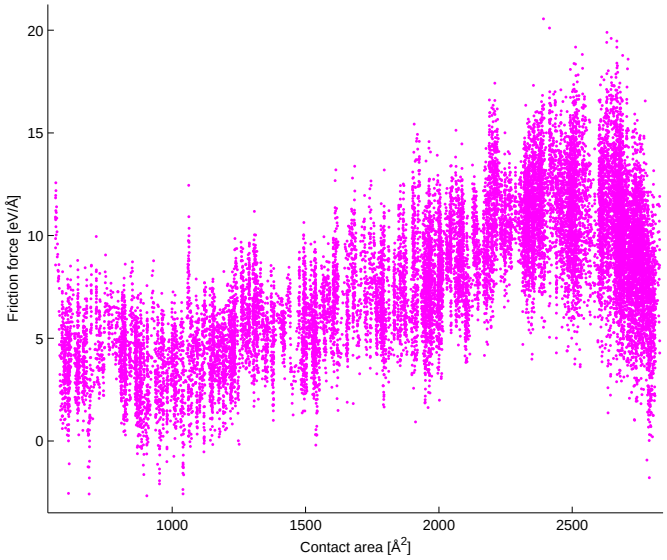


Figure 8.12: F_{friction} vs A_c for a silicon friction experiment with about 4000 particles in a sphere. One data point per time step.

We see a slip phenomenon similar to the one in Fig. 8.3, creating a larger contact surface. The friction force is even harder to measure in this case.

When the electrostatic forces between only the dynamical ions are increased by a factor of 4, the dynamics are more interesting. This was initially the result of an error in the program code. Figure 8.13 shows how the friction force oscillates when the sphere segment slips across the fixed ions in the lower surface. Because of the increased forces inside the sphere segment, it is harder to deform and stays intact during the simulation. By friction force microscopy, oscillations similar to these have been discovered in actual experiments [1]. There seems to be almost no total energy loss to friction between NaCl surfaces in my erroneous experiment. I am unable to reproduce this effect with real force strengths, but it might occur in other ionic lattices if the conditions are right.

Interactions between a spherically shaped asperity and a plane surface have been the main subject of study in this chapter. Both similarities and differences between systems of silicon and sodium chloride have been uncovered. Though the reactions to applied compressive stress are different, sudden plastic deformations occur in both cases, compacting the asperity and lowering the potential energy. Before the deformation, the stress distribution between the surfaces is characteristic for the surface structure of the asperity. Si has the results of a bent/random surface while the NaCl stress graph looks like that of a stepped crystal surface. The reference for these characterizations is Fig. 2.3.

The two materials also have their own adhesion mechanisms, seen in Figs. 8.8 and 8.9. The lattice energies of the two materials are very similar, but much more

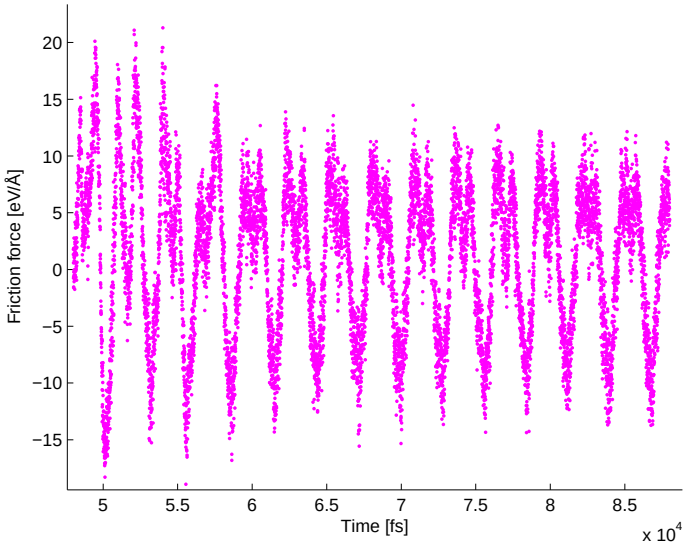


Figure 8.13: F_{friction} vs t for an erroneous NaCl sphere segment friction experiment with 840 dynamical and 2592 fixed ions. The strength of the interactions between dynamical ions are increased fourfold.

energy is still required to break a silicon crystal. This is in part due to the short range of the interatomic potentials, which gives a large change of energy for small deviations from the stable crystal structure (that is, large forces). This again induces rapid motion of silicon atoms when critical phenomena occur, which leads to heat loss. Such effects are not as visible for NaCl. The long-range dual-signed Coulomb interactions even out and create a weaker mean field potential close to a lattice. This is the most important difference between the two potentials with respect to the dynamics of solids. The contact and adhesion results provide insights important for friction, although complete sliding friction simulations are troublesome to create on this scale.

