A molecular dynamics primer

Furio Ercolessi

International School for Advanced Studies (SISSA-ISAS)

Trieste, Italy

Email: `furio@sissa.it`
WWW: `http://www.sissa.it/furio/`

Spring College in Computational Physics, ICTP, Trieste, June 1997.

Contents

Preface

These notes constitute the skeleton for a short *hands-on* course on molecular dynamics. It is assumed that the reader has basic notions of classical mechanics, statistical mechanics and thermodynamics, but knows very little—if anything at all—on methods for atomistic computer simulation. The basics of molecular dynamics are explained, together with a few tricks of the trade.

No attempt is made to address "advanced topics" in molecular dynamics, such as calculation of free energies, non equilibrium simulations, etc., and many important subjects are treated only superficially. This document is meant to be just a starting point. Many excellent books devoted to computer simulations in condensed matter physics are available [1, 2, 3, 4, 5, 6, 7], and the reader is advised to make use of them to dig into more specific subjects.

These note are accompanied by example programs written in Fortran 90 and available on the World Wide Web at the URL `http://www.sissa.it/furio/md/f90`. While still relatively uncommon among scientists, Fortran 90 is bound to become one of the most important languages for scientific programming in the years to come. Fortran 90 incorporates Fortran 77—allowing it to inherit smoothly the legacy of the immense body of existing codes—but also features elements now recognized to be essential in a language for large projects in scientific computing.

I hope that the examples contained here will be of help to young computational physicists to get familiar with Fortran 90, and to people grown with Fortran 77 (or 66!) to migrate towards this new, more modern environment.

Any comment or suggestion to improve this document, which will also be kept online at the URL `http://www.sissa.it/furio/md/`, will be welcome and appreciated. The best way to contact the author is by email at `furio@sissa.it`.

Chapter 1

Introduction

1.1 The role of computer experiments

Computer experiments play a very important role in science today. In the past, physical sciences were characterized by an interplay between experiment and theory. In experiment, a system is subjected to measurements, and results, expressed in numeric form, are obtained. In theory, a model of the system is constructed, usually in the form of a set of mathematical equations. The model is then validated by its ability to describe the system behavior in a few selected cases, simple enough to allow a solution to be computed from the equations. In many cases, this implies a considerable amount of simplification in order to eliminate all the complexities invariably associated with real world problems, and make the problem solvable.

In the past, theoretical models could be easily tested only in a few simple "special circumstances". So, for instance, in condensed matter physics a model for intermolecular forces in a specific material could be verified in a diatomic molecule, or in a perfect, infinite crystal. Even then, approximations were often required to carry out the calculation. Unfortunately, many physical problems of extreme interest (both academic and practical) fall outside the realm of these "special circumstances". Among them, one could mention the physics and chemistry of defects, surfaces, clusters of atoms, organic molecules, involving a large amount of degrees of freedom; an accurate treatment of temperature effects, including anharmonicities and phase transitions; disordered systems in general, where symmetry is of no help to simplify the treatment; and so on.

The advent of high speed computers—which started to be used in the 50s—altered the picture by inserting a new element right in between experiment and theory: the *computer experiment*. In a computer experiment, a model is still provided by theorists, but the calculations are carried out by the machine by following a "recipe" (the *algorithm*, implemented in a suitable *programming language*). In this way, complexity can be introduced (still with caution!) and more realistic systems can be investigated, opening a road towards a better understanding of real experiments.

Needless to say, the development of computer experiments altered substantially the traditional relationship between theory and experiment. On one side, computer simulations increased the demand for accuracy of the models. For instance, a molecular dynamics simulation allows to evaluate the melting temperature of a material, modeled by means of a certain interaction law. This is a difficult test for the theoretical model to pass—and a test which has not been available in the past. Therefore, simulation "brings to life" the models, disclosing critical areas and providing suggestions to improve them.

On the other side, simulation can often come very close to experimental conditions, to the extent that computer results can sometimes be compared directly with experimental results. When this happens, simulation becomes an extremely powerful tool not only to understand and interpret the experiments at the microscopic level, but also to study regions which are not accessible experimentally, or which would imply very expensive experiments, such as under extremely high pressure.

Last but not least, computer simulations allow *thought experiments*—things which are just impossible to do in reality, but whose outcome greatly increases our understanding of phenomena—to be realized. Fantasy and creativity are important qualities for the computer simulator!

1.1.1 But is it theory or experiment?

Simulation is seen sometimes as theory, sometimes as experiment. On one side, we are still dealing with models, not with the "real thing": this suggests to classify simulation as belonging to theoretical methods without hesitation. On the other side, the procedure of verifying a model by computer simulation resembles an experiment quite closely: we perform a run, and then analyze the results in pretty much the same way as experimental physicists do. So how should we classify simulation?

There is no sharp answer to this question: both sides represent legitimate point of views, and this is what makes computational science a branch on its own. There is however another important consideration.

Theory is traditionally based on the *reductionistic* approach: we deal with complexity by reducing a system to simpler subsystems, continuing until the subsystems are simple enough to be represented with solvable models. When we look at simulation as simply a practical tool to "verify and test" models in situations which are too complex to handle analytically (for example, when computing the phase diagram of a substance modeled by a certain force law), we are implicitly assuming that the model represents the "theory level" where the interest is focused.

But it is important to realize that simulation may play a more important and interesting role. We can consider it not as an aid to reductionism but—to some extent—as an *alternative* to it. Simulation increases the threshold of complexity which separates "solvable" and "unsolvable" models. We can take advantage of this threshold shift and move up one level in our description of physical systems. Thanks to the presence of simulation, we do not need to work with models as simple as those used in the past. This gives us an additional degree of freedom to explore and opens entirely new possibilities.

One example of this point of view is represented by interatomic potentials.

In the past, interactions were obtained by two-body potentials with simple analytical form, such as Morse or Lennard-Jones. Today, the most accurate potentials contain many-body terms and are *determined numerically* by reproducing as closely as possible forces predicted by *first-principle* methods (this is discussed in 4.8). We have thus moved up one level in the degree of reductionism contained in the potential, now limited only to the selection of its analytical form. The advantage is of course a much better realism, which in turn allows investigation of physics problem which require a level of accuracy in the model that was not achieved before. These new potentials could not exist without simulation: simulation is not only a link between experiment and theory, it is also a poweful tool to propel progress in new directions.

The reader interested in these "philosophical" aspects of computational science can find a very enjoyable discussion in chapter 1 of ref. [6].

1.2 What is molecular dynamics?

We call *molecular dynamics* (MD) a computer simulation technique where the time evolution of a set of interacting atoms is followed by integrating their equations of motion.

In molecular dynamics we follow the laws of classical mechanics, and most notably Newton's law:

$$\mathbf{F}_i = m_i \mathbf{a}_i \tag{1.1}$$

for each atom i in a system constituted by N atoms. Here, m_i is the atom mass, $\mathbf{a}_i = d^2\mathbf{r}_i/dt^2$ its acceleration, and $\mathbf{F}_i$ the force acting upon it, due to the interactions with other atoms. Therefore, in contrast with the Monte Carlo method, molecular dynamics is a deterministic technique: given an initial set of positions and velocities, the subsequent time evolution is *in principle*[1] completely determined. In more pictorial terms, atoms will "move" into the computer, bumping into each other, wandering around (if the system is fluid), oscillating in waves in concert with their neighbors, perhaps evaporating away from the system if there is a free surface, and so on, in a way pretty similar to what atoms in a real substance would do.

The computer calculates a trajectory in a $6N$-dimensional phase space ($3N$ positions and $3N$ momenta). However, such trajectory is usually not particularly relevant by itself. *Molecular dynamics is a statistical mechanics method.* Like Monte Carlo, it is a way to obtain a set of configurations distributed according to some statistical distribution function, or statistical ensemble. An example is the microcanonical ensemble, corresponding to a probability density in phase space where the total energy is a constant E:

$$\delta(H(\Gamma) - E).$$

Here, $H(\Gamma)$ is the Hamiltonian, and Γ represents the set of positions and momenta. δ is the Dirac function, selecting out only those states which have a

[1]In practice, the finiteness of the integration time step and arithmetic rounding errors will eventually cause the computed trajectory to deviate from the true trajectory.

specific energy E. Another example is the canonical ensemble, where the temperature T is constant and the probability density is the Boltzmann function

$$\exp(-H(\Gamma)/k_B T).$$

According to statistical physics, physical quantities are represented by averages over configurations distributed according to a certain statistical ensemble. A trajectory obtained by molecular dynamics provides such a set of configurations. Therefore, a measurements of a physical quantity by simulation is simply obtained as an arithmetic average of the various instantaneous values assumed by that quantity during the MD run.

Statistical physics is the link between the microscopic behavior and thermodynamics. In the limit of very long simulation times, one could expect the phase space to be fully sampled, and in that limit this averaging process would yield the thermodynamic properties. In practice, the runs are always of finite length, and one should exert caution to estimate when the sampling may be good ("system at equilibrium") or not. In this way, MD simulations can be used to measure thermodynamic properties and therefore evaluate, say, the phase diagram of a specific material.

Beyond this "traditional" use, MD is nowadays also used for other purposes, such as studies of non-equilibrium processes, and as an efficient tool for optimization of structures overcoming local energy minima (*simulated annealing*).

1.3 Historical notes

A full account of early developments of the molecular dynamics technique is certainly beyond the scope of this document. I will simply mention below a few key papers appeared in the 50s and in the 60s that can be regarded as milestones in molecular dynamics, and whose reading is certainly recommended. I will not mention developments in the Monte Carlo method, which also occurred during the same period of time. The interested reader can find a first-hand account by Wood in [9].

Reprints of all these papers, and of many other important works in the area of computer simulation of liquids and solids published up to 1986, can be found in ref. [8].

- The first paper reporting a molecular dynamics simulation was written by Alder and Wainwright [10] in 1957. The purpose of the paper was to investigate the phase diagram of a hard sphere system, and in particular the solid and liquid regions. In a hard sphere system, particles interact via instantaneous collisions, and travel as free particles between collisions. The calculations were performed on a UNIVAC and on an IBM 704.

- The article *Dynamics of radiation damage* by J. B. Gibson, A. N. Goland, M. Milgram and G. H. Vineyard from Brookhaven National Laboratory, appeared in 1960 [11], is probably the first example of a molecular dynamics calculation with a continuous potential based on a finite difference time integration method. The calculation for a 500-atoms system was performed on an IBM 704, and took about a minute per time step. The paper, dealing with creation of defects induced by radiation damage (a theme appropriate to *cold war* days), is done exceedingly well, and is hard to believe that it is almost 40 years old!

- Aneesur Rahman at Argonne National Laboratory has been a well known pioneer of molecular dynamics. In his famous 1964 paper *Correlations in the motion of atoms in liquid argon* [12], he studies a number of properties of liquid Ar, using the Lennard-Jones potential on a system containing 864 atoms and a CDC 3600 computer. The legacy of Rahman's computer codes is still carried by many molecular dynamics programs in operation around the world, descendant of Rahman's.

- Loup Verlet calculated in 1967 [13] the phase diagram of argon using the Lennard-Jones potential, and computed correlation functions to test theories of the liquid state. The bookkeeping device which became known as *Verlet neighbor list* was introduced in these papers. Moreover the "Verlet time integration algorithm" (2.3.1) was used. Phase transitions in the same system were investigated by Hansen and Verlet a couple of years later [14].

1.4 Today's role of molecular dynamics

Reviewing even a tiny subset of the applications of molecular dynamics is far beyond the purpose of these notes. I will only briefly mention a few areas of current interest where MD has brought and/or could bring important contributions. The list should not be considered as exhaustive, and certainly reflects my background in condensed matter physics:

Liquids. As shown in 1.3, liquids are where everything started! Nevertheless, they remain an important subject. Availability of new realistic interaction models allows to study new systems, elemental and multicomponent. Through non-equilibrium techniques, transport phenomena such as viscosity and heat flow have been investigated [2].

Defects. Another subject whose investigation by MD started a long time ago (see [8]), defects in crystals—crucial for their mechanical properties and therefore of technological interest—remain a favoured topic. The focus shifted perhaps from point defects (vacancies, interstitials) to linear (dislocations) and planar (grain boundaries, stacking faults) defects [15]. Again, improved realism thanks to better potentials constitutes a driving force.

Fracture. Under mechanical action, solids break into two or more pieces. The fracture process can occur in different ways and with different speeds depending of several parameters. The technological importance is obvious, and simulation is providing insight.

Surfaces. Surface physics had a boom starting in the 80s, thanks to the availability of new wonderful experimental tools with microscopic resolution (scanning tunneling microscopy, high resolution electron microscopy, several scattering-based techniques). Simulation is still playing a big role in understanding phenomena such as surface reconstructions, surface melting, faceting, surface diffusion, roughening, etc, often requiring large samples and simulation times.

Friction. Even more recent are investigations of adhesion and friction between two solids, propelled by the development of the atomic force microscope (AFM). The body of "macroscopic" knowledge is being revised and expanded on microscopic grounds.

Clusters. Clusters—conglomerates of a number of atoms ranging from a few to several thousands—constitute a bridge between molecular systems and solids, and exhibit challenging features. Frequently, an astonishingly large number of different configurations have very similar energies, making it difficult to determine stable structures. Their melting properties can also be significantly different from those of the solid, due to the finite size, the presence of surfaces and the anisotropy. Metal clusters are extremely important from the technological point of view, due to their role as catalysts in important chemical reactions (for instance, in catalytic exhaust pipes of cars).

Biomolecules. MD allows to study the dynamics of large macromolecules, including biological systems such as proteins, nucleic acids (DNA, RNA), membranes [16]. Dynamical events may play a key role in controlling processes which affect functional properties of the biomolecule. *Drug design* is commonly used in the pharmaceutical industry to test properties of a molecule at the computer without the need to synthesize it (which is far more expensive).

Electronic properties and dynamics. The development of the Car-Parrinello method (see 1.5.2), where the forces on atoms are obtained by solving the electronic structure problem instead of by an interatomic potential, allows to study electronic properties of materials fully including their dynamics (and, therefore, phase transitions and other temperature-dependent phenomena). This important work gave rise to a very successful research line during the last decade.

1.5 Limitations

Molecular dynamics is a very powerful technique but has—of course—limitations. We quickly examine the most important of them.

1.5.1 Use of classical forces

One could immediately ask: how can we use Newton's law to move atoms, when everybody knows that systems at the atomistic level obey quantum laws rather than classical laws, and that Schrödinger's equation is the one to be followed?

A simple test of the validity of the classical approximation is based on the de Broglie thermal wavelength [17], defined as

$$\Lambda = \sqrt{\frac{2\pi\hbar^2}{Mk_BT}} \tag{1.2}$$

where M is the atomic mass and T the temperature. The classical approximation is justified if $\Lambda \ll a$, where a is the mean nearest neighbor separation. If one considers for instance liquids at the triple point, Λ/a is of the order of 0.1 for light elements such as Li and Ar, decreasing further for heavier elements. The classical approximation is poor for very light systems such as H_2, He, Ne.

Moreover, quantum effects become important in any system when T is sufficiently low. The drop in the specific heat of crystals below the Debye temperature [18], or the anomalous behavior of the thermal expansion coefficient, are well known examples of measurable quantum effects in solids.

Molecular dynamics results should be interpreted with caution in these regions.

1.5.2 Realism of forces

How *realistic* is a molecular dynamics simulation?

In molecular dynamics, atoms interact with each other. These interactions originate forces which act upon atoms, and atoms move under the action of these instantaneous forces. As the atoms move, their relative positions change and forces change as well.

The essential ingredient containing the physics is therefore constituted by the forces. A simulation is realistic—that is, it mimics the behavior of the real system—only to the extent that interatomic forces are similar to those that real atoms (or, more exactly, nuclei) would experience when arranged in the same configuration.

As described in 2.1, forces are usually obtained as the gradient of a *potential energy function*, depending on the positions of the particles. The realism of the simulation therefore depends on the ability of the potential chosen to reproduce the behavior of the material under the conditions at which the simulation is run.

The problem of selecting—or constructing—potentials is addressed in more detail in chapter 4.

1.5.3 Time and size limitations

Typical MD simulations can be performed on systems containing thousands—or, perhaps, millions—of atoms, and for simulation times ranging from a few picoseconds to hundreds of nanoseconds. While these numbers are certainly respectable, it may happen to run into conditions where time and/or size limitations become important.

A simulation is "safe" from the point of view of its duration when the simulation time is much longer than the relaxation time of the quantities we are interested in. However, different properties have different relaxation times.

In particular, systems tend to become slow and sluggish in the proximity of phase transitions, and it is not uncommon to find cases where the relaxation time of a physical property is orders of magnitude larger than times achievable by simulation.

A limited system size can also constitute a problem. In this case one has to compare the size of the MD cell with the correlation lengths of the spatial correlation functions of interest. Again, correlation lengths may increase or even diverge in proximity of phase transitions, and the results are no longer reliable when they become comparable with the box length.