Chapter 9

Concluding remarks

It is seen in the fracture simulations that Si crystals break with a much higher stress than NaCl crystals. This is the case for solids both with and without defects. Much of the breaking energy is excessive, which means that it will transform to heat and sound, but also continued motion. If you bend a silicon slab with your hands, it snaps and pieces may fly about. Meanwhile, sodium chloride is softer and more ductile, and will fracture easier and with less fuzz.

I stated in chapter 2 that ductile materials dissipate more energy in fracture processes. This is because of plastic deformations, which become more important than heat at larger scales. The energy is dissipated more gradually than in brittle materials, causing the dissipation to be less noticeable when performing macroscopic experiments. The quite different behaviour of the two material classes owes almost exclusively to the range and symmetries of the interatomic potentials.

In numerical asperity squeezing experiments, Fig. 8.2 indicate a sudden plastic deformation upon an increased applied load. The threshold for deforming is very similar in value for Si and NaCl. On a macroscopic surface, earlier experiments have shown that real contact area increases continuously with the applied load. One can draw lines between my results and this behaviour by stating that this increase is due to asperities deforming. The contact area reflects how many asperities have undergone this process, which can be called the degree of asperity deformation.

The form of the surface stress between a sphere and plane surface does not vary drastically between different forms of interatomic potentials, for given atomic positions. However, the potentials produce differing atomic position configurations upon surface contact, giving radically different stress distributions. The absolute values are greater for NaCl than for Si, displaying a higher resistance to compression (compressive bulk modulus). This behaviour is odd, seeing that the opposite is true for tensile stress. This is observed both in fracture and adhesion experiments.

Amontons' law, first shown in Eq. (2.6), is not reproduced in the numerical friction experiments. Only single asperities are simulated, and plastic deformations are seen to change the contact area, and thereby contact stress, significantly. Thus A_C is far from varying linearly with L. On the macroscopic scale, however, the degree of asperity deformation should be able to vary such that A_C is proportional to L when averaging over the whole surface of a material. The second condition that must be fulfilled is $F_{\text{friction}} \propto A_C$. Even for a single asperity, this is seen to be approximately satisfied for silicon.

Asperities are seen to disintegrate quickly in the friction experiments. This deformation is not mainly due to the load, but the shear forces resulting from the sliding. Asperities of two materials in contact must be deformed before relative sliding can occur. Consequently, the magnitude of static friction is dependent on both the adhesive properties and the shear strength of a material.

The dissipation behaviour of materials in contact is the same as in the fracture experiments. Silicon has the highest heat loss, while sodium chloride may require great amounts of energy to deform the two surfaces. These very different processes of energy dissipation are again caused fundamentally by the differing forms of the interatomic potentials. A larger-scale model including many asperities is required for finding out which material experiences the larger friction force for the same surface geometry.

The dynamics investigated in this project can be made more accurate both by scaling the system size up and down. Vast parallelization improvements are possible (by e.g. the method of domain decomposition), enabling the program to run on computer clusters with hundreds of CPUs. This allows for a larger number of interacting particles, so that fracture propagation can be studied and surface asperities can be enlarged for better detail. What I here refer to as down-scaling means taking "smaller" phenomena into account, by describing quantum mechanical effects more accurately. The electronic degrees of freedom may be simulated using DFT calculations, which are coupled to MD calculations simulating movement of atomic nuclei. This will also require great amounts of computing power.

This project was initially meant to include such a DFT coupling and the effects on corrosion (oxidizing chemical reactions) on fracture mechanisms, similar to the work of Vashishta et. al. [5]. A more composite system will show a more complex behaviour which is realistic to some systems. In addition to increased accuracy, DFT is able to handle metals, where (conduction) electrons can have widespread wave functions. A DFT-MD coupling is too great a challenge for a master's thesis when the code must be developed independently.

Appendix A

Code structure

Details regarding the code are provided in this appendix. As the program uses object-orientation, I present the classes the code is naturally divided into, and the functions these contain. Adjustable simulation parameters are also listed.

The `main`() function, which is executed on program start, reads parameters from a file, acts as a stopwatch, manages and backs up the files in the output folder and outputs most of the terminal messages. It contains the phase and 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 were presented earlier, 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 executed.