This problem can be partially alleviated by a method known as *finite size scaling*. This consist of computing a physical property A using several box with different sizes L, and then fitting the results on a relation

$$A(L) = A_\circ + \frac{c}{L^n}$$

using $A_\circ, c, n$ as fitting parameters. $A_\circ$ then corresponds to $\lim_{L\to\infty} A(L)$, and should therefore be taken as the most reliable estimate for the "true" physical quantity.

Chapter 2

The basic machinery

2.1 Modeling the physical system

The main ingredient of a simulation is a model for the physical system. For a molecular dynamics simulation this amounts to choosing the *potential*: a function $V(\mathbf{r}_1, \ldots, \mathbf{r}_N)$ of the positions of the nuclei, representing the potential energy of the system when the atoms are arranged in that specific configuration. This function is translationally and rotationally invariant, and is usually constructed from the *relative* positions of the atoms with respect to each other, rather than from the absolute positions.

Forces are then derived as the gradients of the potential with respect to atomic displacements:

$$\mathbf{F}_i = -\nabla_{\mathbf{r}_i} V(\mathbf{r}_1, \ldots, \mathbf{r}_N) \tag{2.1}$$

This form implies the presence of a conservation law of the total energy $E = K + V$, where K is the instantaneous kinetic energy.

The simplest choice for V is to write it as a sum of pairwise interactions:

$$V(\mathbf{r}_1, \ldots, \mathbf{r}_N) = \sum_i \sum_{j>i} \phi(|\mathbf{r}_i - \mathbf{r}_j|) \tag{2.2}$$

The clause $j > i$ in the second summation has the purpose of considering each atom pair only once. In the past most potentials were constituted by pairwise interactions, but this is no longer the case. It has been recognized that the two-body approximation is very poor for many relevant systems, such as metals and semiconductors. Various kinds of many-body potentials are now of common use in condensed matter simulation, and will be briefly reviewed in §4.4 and §4.5.

The development of accurate potentials represents an important research line. A quick overview of the current situation will be presented in Chapter 4. For now, let us mention only the most commonly used interaction model: the *Lennard-Jones pair potential.*

2.1.1 The Lennard-Jones potential

The Lennard-Jones 12-6 potential is given by the expression

$$\phi_{\mathrm{LJ}}(r) = 4\varepsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] \tag{2.3}$$

for the interaction potential between a pair of atoms. The total potential of a system containing many atoms is then given by 2.2.

This potential has an attractive tail at large r, it reaches a minimum around $1.122\,\sigma$, and it is strongly repulsive at shorter distance, passing through 0 at $r = \sigma$ and increasing steeply as r is decreased further.

The term $\sim 1/r^{12}$, dominating at short distance, models the repulsion between atoms when they are brought very close to each other. Its physical origin is related to the Pauli principle: when the electronic clouds surrounding the atoms starts to overlap, the energy of the system increases abruptly. The exponent 12 was chosen exclusively on a practical basis: equation (2.3) is particularly easy to compute. In fact, on physical grounds an exponential behavior would be more appropriate.

The term $\sim 1/r^6$, dominating at large distance, constitute the attractive part. This is the term which gives cohesion to the system. A $1/r^6$ attraction is originated by van der Waals dispersion forces, originated by dipole-dipole interactions in turn due to fluctuating dipoles. These are rather weak interactions, which however dominate the bonding character of closed-shell systems, that is, rare gases such as Ar or Kr. Therefore, these are the materials that a LJ potential could mimic fairly well[1]. The parameters ε and σ are chosen to fit the physical properties of the material.

On the other hand, a LJ potential is not at all adequate to model situations with open shells, where strong localized bonds may form (as in covalent systems), or where there is a delocalized "electron sea" where the ions sit (as in metals). In these systems the two-body interactions scheme itself fails very badly. Potentials for these systems will be discussed in chapter 4.

However, regardless of how well it is able to model actual materials, the LJ 12-6 potential constitutes nowadays an extremely important *model system*. There is a vast body of papers who investigated the behavior of atoms interacting via LJ on a variety of different geometries (solids, liquids, surfaces, clusters, two-dimensional systems, etc). One could say that LJ is the *standard potential* to use for all the investigations where the focus is on fundamental issues, rather than studying the properties of a specific material. The simulation work done on LJ systems helped us (and still does) to understand basic points in many areas of condensed matter physics, and for this reason the importance of LJ cannot be underestimated.

When using the LJ potentials in simulation, it is customary to work in a system of units where $\sigma = 1$ and $\varepsilon = 1$. The example codes accompanying these notes follow this convention.

2.1.2 Potential truncation and long-range corrections

The potential in eq. (2.3) has an infinite range. In practical applications, it is customary to establish a cutoff radius R_c and disregard the interactions between atoms separated by more than R_c. This results in simpler programs and

[1]Many-body effects in the interaction are however present also in these systems and, in all cases, pair potentials more accurate than LJ have been developed for rare gases.

enormous savings of computer resources, because the number of atomic pairs separated by a distance r grows as r^2 and becomes quickly huge.

A simple truncation of the potential creates a new problem though: whenever a particle pair "crosses" the cutoff distance, the energy makes a little jump. A large number of these events is likely to spoil energy conservation in a simulation. To avoid this problem, the potential is often shifted in order to vanish at the cutoff radius [2]:

$$V(r) = \begin{cases} \phi_{\rm LJ}(r) - \phi_{\rm LJ}(R_c) & \text{if } r \le R_c \\ 0 & \text{if } r > R_c \end{cases} \tag{2.4}$$

Physical quantities are of course affected by the potential truncation. The effects of truncating a full-ranged potential can be approximately estimated by treating the system as a uniform (constant density) continuum beyond R_c. For a bulk system (periodic boundary conditions along each direction, see §2.2), this usually amounts to a constant additive correction. For example, the potential tail (attractive) brings a small additional contribution to the cohesive energy, and to the total pressure. Truncation effects are not so easy to estimate for geometries with free surfaces, due to the lower symmetry, and can be rather large for quantities like surface energy.

Commonly used truncation radii for the Lennard-Jones potential are $2.5\,\sigma$ and $3.2\,\sigma$. It should be mentioned that these truncated Lennard-Jones models are so popular that they acquired a value on their own as reference models for generic two-body systems (as also discussed at the end of the previous section). In many cases, there is no interest in evaluating truncation corrections because the truncated model itself is the subject of the investigation.

Potentials for metals and semiconductors are usually designed from the start with a cutoff radius in mind, and go usually to zero at R_c together with their first two derivatives at least. Such potentials do not therefore contain true van der Waals forces. Since such forces are much weaker than those associated with metallic or covalent bonding, this is usually not a serious problem except for geometries with two or more separate bodies.

2.2 Periodic boundary conditions

What should we do at the boundaries of our simulated system? One possibility is doing nothing special: the system simply terminates, and atoms near the boundary would have less neighbors than atoms inside. In other words, the sample would be surrounded by surfaces.

Unless we really want to simulate a cluster of atoms, this situation is not realistic. No matter how large the simulated system is, its number of atoms N would be negligible compared with the number of atoms contained in a macroscopic piece of matter (of the order of 10^{23}), and the ratio between the number of surface atoms and the total number of atoms would be much larger

[2]Shifting eliminates the energy discontinuity, but not the force discontinuity. Some researchers altered the form of the potential near R_c to obtain a smoother junction, but there is no standard way to do that.

than in reality, causing surface effects to be much more important than what they should.

A solution to this problem is to use *periodic boundary conditions* (PBC). When using PBC, particles are enclosed in a box, and we can imagine that this box is replicated to infinity by rigid translation in all the three cartesian directions, completely filling the space. In other words, if one of our particles is located at position $\mathbf{r}$ in the box, we assume that this particle really represents an infinite set of particles located at

$$\mathbf{r} + \ell\mathbf{a} + m\mathbf{b} + n\mathbf{c} \;\; , \;\; (\ell, m, n = -\infty, \infty),$$

where ℓ, m, n are integer numbers, and I have indicated with $\mathbf{a}, \mathbf{b}, \mathbf{c}$ the vectors corresponding to the edges of the box. All these "image" particles move together, and in fact only one of them is represented in the computer program.

The key point is that now each particle i in the box should be thought as interacting not only with other particles j in the box, but also with their images in nearby boxes. That is, interactions can "go through" box boundaries. In fact, one can easily see that *(a)* we have virtually eliminated surface effects from our system, and *(b)* the position of the box boundaries has no effect (that is, a translation of the box with respect to the particles leaves the forces unchanged).

Apparently, the number of interacting pairs increases enormously as an effect of PBC. In practice, this is not true because potentials usually have a short interaction range. The *minimum image criterion* discussed next simplifies things further, reducing to a minimum the level of additional complexity introduced in a program by the use of PBC.

2.2.1 The minimum image criterion

Let us suppose that we are using a potential with a finite range: when separated by a distance equal or larger than a cutoff distance R_c, two particles do not interact with each other. Let us also suppose that we are using a box whose size is larger than $2R_c$ along each cartesian direction.

When these conditions are satisfied, it is obvious that *at most one* among all the pairs formed by a particle i in the box and the set of all the periodic images of another particle j will interact.

To demonstrate this, let us suppose that i interacts with two images *j1* and *j2* of j. Two images must be separated by the translation vector bringing one box into another, and whose length is at least $2R_c$ by hypothesis. In order to interact with both *j1* and *j2*, i should be within a distance R_c from each of them, which is impossible since they are separated by more than $2R_c$.

When we are in these conditions, we can safely use is the *minimum image criterion*: among all possible images of a particle j, select the closest, and throw away all the others. In fact only the closest is a candidate to interact; all the others certainly do not.

This operating conditions greatly simplify the set up of a MD program, and are commonly used. Of course, one must always make sure that the box size is at least $2R_c$ along all the directions where PBCs are in effect.

2.2.2 Geometries with surfaces

The purpose of PBCs is to eliminate surface effects. However, we may also be interested in situations where we *want to have surfaces*. Obviously, all surface physics problems belong to this class!

For a surface simulation, the model usually adopted is that of the *slab*: a thick slice of material, delimited by two free surfaces. This is simply obtained by removing PBCs along one direction (usually taken to be z) while retaining them in the orthogonal plane. Therefore, a slab must be thought as replicated to infinity in the xy plane, but there is no replication along the slab normal z.

If the slab is thick enough, its inner part is expected to be quite similar to the bulk of the material. The two surfaces (at the top and at the bottom of the slab) can then be thought of as two decoupled, independent surfaces. With this assumption, the system behavior would be close to that of a single surface at the top of a semiinfinite system—a condition which would be closer to actual surface experiments but which is unfortunately impossible to realize in simulation.

There are also circumstances where researchers find it preferable to "freeze" a few layers of material at one side of the slab—with the atoms constrained to sit at perfect bulk-like crystal positions—leaving only the other side free. This is done when it is believed that spurious effects induced by the frozen side into the "good" side of the slab are of smaller entity than the effects induced by another free surface at the same separation distance. This is to be preferred in those cases where massive perturbations occur at the surface, such as atomic rearrangements (surface reconstructions) or local melting. If a slab with a frozen side is chosen, care must be taken to freeze a number of layers corresponding to a thickness of at least R_c, to guarantee that no mobile atom would be able to "see" the surface "through" the fixed atoms.

One can also leave PBCs along only one direction only, thus obtaining a *wire* geometry with the wire axis along the PBC direction, or remove them altogether. This latter condition correspond to a *cluster* of atoms. Clusters are important systems and simulation of clusters have been performed since the early days of molecular dynamics and Monte Carlo.

2.3 Time integration algorithm

The engine of a molecular dynamics program is its time integration algorithm, required to integrate the equation of motion of the interacting particles and follow their trajectory.

Time integration algorithms are based on *finite difference methods*, where time is discretized on a finite grid, the *time step* Δt being the distance between consecutive points on the grid. Knowing the positions and some of their time derivatives at time t (the exact details depend on the type of algorithm), the integration scheme gives the same quantities at a later time $t + \Delta t$. By iterating the procedure, the time evolution of the system can be followed for long times.

Of course, these schemes are approximate and there are errors associated with them. In particular, one can distinguish between

- *Truncation errors*, related to the accuracy of the finite difference method with respect to the true solution. Finite difference methods are usually based on a Taylor expansion truncated at some term, hence the name. These errors do not depend on the implementation: they are intrinsic to the algorithm.

- *Round-off errors*, related to errors associated to a particular implementation of the algorithm. For instance, to the finite number of digits used in computer arithmetics.

Both errors can be reduced by decreasing Δt. For large Δt, truncation errors dominate, but they decrease quickly as Δt is decreased. For instance, the Verlet algorithm discussed in §2.3.1 has a truncation error proportional to Δt^4 for each integration time step. Round-off errors decrease more slowly with decreasing Δt, and dominate in the small Δt limit. Using 64-bit precision (corresponding to "double precision" when using Fortran on the majority of today's workstations) helps to keep round-off errors at a minimum.

Two popular integration methods for MD calculations are the Verlet algorithm and predictor-corrector algorithms. They are quickly presented in the sections below. For more detailed informations on time integration algorithms, the reader is referred to refs. [3, 6] for a general survey, and to ref. [19] for a deeper analysis.

2.3.1 The Verlet algorithm

In molecular dynamics, the most commonly used time integration algorithm is probably the so-called Verlet algorithm [13]. The basic idea is to write two third-order Taylor expansions for the positions $\mathbf{r}(t)$, one forward and one backward in time. Calling $\mathbf{v}$ the velocities, $\mathbf{a}$ the accelerations, and $\mathbf{b}$ the third derivatives of $\mathbf{r}$ with respect to t, one has:

$$\mathbf{r}(t+\Delta t) = \mathbf{r}(t) + \mathbf{v}(t)\Delta t + (1/2)\mathbf{a}(t)\Delta t^2 + (1/6)\mathbf{b}(t)\Delta t^3 + O(\Delta t^4)$$

$$\mathbf{r}(t-\Delta t) = \mathbf{r}(t) - \mathbf{v}(t)\Delta t + (1/2)\mathbf{a}(t)\Delta t^2 - (1/6)\mathbf{b}(t)\Delta t^3 + O(\Delta t^4)$$

Adding the two expressions gives

$$\mathbf{r}(t+\Delta t) = 2\mathbf{r}(t) - \mathbf{r}(t-\Delta t) + \mathbf{a}(t)\Delta t^2 + O(\Delta t^4) \tag{2.5}$$

This is the basic form of the Verlet algorithm. Since we are integrating Newton's equations, $\mathbf{a}(t)$ is just the force divided by the mass, and the force is in turn a function of the positions $\mathbf{r}(t)$:

$$\mathbf{a}(t) = -(1/m)\nabla V\left(\mathbf{r}(t)\right) \tag{2.6}$$

As one can immediately see, the truncation error of the algorithm when evolving the system by Δt is of the order of Δt^4, even if third derivatives do not appear explicitly. This algorithm is at the same time simple to implement, accurate and stable, explaining its large popularity among molecular dynamics simulators.

A problem with this version of the Verlet algorithm is that velocities are not directly generated. While they are not needed for the time evolution, their knowledge is sometimes necessary. Moreover, they are required to compute the kinetic energy K, whose evaluation is necessary to test the conservation of the total energy $E = K + V$. This is one of the most important tests to verify that a MD simulation is proceeding correctly. One could compute the velocities from the positions by using

$$\mathbf{v}(t) = \frac{\mathbf{r}(t + \Delta t) - \mathbf{r}(t - \Delta t)}{2\Delta t}.$$

However, the error associated to this expression is of order Δt^2 rather than Δt^4.

To overcome this difficulty, some variants of the Verlet algorithm have been developed. They give rise to exactly the same trajectory, and differ in what variables are stored in memory and at what times. The *leap-frog* algorithm, not reported here, is one of such variants [20] where velocities are handled somewhat better.

An even better implementation of the same basic algorithm is the so-called *velocity Verlet* scheme, where positions, velocities and accelerations at time $t + \Delta t$ are obtained from the same quantities at time t in the following way:

$$\begin{aligned} \mathbf{r}(t + \Delta t) &= \mathbf{r}(t) + \mathbf{v}(t)\Delta t + (1/2)\mathbf{a}(t)\Delta t^2 \\ \mathbf{v}(t + \Delta t/2) &= \mathbf{v}(t) + (1/2)\mathbf{a}(t)\Delta t \\ \mathbf{a}(t + \Delta t) &= -(1/m)\nabla V\left(\mathbf{r}(t + \Delta t)\right) \\ \mathbf{v}(t + \Delta t) &= \mathbf{v}(t + \Delta t/2) + (1/2)\mathbf{a}(t + \Delta t)\Delta t \end{aligned}$$

Note how we need $9N$ memory locations to save the $3N$ positions, velocities and accelerations, but we never need to have simultaneously stored the values at two different times for any one of these quantities.