A.1 Molecular dynamics classes

- `class 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 A.3).
  - `initialize`() (**A**): Allocates 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 Eqs. (3.3) and (3.4)].
  - `velocity_verlet_2`() (**A,D**): Calculates new velocities [see Eq. (3.6)]. Run after the first force evaluation, causing the initial 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. Contains all loops over atoms and reciprocal lattice vectors, and calls the `fpot` functions. The formulas are found in chapter chapter 4.
- `update_neighbours`() (**A,E**): Updates the neighbour lists of all particles. Called e.g. every 10 time steps.
- `rescale_temperature`() (**E**): Simple temperature rescaling, as in Eq. (3.16). Normally not in use.
- `andersen_thermostat`() (**E**): Adjusts the temperature using the Andersen thermostat (see section 3.2.2). Used for equilibration in specific cases.
- `bdp_thermostat`() (**E**): Adjusts the temperature using the BDP thermostat, as in Eq. (3.18).
- `bdp_thermostat_sides`() (**E**): Adjusts the temperature of the simulation box sides using the BDP thermostat (see section 5.1.3).
- `dissipative_force`() (**D**): Adds a friction force term proportional to velocity to all dynamical atoms. Used for damping oscillations.
- `expand_volume`() (**C**): Expands the simulation box in the x direction with a specified length. Used every time step in tensile stress and sphere segment contact simulations.
- `displace_lattice`() (**C**): Displaces the fixed particles constituting the perfect crystal layers on the sides of the simulation box. Used every time step in shear stress and friction 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 a corresponding potential energy term. Used for simulating a homogeneous gravitational field.
- `adjust_vcoll`() (**D**): Forces the collective velocity components to have certain values by adding a correction term to all dynamical atoms.
- `adjust_load`() (**C**): Function for moving the particle stepwise until a desired normal force is acquired. Uses `expand_volume`().
- `output_pos`() (**G**): Outputs the positions of all atoms, including the vector and scalar values mentioned in section 6.4.
- `output_field`() (**G**): Outputs the temperature and density fields. Rarely used.
- `output_xyz`() (**G**): Outputs the positions of the simulated particles to a simple `.xyz` file for visualization with VMD.
- `kin_energy`(): Calculates the kinetic energy of the particles.
- `temperature`(): Calculates the average temperature of atoms belonging to a specific type, subtracting the collective velocity.
- `fpot_short`(): Adds up short range forces and potentials, including LJ and short range Ewald terms.

  - `fpot_coloumb`(): Adds up LJ and direct Coloumb forces and potentials, for systems with no PBC.
  - `fpot_long`(): Adds up 3D long range Ewald forces and potentials.
  - `fpot_long_2d_1`(): Adds up $\mathbf{k} \neq 0$ 2D long range Ewald forces and potentials.
  - `fpot_long_2d_2`(): Adds up $\mathbf{k} = 0$ 2D long range Ewald forces and potentials.
  - `potential_constant`(): Calculates the constant correction term in the Ewald summation. Only called in the `initialize`() function.
  - `fpot_sw_2b`(): Adds up SW potential two-body forces and potentials.
  - `fpot_sw_3b`(): Adds up SW potential three-body forces and potentials.

- `class 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 the `phase` class. It also contains timer objects for timing the execution of each phase.

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

- `class atom`: Contains the position, velocity, and force on an individual atom. The methods are used basically everywhere, 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, which is 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 (see section 6.4).
  - `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.

- `class 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. These positions are used to initialize `atom` objects.
  - `spherical_void`() (**A**): Removes particles inside a sphere with a specific center position and radius.
  - `cylindrical_void`() (**A**): Removes particles inside a cylinder with a specific center position, radius and height.
  - `sphere`() (**A**): The "opposite" of `spherical_void`(). Cuts the crystal to a sphere, can leave atomic layers of dynamical particles on top of the fixed ones. Preserves charge neutrality.

  - `remove_particle`(): Used by the particle removal functions to remove pointers to a particle and assure that all particle indices stay correct.

- `class 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, like $\|\mathbf{k}\|^2$.
  - `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 ions which experience PBC.
  - `update_structure_factor`() (**C**): Corrects the 2D structure factor of the fixed particles by multiplying it with $\exp(i\mathbf{k} \cdot \Delta\mathbf{r} + ih\Delta x)$ for constant fixed particle displacements, as described in section 4.2.3.

A.2 Utility classes

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:

- `class complex`: Custom class for complex number arithmetic. Works in the same fashion as the C++ standard complex type.
- `class parameters`: Class for reading parameters from a file. Reads both integers and floating-point numbers. The parameters must be sorted in a predefined order.
- `class random_generator`: Generation of uniformly and normally distributed random numbers.
- `class vtkwriter`: Class for output of 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.

A.3 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.

Some parameters used in the program were only modified during the testing process until I found a suitable value. These are defined as constants in the code

and are not adjustable outside the compiled code. Examples are the number of measurement bins for contact stress and 3D field values. The h integration interval and step size from section 4.2.2 are never changed, but take the values given in section 4.2.3. The heat conduction cutoff point from section 5.1.3 is always set to 20% of the system size in the x direction L_x. As L_x may vary, this automatic parameter choice is highly unphysical, but is expected to give no significant change in simulation dynamics.

Following is a list of what the parameters file contains. Every vector component parameter ending with `_x` has corresponding parameters ending with `_y` and `_z` for the other two vector components.

A.3.1 Global parameters

- `system_id` : Flag for specifying material. 1: Argon (for testing only), 2: Sodium chloride, 3: Silicon.
- `N_x` : The number of conventional crystal unit cells in each direction.
- `PBC_x` : PBC flags deciding whether the dynamical atoms should experience PBC in each of the directions.
- `fixed_flag` : Flag for the existence of fixed particles at the x-sides. 0: No fixed particles, 1: Left side only, 2: Right side only, 3: Both sides.
- `fixed_layers` : The number of fixed particle unit cell layers.
- `density` : Initial particle density [Å^{-3}]. Used to set the lattice constant.
- `T_init` : Initial temperature [K]. Used in the initial Maxwell-Boltzmann velocity distribution.
- `tau` : Relaxation time parameter τ for the Andersen and BDP thermostats [fs].
- `dt` : Time integration step length Δt [fs].
- `tspvf` : Time steps per visualized frame.
- `tsptr` : Time steps per temperature rescaling.
- `tspnlu` : Time steps per neighbour list update.
- `lj_cutoff` : Minimal neighbour list cutoff and LJ interaction cutoff [Å].
- `ewald_constant` : Proportionality constant C for the Gaussian smearing parameter α [Å^{-1}]. See section 4.2.3.
- `accuracy` : The round off error in the Coloumb force and potential calculations will be $\epsilon = 10^{-\texttt{accuracy}}$. See section 4.2.3.
- `gap_position` : A cuboidal gap at $x =$ `gap_position` is inserted at simulation start [Å]. Not used if negative.
- `gap_width` : The width of the inserted gap [Å].

- `void_size` : The radius of the spherical or cylindrical void removing atoms in the middle of the simulation box [Å]. Not used if negative.
- `sphere_flag` : Determines the sphere cutting mode. 0: Cuboidal lattice, 1: Spherical cut, radius $\frac{1}{2}L_x$, 2: Sphere segment cut, radius $\frac{1}{2}L_y$.
- `phases` : The number of simulation phases. The phase-specific parameters must have this many values each.

A.3.2 Phase-specific parameters

- `timesteps` : Number of time steps for one phase.
- `T_bath` : Heat bath temperature [K]. Used with a thermostat.
- `temp_flag` : Temperature adjustment flag. 0: No adjustment (microcanonical ensemble), 1: Global BDP thermostat adjustment, 2: x-side BDP thermostat adjustment, 3: Andersen thermostat adjustment.
- `dissipation` : Dissipative force parameter $\gamma = -F/v$ [eVfs/Å^2]. Not used if negative.
- `expand_rate_x` : The rate of constant volume expansion in the x direction [Å/fs].
- `left_displace_lattice` : The y direction speed of the fixed particles on the left x-side of the simulation cell [Å/fs].
- `right_displace_lattice` : The y direction speed of the fixed particles on the right x-side of the simulation cell [Å/fs].
- `force_x` : Constant force in the x direction on upper sphere particles [eV/Å].
- `vcoll_x` : The collective velocity in the x direction will be adjusted to this number each time step, unless the number if negative [Å/fs].
- `target_load` : The desired normal force in friction experiments [eV/Å].
- `beta` : Parameter for tuning the speed of fixed particle movement when aiming for a target normal force. The feature is turned off with a negative value.
- `output_level` : Flag for output levels. 0: No output, 1: Macroscopic quantities (MQ) only, 2: MQ and `.vtk` positions, 3: MQ, `.vtk` positions, and `.vtk` field values, 4: MQ and `.xyz` positions.