2.3.2 Predictor-corrector algorithm

Predictor-corrector algorithms constitute another commonly used class of methods to integrate the equations of motion. Those more often used in molecular dynamics (see e.g. [12]) are due to Gear, and consists of three steps:

1. *Predictor.* From the positions and their time derivatives up to a certain order q, all known at time t, one "predicts" the same quantities at time $t + \Delta t$ by means of a Taylor expansion. Among these quantities are, of course, accelerations $\mathbf{a}$.

2. *Force evaluation.* The force is computed taking the gradient of the potential at the predicted positions. The resulting acceleration will be in general different from the "predicted acceleration". The difference between the two constitutes an "error signal".

3. *Corrector.* This error signal is used to "correct" positions and their derivatives. All the corrections are proportional to the error signal, the coefficient of proportionality being a "magic number" determined to maximize the stability of the algorithm.

For more details the reader is referred to [3], §3.2, or [6], §4.4. A detailed comparison between the Verlet and the Gear scheme can be found in [19].

Chapter 3

Running, measuring, analyzing

In this chapter I discuss how we interact with a molecular dynamics program:

- how we start a simulation
- how we control a simulation, for instance to explore a phase diagram of a substance
- how we extract results from a simulation

plus a couple of extra topics.

3.1 Starting a simulation

To start a simulation we need to define the MD box and the set of positions and velocities to assign initially to the particles. There are two common ways of doing this:

3.1.1 Starting from scratch

If we are starting from scratch, we have to "create" a set of initial positions and velocities.

Positions are usually defined on a lattice, assuming a certain crystal structure. This structure is typically the most stable at T=0 with the given potential. Initial velocities may be taken to be zero, or from a Maxwellian distribution as described below.

Such initial state will not of course correspond to an equilibrium condition. However, once the run is started equilibrium is usually reached within a time of the order of 100 time steps.

Some randomization must be introduced in the starting sample. If we do not do so, then all atoms are symmetrically equivalent, and the equation of motions cannot do anything different that evolving them in exactly the same way. In the case of a perfect lattice, there would be no net force on the atoms for symmetry reasons, and therefore atoms would sit idle indefinitely.

Typical ways to introduce a random component are

1. small random displacements are added to the lattice position. The amplitude of these displacements must not be too large, in order to avoid overlap of atomic cores. A few percent of the lattice spacing is usually more than adequate.

2. the initial velocities are assigned taking them from a Maxwell distribution at a certain temperature T. When doing this, the system will have a small total linear momentum, corresponding to a translational motion of the whole system. Since this is somewhat inconvenient to have, it is common practice to subtract this component from the velocity of each particle in order to operate in a zero total momentum condition.

Initial randomization is usually the only place where *chance* enters a molecular dynamics simulation. The subsequent time evolution is completely deterministic.

3.1.2 Continuing a simulation

Another possibility is to take the initial positions and velocities to be the final positions and velocities of a previous MD run.

This is in fact the most commonly used method in actual production. For instance, if one has to take "measurements" at different temperatures, the standard procedure is to set up a *chain of runs*, where the starting point at each T is taken to be the final point at the preceding T (lower if we are heating, higher if we are cooling).

3.2 Controlling the system

In a molecular dynamics simulation a system could be in a state characterized by a certain density, temperature, pressure (the calculation of these quantities is described below): the phase diagram of the simulated material can be investigated. However, how can we bring the system in the desired region? In other words, how can we *control* the system?

In a standard calculation, the density is controlled by the choice of the box volume V. Programs usually have provisions for rescaling the volume upon request at the beginning of the calculation. Volume changes should be modest, of the order of at most a few percent. The pressure will be measured during the run, following the method described in §3.5.7. In other, more advanced simulation schemes, the user chooses the pressure, and the volume of the box is a variable to be measured (see §3.11).

Temperature changes are usually achieved by enabling a device in the code which brings the system to a desired temperature by rescaling the velocities. In the *velocity Verlet* algorithm discussed at the end of §2.3.1, this may be accomplished by replacing the equation

$$\mathbf{v}(t + \Delta t/2) = \mathbf{v}(t) + (1/2)\mathbf{a}(t)\Delta t$$

with

$$\mathbf{v}(t+\Delta t/2) = \sqrt{\frac{T_\circ}{T(t)}}\mathbf{v}(t) + (1/2)\mathbf{a}(t)\Delta t$$

where $T_\circ$ is the desired temperature, and $T(t)$ the "instantaneous temperature". Such a modification means that we are no longer following Newton's equations, and the total energy is no longer conserved. Important data should not be collected in this stage: these "controlled temperature" simulations should be used only to bring the system from one state to another. One should always wait for the system to *reach equilibrium* under clean constant energy conditions before collecting data.

3.3 Equilibration

Every time the state of the system changes, the system will be "out of equilibrium" for a while. We are referring here to thermodynamic equilibrium. By this, it is meant that the indicators of the system state—of which many are described below—are not stationary (that is, fluctuating around a fixed value), but relaxing towards a new value (that is, fluctuating around a value which is slowly drifting with time).

The state change may be induced by us or spontaneous. It is induced by us when we change a parameter of the simulation—such as the temperature, or the density—thereby perturbing the system, and then wait for a new equilibrium to be reached. It is spontaneous when, for instance, the system undergoes a phase transition, thereby moving from one equilibrium state to another.

In all cases, we usually want equilibrium to be reached before starting performing measurements on the system. A physical quantity A generally approaches its equilibrium value exponentially with time:

$$A(t) = A_\circ + C\exp(-t/\tau) \tag{3.1}$$

where $A(t)$ indicates here a physical quantities averaged over a short time to get rid of instantaneous fluctuations, but not of its long-term drift. The relevant variable is here the *relaxation time* τ. We may have cases where τ is of the order of hundreds of time steps, allowing us to see $A(t)$ converge to $A_\circ$ and make a direct measurement of the equilibrium value. In the opposite case, τ could be much larger than our overall simulation time scale; for instance, of the order of one second. In this case, we do not see any relaxation occurring during the run, and molecular dynamics hardly seems a valid technique to use in such a situation. In intermediate situations, we may see the drift but we cannot wait long enough to observe convergency of $A(t)$ to $A_\circ$. In these cases, one can often obtain an estimate for $A_\circ$ by applying (3.1) on the available data, even if the final point is still relatively far from it.

3.4 Looking at the atoms

The simplest—at least conceptually—kind of "probe" that we can apply to our system to understand what is going on is "looking at it". One can assign a radius to the atoms (for instance corresponding to half the distance at which the pair potential between two atoms becomes strongly repulsive), represent the atoms as balls having that radius, and have a plotting program construct a "photograph" of the system from a certain point of observation. The resulting image can be observed on a graphic screen or printed.

Programs capable of doing this are commonly available. One of them, called BallRoom[1], has been written by the author and its source code is freely available. Among other things, this program allows to map properties attached to individual atoms (for instance, diffusion, potential energy, coordination, etc) to colors, providing a vivid representation of the spatial distribution of these properties.

In some cases, such as in the analysis of complex structural arrangements of atoms, visual inspection is an invaluable tool. In other cases, however, it gives little information, and useful data must be extracted by processing atomic coordinates and/or velocities, obtaining for instance correlation functions, as discussed in the following.

3.5 Simple statistical quantities to measure

Measuring quantities in molecular dynamics usually means performing *time averages* of physical properties over the system trajectory. Physical properties are usually a function of the particle coordinates and velocities. So, for instance, one can define the instantaneous value of a generic physical property A at time t:

$$A(t) = f\left(\mathbf{r}_1(t), \ldots, \mathbf{r}_N(t), \mathbf{v}_1(t), \ldots, \mathbf{v}_N(t)\right)$$

and then obtain its average

$$\langle A \rangle = \frac{1}{N_T} \sum_{t=1}^{N_T} A(t)$$

where t is an index which runs over the time steps from 1 to the total number of steps N_T.

There are two equivalent ways to do this in practice:

1. $A(t)$ is calculated at each time step by the MD program while running. The sum $\sum_t A(t)$ is also updated at each step. At the end of the run the average is immediately obtained by dividing by the number of steps. This is the preferred method when the quantity is simple to compute and/or particularly important. An example is the system temperature.

2. Positions (and possibly velocities) are periodically dumped in a "trajectory file" while the program is running. A separate program, running after the simulation program, processes the trajectory and computes the desired quantities. This approach can be very demanding in terms of disk space: dumping positions and velocities using 64-bit precision takes 48

[1] http://www.sissa.it/furio/ballroom.html

bytes per step and per particle. However, it is often used when the quantities to compute are complex, or combine different times as in dynamical correlations, or when they are dependent on other additional parameters that may be unknown when the MD program is run. In these cases, the simulation is run once, but the resulting trajectory can be processed over and over.