A.3.3 Parameter file example

The following example of a parameters file is used for two of the data points in Fig. 8.10(b). The system starts out as a $3 \times 7 \times 7$ unit cell cuboid of dynamical NaCl ions with $2 \times 7 \times 7$ unit cell cuboids of fixed ions on each x-side. The dynamical ions are cut to a sphere segment, and a 4.0 Å gap between it and the lower fixed particle layers are created. Phases of equilibration, indentation, re-equilibration and separation follow.

```
system_id               2

N_x                     3
N_y                     7
N_z                     7
PBC_x                   0
PBC_y                   0
PBC_z                   0
fixed_flag              3
fixed_layers            2

density                 0.045
T_init                  300.0
tau                     20.0

dt                      4.0
tspvf                   50
tsptr                   1
tspnlu                  10

lj_cutoff               12.0
ewald_constant          2.0
accuracy                5

gap_position            0.0
gap_width               4.0
void_size               0.0
sphere_flag             2

phases                  4

timesteps               2000    8000    2000    16000
T_bath                  300.0   300.0   300.0   300.0
temp_flag               1       2       2       2
dissipation             -1.0    -1.0    -1.0    -1.0

expand_rate_x           0.0     -0.0002 0.0     0.0002
expand_rate_y           0.0     0.0     0.0     0.0
expand_rate_z           0.0     0.0     0.0     0.0
left_displace_rate      0.0     0.0     0.0     0.0
right_displace_rate     0.0     0.0     0.0     0.0
force_x                 0.0     0.0     0.0     0.0
force_y                 0.0     0.0     0.0     0.0
force_z                 0.0     0.0     0.0     0.0
vcoll_x                 -1.0    -1.0    -1.0    -1.0
vcoll_y                 -1.0    -1.0    -1.0    -1.0
vcoll_z                 -1.0    -1.0    -1.0    -1.0
target_load             0.0     0.0     0.0     0.0
beta                    -1.0    -1.0    -1.0    -1.0

output_level            2       2       2       2
```

Bibliography

[1] B.N.J.Persson, *Sliding Friction*. Springer, 1998.

[2] T. L. Anderson, *Fracture mechanics*. Boston, Mass, USA: CRC Press, 2nd ed., 1995.

[3] Z. G. Wang, U. Landman, R. L. B. Selinger, and W.-M. Gelbart, "Molecular-dynamics study of elasticity and failure of ideal solids," *Phys. Rev. B*, vol. 44, pp. 378–381, July 1991.

[4] M. Buehler, A. Hartmaier, M. Duchaineau, F. Abraham, and H. Gao, "The dynamical complexity of work-hardening: a large-scale molecular dynamics simulation," *Acta Mechanica Sinica*, vol. 21, pp. 103–111, APR 2005.

[5] P. Vashishta, R. K. Kalia, A. Nakano, E. Kaxiras, A. Grama, G. Lu, S. Eidenbenz, A. F. Voter, R. Q. Hood, J. A. Moriarty, and L. H. Yang, "Hierarchical petascale simulation framework for stress corrosion cracking," in *SciDac 2007*, vol. 78, (Dirac House, Temple Back, Bristol BS1 6BE, England), pp. U288–U294, US DOE Off Sci; Natl Nucl Security Adm; US NSF, IOP publishing ltd., 2007.

[6] D. Holland and M. Marder, "Ideal brittle fracture of silicon studied with molecular dynamics," *Phys. Rev.*, vol. 81, p. 4029, NOV 1998.

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

[8] 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.

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

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

[11] C. Yang and B. N. J. Persson, "Contact mechanics: contact area and interfacial separation from small contact to full contact," *Journal of Physics: Condensed Matter*, vol. 20, no. 21, p. 215214, 2008.

[12] J. S. Coursey, D. J. Schwab, and R. A. Dragoset, "Atomic weights and isitopic compositions." http://physics.nist.gov/PhysRefData/Compositions/.

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

[14] D. Frenkel and B. Smit, *Understanding Molecular Simulation, From Algorithms to Applications*. Academic Press, 2nd ed., November 2001.

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

[16] D. Schroeder, *An Introduction to Thermal Physics*. Addison Wesley Longman, August 2005.

[17] G. Bussi, D. Donadio, and M. Parrinello, "Canonical sampling through velocity rescaling," *Chem. Phys.*, vol. 126, no. 1, p. 014101, 2007.

[18] D. Griffiths, *Introduction to Quantum Mechanics*. Pearson Prentice Hall, 2nd ed., 2005.

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

[20] 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.

[21] 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.

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

[23] 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.

[24] T. Matthey, "Plain Ewald and PME." http://protomol.sourceforge.net/, May 2005.

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

[26] 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.

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

[28] 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*, vol. 20, no. 10, pp. 2733–2742, 1987.

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

[30] Division of Chemical Education, Purdue University, "Lattice energy." http://chemed.chem.purdue.edu/genchem/topicreview/.

[31] 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.

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