The most commonly measured physical properties are discussed in the following.

3.5.1 Potential energy

The average potential energy V is obtained by averaging its instantaneous value, which is usually obtained straightforwardly at the same time as the force computation is made. For instance, in the case of two-body interactions

$$V(t) = \sum_i \sum_{j>i} \phi\left(|\mathbf{r}_i(t) - \mathbf{r}_j(t)|\right)$$

Even if not strictly necessary to perform the time integration (forces are all is needed), knowledge of V is required to verify energy conservation. This is an important check to do in any MD simulation.

3.5.2 Kinetic energy

The instantaneous kinetic energy is of course given by

$$K(t) = \frac{1}{2} \sum_i m_i [v_i(t)]^2$$

and is therefore extremely easy to compute. We will call K its average on the run.

3.5.3 Total energy

The total energy $E = K + V$ is a conserved quantity in Newtonian dynamics. However, it is common practice to compute it at each time step in order to check that it is indeed constant with time. In other words, during the run energy flows back and forth between kinetic and potential, causing *K(t)* and *V(t)* to fluctuate while their sum remains fixed.

In practice there could be small fluctuations in the total energy, in a typical amount of, say, one part in 10^4 or less. These fluctuations are usually caused by errors in the time integration (see 2.3), and can be reduced in magnitude by reducing the time step if considered excessive. Ref. [19] contains an in-depth analysis of total energy fluctuations using various time integration algorithms.

Slow *drifts* of the total energy are also sometimes observed in very long runs. Such drifts could also be originated by an excessive Δt. Drifts are more disturbing than fluctuations because the thermodynamic state of the system is also changing together with the energy, and therefore time averages over the run do not refer to a single state. If drifts over long runs tend to occur, they can be prevented, for instance by breaking the long run into smaller pieces and restoring the energy to the nominal value between one piece and the next. A common mechanism to adjust the energy is to modify the kinetic energy via rescaling of velocities.

A final word of caution: while one may be tempted to achieve "perfect" energy conservation by reducing the time step as much as desired, working with an excessively small time step may result in waste of computer time. A practical compromise would probably allow for small energy fluctuations and perhaps slow energy drifts, as a price to pay to work with a reasonably large Δt. See also refs. [3], §3.5, and [6], §4.4.4.

3.5.4 Temperature

The temperature T is directly related to the kinetic energy by the well-known equipartition formula, assigning an average kinetic energy $k_B T/2$ per degree of freedom:

$$K = \frac{3}{2} N k_B T \tag{3.2}$$

An estimate of the temperature is therefore directly obtained from the average kinetic energy K (see §3.5.2). For practical purposes, it is also common practice to define an "instantaneous temperature" *T(t)*, proportional to the instantaneous kinetic energy *K(t)* by a relation analogous to the one above.

3.5.5 The caloric curve

Once the total energy E (also called internal energy) and the temperature T are measured in different runs corresponding to different thermodynamic states, one can construct the *caloric curve E(T)*. This is a useful tool to monitor the occurrence of phase transitions, which imply a jump of $E(T)$ if first-order, or in a derivative of $E(T)$ if second- or higher order. This strategy, however, is not suitable for a reliable estimate of the melting temperature T_m, as discussed later in §3.6.

The most common first-order transition is melting, easily observable by simulation. When the system abandons the crystalline structure and becomes a disordered liquid, we observe a jump of $E(T)$, corresponding to the latent heat of fusion. This usually happens at a temperature somewhat higher than the true melting temperature T_m of the model, due to hysteresis effects associated with the necessity to wait for a seed of the liquid phase to appear by a spontaneous fluctuation. Only when a liquid seed containing a few atoms is formed, the liquid phase can start to grow at the expense of the solid one. On the typical molecular dynamics time scale, one has to "overshoot" and set the temperature 20-30% above T_m to see melting happening.

3.5.6 Mean square displacement

The mean square displacement of atoms in a simulation can be easily computed by its definition

$$\mathrm{MSD} = \langle |\mathbf{r}(t) - \mathbf{r}(0)|^2 \rangle \tag{3.3}$$

where $\langle \ldots \rangle$ denotes here averaging over all the atoms (or all the atoms in a given subclass). Care must be taken to *avoid* considering the "jumps" of particles to refold them into the box when using periodic boundary conditions as contributing to diffusion.

The MSD contains information on the atomic diffusivity. If the system is solid, MSD saturates to a finite value, while if the system is liquid, MSD grows linearly with time. In this case it is useful to characterize the system behavior in terms of the slope, which is the *diffusion coefficient* D:

$$D = \lim_{t\to\infty} \frac{1}{6t} \langle |\mathbf{r}(t) - \mathbf{r}(0)|^2 \rangle \tag{3.4}$$

The 6 in the above formula must be replaced with 4 in two-dimensional systems.

3.5.7 Pressure

The measurement of the pressure in a molecular dynamics simulation is based on the Clausius virial function

$$W(\mathbf{r}_1, \ldots, \mathbf{r}_N) = \sum_{i=1}^{N} \mathbf{r}_i \cdot \mathbf{F}_i^{\mathrm{TOT}} \tag{3.5}$$

where $\mathbf{F}_i^{\mathrm{TOT}}$ is the total force acting on atom i. Its statistical average $\langle W \rangle$ will be obtained, as usual, as an average over the molecular dynamics trajectory:

$$\langle W \rangle = \lim_{t\to\infty} \frac{1}{t} \int_0^t d\tau \sum_{i=1}^{N} \mathbf{r}_i(\tau) \cdot m_i \ddot{\mathbf{r}}_i(\tau).$$

where use has been made of Newton's law. By integrating by parts,

$$\langle W \rangle = -\lim_{t\to\infty} \frac{1}{t} \int_0^t d\tau \sum_{i=1}^{N} m_i |\dot{\mathbf{r}}_i(\tau)|^2.$$

This is twice the average kinetic energy, therefore by the equipartition law of statistical mechanics

$$\langle W \rangle = -DNk_BT \tag{3.6}$$

where D is the dimensionality of the system (2 or 3), N the number of particles, k_B the Boltzmann constant.

Now, one may think of the total force acting on a particle as composed of two contributions:

$$\mathbf{F}_i^{\mathrm{TOT}} = \mathbf{F}_i + \mathbf{F}_i^{\mathrm{EXT}} \tag{3.7}$$

where $\mathbf{F}_i$ is the internal force (arising from the interatomic interactions), and $\mathbf{F}_i^{\mathrm{EXT}}$ is the external force exerted by the container's walls. If the particles are enclosed in a parallelepipedic container of sides L_x, L_y, L_z, volume $V = L_xL_yL_z$, and with the coordinates origin on one of its corners, the part $\langle W^{\mathrm{EXT}}\rangle$ due to the container can be evaluated using the definition (3.5):

$$\langle W^{\mathrm{EXT}}\rangle = L_x(-PL_yL_z) + L_y(-PL_xL_z) + L_z(-PL_xL_y) = -DPV$$

where $-PL_yL_z$ is, for instance, the external force F_x^{EXT} applied by the yz wall along the x directions to particles located at $x = L_x$, etc. Eq. (3.6) can then be written

$$\left\langle \sum_{i=1}^{N} \mathbf{r}_i \cdot \mathbf{F}_i \right\rangle - DPV = -DNk_BT$$

or

$$PV = Nk_BT + \frac{1}{D}\left\langle \sum_{i=1}^{N} \mathbf{r}_i \cdot \mathbf{F}_i \right\rangle \tag{3.8}$$

This important result is known as the *virial equation.* All the quantities except the pressure P are easily accessible in a simulation, and therefore (3.8) constitutes a way to measure P. Note how (3.8) reduces to the well-known equation of state of the perfect gas if the particles are non-interacting.

In the case of pairwise interactions via a potential $\phi(r)$, it is left as an exercise to the reader to verify that (3.8) becomes

$$PV = Nk_BT - \frac{1}{D}\left\langle \sum_{i}\sum_{j>i} r_{ij} \left.\frac{d\phi}{dr}\right|_{r_{ij}} \right\rangle . \tag{3.9}$$

This expression has the additional advantage over (3.8) to be naturally suited to be used when periodic boundary conditions are present: it is sufficient to take them into account in the definition of r_{ij}.

3.6 Measuring the melting temperature

The behavior of the mean square displacement as a function of time easily allows to discriminate between solid and liquid. One might be then tempted to locate the melting temperature T_m of the simulated substance by increasing the temperature of a crystalline system until diffusion appears, and the caloric curve exhibits a jump indicating absorption of latent heat.

While these are indeed indicators of a transition from solid to liquid, the temperature at which this happen in a MD simulation is invariably higher than the melting temperature. In fact, the melting point is by definition the temperature at which the solid and the liquid phase coexist (they have the same free energy). However, lacking a liquid seed from where the liquid could nucleate and grow, overheating above melting commonly occurs. In this region the system is in a thermodynamically metastable state, nevertheless it appears stable within the simulation time. An overheated bulk crystal breaks down when its *mechanical instability point* is reached. This point may correspond to the vanishing of one of the shear moduli of the material or to similar instabilities, and is typically larger than T_m by an amount of the order of 20-30%.

While mechanical instability is certainly useful to estimate roughly the location of T_m, more precise methods exist. One of them consists of setting up a large sample consisting approximately of 50% solid and 50% liquid. Such a sample could be initially constructed artificially[2], and it will contain interface regions between solid and liquid. One then tries to establish an equilibrium state where solid and liquid can coexist. When this goal is achieved, the temperature of the state has to be T_m by definition.

To this end, the system is evolved microcanonically at an energy E believed to be in proximity of the melting jump in the caloric curve. Let us call $T_\circ$ the initial temperature assumed by the system. This temperature is an increasing but unknown function of the energy E, and it will in general differ from T_m. Suppose that $T_\circ > T_m$. Then, the liquid is favored over the solid, and the solid-liquid interfaces present in the system will start to move in order to increase the fraction of liquid at the expense of the solid. In other words, some material melts. As this happens, the corresponding latent heat of melting is absorbed by the system. Since the total energy is conserved, absorption of latent heat automatically implies a decrease of the kinetic energy. Therefore, the temperature goes down. The force driving the motion of the solid-liquid interface will also decrease, and as time goes on, the position of the interface and the temperature will exponentially reach an equilibrium state. The temperature will then be T_m.

If $T_\circ < T_m$, the inverse reasoning applies: latent heat is released and converted into kinetic energy, and temperature again converges exponentially to T_m, this time from below.

The melting temperature depends on the pressure of the system. A pressure measurement on the final state is also required to characterize it thermodynamically. More often, the above procedure is carried out using constant pressure techniques (see §3.11). In this case, the volume of the box automatically changes as material melts or crystallizes, to accommodate the difference in density between the two phases at constant pressure.

3.7 Real space correlations

Real space correlation functions are typically of the form

$$\langle A(\mathbf{r})A(\mathbf{0})\rangle$$

and are straightforward to obtain by molecular dynamics: one has to compute the quantity of interest $A(\mathbf{r})$ starting from the atomic positions and velocities for several configurations, construct the correlation function for each configuration, and average over the available configurations.

The simplest example is the pair correlation function $g(r)$, which is essentially a density-density correlation. $g(r)$ is the probability to find a pair a distance r apart, relative to what expected for a uniform random distribution

[2]A possible way to do this is to "freeze" half of the particles (no motion allowed), raise the temperature in order to melt the other half, and then release the frozen particles.

of particles at the same density:

$$\rho g(r) = \frac{1}{N} \left\langle \sum_{i=1}^{N} \sum_{\substack{j=1 \\ (j \neq i)}}^{N} \delta(r - r_{ij}) \right\rangle \tag{3.10}$$

This function carries information on the structure of the system. For a crystal, it exhibits a sequence of peaks at positions corresponding to shells around a given atom. The positions and magnitude of the peaks are a "signature" of the crystal structure (fcc, hcp, bcc, ...) of the system. For a liquid, $g(r)$ exhibits its major peak close to the average atomic separation of neighboring atoms, and oscillates with less pronounced peaks at larger distances. The magnitude of the peaks decays exponentially with distance as $g(r)$ approaches 1. In all cases, $g(r)$ vanishes below a certain distance, where atomic repulsion is strong enough to prevent pairs of atoms from getting so close.

One quantity often computed from $g(r)$ is the average number of atoms located between r_1 and r_2 from a given atom,

$$\rho \int_{r_1}^{r_2} g(r) 4\pi r^2 \, dr.$$

This allows to define coordination numbers also in situations where disorder is present.

The calculation of $g(r)$ is intrinsically a $O(N^2)$ operation, and therefore it can slow down considerably an optimized molecular dynamics program. If the behavior at large r is not important, it might be convenient to define a cutoff distance, and use techniques borrowed from fast force calculations using short-ranged potentials to decrease the computational burden. It should also be noted that periodic boundary conditions impose a natural cutoff at $L/2$, where L is the minimum between the box sizes L_x, L_y, L_z in the three directions. For larger distance the results are spoiled by size effects.

3.8 Reciprocal space correlations

Structural informations can also be collected by working in reciprocal space. The most basic quantity is the space Fourier transform of the density $\rho(\mathbf{r}) = \sum_i \delta(\mathbf{r} - \mathbf{r}_i)$:

$$\rho(\mathbf{k}) = \sum_{i=1}^{N} \mathrm{e}^{i\mathbf{k}\cdot\mathbf{r}_i} \tag{3.11}$$

This quantity, easily *and quickly* obtained for a given configuration, is the building block to construct the *static structure factor*

$$\begin{aligned} S(\mathbf{k}) &= \tfrac{1}{N} \langle \rho(\mathbf{k}) \rho(-\mathbf{k}) \rangle \\ &= \tfrac{1}{N} \left\langle \textstyle\sum_{ij} \mathrm{e}^{i\mathbf{k}\cdot(\mathbf{r}_i - \mathbf{r}_j)} \right\rangle \\ &= \tfrac{1}{N} \left\langle \textstyle\sum_{ij} \cos(\mathbf{k} \cdot \mathbf{r}_{ij}) \right\rangle \end{aligned} \tag{3.12}$$

This quantity is extremely useful for comparisons with results from scattering experiments. In liquids, which are isotropic, $S(k)$ depends only on the modulus of the wavevector, and is essentially a Fourier transform of the pair correlation function:

$$S(k) = 1 + \rho g(k).$$

The computation of S(k) via (3.12) is however much faster than that of $g(r)$.

By replacing $\rho(\mathbf{k})$ with some other generic observable $A(\mathbf{k})$, any other correlation function in reciprocal space can be computed.

It should be noted that, working with periodic boundary conditions, results could not depend on the choice of a particular particle image. That is, one must impose that (choosing for instance direction x) $\mathrm{e}^{ik_x(x_i+L_x)} = \mathrm{e}^{ik_x x_i}$ where L_x is the length of the repeated box. This implies a restriction in the choice of the allowable wavevectors:

$$k_x = \frac{2\pi}{L_x} n_x$$

where n_x is an integer, and similarly along y and z. Therefore, the allowed wavevectors are quantized as a consequence of adopting periodic boundary conditions.

3.9 Dynamical analysis

One strength of molecular dynamics with respect to other methods such as Monte Carlo is the fact that the time evolution of the system is followed and—therefore—information on the dynamics of the system is fully present. Experimental data (for instance, phonon dispersion curves in the case of crystals) are usually available in the wavevector-frequency space. Information obtained by MD in real space and real time will therefore be Fourier-transformed with respect to both space and time for useful comparisons with experimental results. To perform dynamical analysis, one typically instructs the simulation program to dump periodically on a mass storage medium the positions and possibly the velocities of all the particles. These can easily produce very large files: the calculation of dynamical quantities requires plenty of room for data. These data are then analyzed later by other programs to extract the quantities of interest. In some cases, dynamical analysis are directly performed by the MD program while it is running, thus saving disk space.

One typical quantity is the Van Hove correlation function $G(\mathbf{r}, t)$, representing the probability to find an atom at position $\mathbf{r}$ at time t if there was an atom at position $\mathbf{r} = 0$ at time $t = 0$. This quantity is connected by a double Fourier transform to the *dynamic structure factor* $S(\mathbf{k}, \omega)$ reported by experiments:

$$S(\mathbf{k}, \omega) = \int\int d\mathbf{r}\, dt\, e^{i(\mathbf{k}\cdot\mathbf{r} - \omega t)} G(\mathbf{r}, t). \qquad (3.13)$$

As for static quantities, the most convenient way to proceed in terms of computational effort is that of using the space Fourier transform of the density as a building block:

$$\rho(\mathbf{k}, t) = \sum_i \mathrm{e}^{i\mathbf{k}\cdot\mathbf{r}_i(t)} \qquad (3.14)$$

This is a single-particle quantity which is very fast to compute. From here, one constructs a time-dependent density-density correlation function (also called *intermediate scattering function*) by performing an average over a molecular dynamics trajectory:

$$F(\mathbf{k},t) = \frac{1}{N} \langle \rho(\mathbf{k}, t+t_\circ)\rho(-\mathbf{k}, t_\circ) \rangle . \tag{3.15}$$

The average is made for all the available choices for $t_\circ$. The maximum t (connected with the frequency resolution) is clearly limited by the length of the run, while the periodicity with which data are collected—typically a few time steps—controls the maximum frequency in the final spectra. The dynamic structure factor is finally obtained by a time Fourier transform:

$$S(\mathbf{k},\omega) = \int dt\, \mathrm{e}^{-i\omega t} F(\mathbf{k},t) \tag{3.16}$$

Again, the allowed $\mathbf{k}$ are quantized as a consequence of periodic boundary conditions. $S(\mathbf{k},\omega)$ will exhibit peaks in the $(\mathbf{k},\omega)$ plane, corresponding to the dispersion of propagating modes. The broadening of these peaks is related with anharmonicity. By performing calculations at different temperatures, one can for instance study the effect of temperature on the phonon spectra, fully including anharmonic effects: in this sense, the approach is clearly superior to standard lattice dynamics techniques. The main inconvenience of MD is that it is not so straightforward to extract eigenvectors.

When dealing with vibrational modes and different polarizations, it is often convenient to follow the approach outlined above, but using currents

$$\mathbf{j}(\mathbf{k},t) = \sum_i \mathbf{v}_i(t) \mathrm{e}^{i\mathbf{k}\cdot\mathbf{r}_i(t)} \tag{3.17}$$

in place of the density $\rho(\mathbf{k},t)$.

3.10 Annealing and quenching: MD as an optimization tool

Molecular dynamics may also have a role as on optimization tool. Let us suppose that a set of N particles has many possible equilibrium configurations. The energy of these configurations is in general different, and one of them will be the optimal one; all of them, however, correspond to *local minima* in the energy, and are each other separated by energy barriers. Such a situation occurs very commonly, for instance, in cluster physics.

Finding the optimal structure within an approach based on traditional minimization techniques (steepest descent method, conjugate gradient method, etc.) is tricky, as these methods do not normally overcome energy barriers and tend to fall into the nearest local minimum. One therefore would have to try out several different starting points, corresponding to different "attraction basins" in the energy landscape, and relax each of them to the bottom of the basin. The optimal structure would then be the one with the lowest energy, provided that we were sage enough to select it in the list of candidates to try.

Temperature in a molecular dynamics (or Monte Carlo) calculation provides a way to "fly over the barriers": states with energy E are visited with a probability $\exp(-E/k_BT)$. If T is sufficiently large, the system can "see" the simultaneous existence of many different minima, spending more time in the deeper ones. By decreasing slowly T to 0, there is a good chance that the system will be able to pick up the best minimum and land into it. This consideration is the base of *simulated annealing* methods, where the system is equilibrated at a certain temperature and then slowly cooled down to $T = 0$. While this procedure does not guarantee that the true minimum will be reached, it often brings the system into a good minimum, and as no a priori assumptions are made about the optimal structure, it often brings about structures which were difficult to foresee by intuition alone.

This method is often used to optimize atomic structures, but its validity is more general: given an objective function $Z(\alpha_1, \ldots, \alpha_N)$ depending on N parameters, one can regard each of these parameters as a degree of freedom, assign a "mass' to it, and move the system with a molecular dynamics or Monte Carlo algorithm to perform simulated annealing. One of the early application of this method can be found in a famous paper discussing an application to the "traveling salesman" problem [21].

3.11 Other statistical ensembles

We have discussed so far the standard molecular dynamics scheme, based on the time integration of Newton's equations and leading to the conservation of the total energy. In the statistical mechanics parlance, these simulations are performed in the *microcanonical ensemble*, or NVE ensemble: the number of particles, the volume and the energy are constant quantities.

The microcanonical average of a physical quantity A is obtained as a time average on the trajectory:

$$\langle A \rangle_{\mathrm{NVE}} = \frac{1}{N_T} \sum_{t=1}^{N_T} A(\Gamma(t)) \tag{3.18}$$

where I denote with $\Gamma(t)$ the phase space coordinate of the system ($3N$ positions and $3N$ velocities).

There are other important alternatives to the NVE ensemble, that we will mention here only briefly. The basic idea is that of integrating other equations in place of Newton's equations, in such a way that sampling is performed in another statistical ensemble. Averages of physical quantities in the new ensemble will be again obtained as time averages, similarly to eq. (3.18).

A scheme for simulations in the isoenthalpic-isobaric ensemble (NPH) has been developed by Andersen [22]. Here, an additional degree of freedom representing the volume of the box has been introduced, and all the particle coordinates are given in units relative to the box. The volume V of the box becomes a dynamical variable, with a kinetic energy and a potential energy which just PV, where P is the external pressure. The enthalpy $H = E + PV$ is a conserved quantity.

Parrinello and Rahman [23] developed a variant where the shape of the box can vary as well as the volume. This is achieved by introducing 9 new degrees of freedom instead of 1: the components of the three vectors spanning the MD box. Each of them is a new dynamical variable, evolving accordingly to equation of motion derived from an appropriate Lagrangian. This scheme allows to study structural phase transitions as a function of pressure, where for example the system abandons a certain crystal structure in favor of a more compact one.

Another very important ensemble is the canonical ensemble (NVT). In a method developed by Nosè and Hoover [24], this is achieved by introducing a time-dependent frictional term, whose time evolution is driven by the imbalance between the instantaneous kinetic energy and the average kinetic energy $(3N/2)k_BT$.

Chapter 4

Interatomic potentials

In molecular dynamics forces are derived from a potential energy function V, which depend on the particle coordinates:

$$\mathbf{F}_i = -\nabla V(\mathbf{r}_1, \ldots, \mathbf{r}_N) \tag{4.1}$$

The problem of modelling a material can therefore be restated as that of finding a potential function $V(\mathbf{r}_1, \ldots, \mathbf{r}_N)$ for that material.

Can such a function exist? And if yes, how is it related with the behavior of the real material, which we know to be controlled by the law of quantum mechanics rather than classical mechanics, and where the electrons certainly play the major role in determining the bonding properties of the system? And perhaps the most important question: how to *find* a potential in practice?

4.1 The Born-Oppenheimer approximation

What do we mean *exactly* when we talk about "interatomic potentials"? A system of interacting atoms is really made up of nuclei and electrons which interact with each other. The true Hamiltonian for this system may be written as

$$H = \sum_i \frac{P_i^2}{2M_i} + \sum_n \frac{p_n^2}{2m} + \frac{1}{2}\sum_{ij} \frac{Z_i Z_j e^2}{|\mathbf{R}_i - \mathbf{R}_j|} + \frac{1}{2}\sum_{nn'} \frac{e^2}{|\mathbf{r}_n - \mathbf{r}_{n'}|} - \sum_{in} \frac{Z_i e^2}{|\mathbf{R}_i - \mathbf{r}_n|} \tag{4.2}$$

where indexes i, j run on nuclei, n and n' on electrons, $\mathbf{R}_i$ and $\mathbf{P}_i$ are positions and momenta of the nuclei, $\mathbf{r}_n$ and $\mathbf{p}_n$ of the electrons, Z_i the atomic number of nucleus i, M_i its mass and m the electron mass. One clearly recognizes the kinetic energy terms and the various coulombic interactions. In principle, one should solve a Schrödinger equation for the total wavefunction $\Psi(\mathbf{R}_i, \mathbf{r}_n)$ and then everything about the system is known.

Of course, this is impossible to carry out in practice, and approximation schemes have to be employed. In 1923, Born and Oppenheimer noted that nuclei are much heavier than electrons, and move on a time scale which is about two order of magnitude longer than that of the electrons:

$$\frac{\omega_{\mathrm{el}}}{\omega_{\mathrm{nuc}}} \sim \sqrt{\frac{M}{m}} \sim 100 \tag{4.3}$$

It is therefore sensible to regard the nuclei as fixed as far as the electronic part of the problem is concerned, and factorize the total wavefunction as

$$\Psi(\mathbf{R}_i, \mathbf{r}_n) = \Xi(\mathbf{R}_i)\Phi(\mathbf{r}_n; \mathbf{R}_i) \tag{4.4}$$

where $\Xi(\mathbf{R}_i)$ describes the nuclei, and $\Phi(\mathbf{r}_n; \mathbf{R}_i)$ the electrons (depending parametrically on the positions of the nuclei). With this assumptions, the problem is reformulated in terms of two separate Schrödinger equations:

$$H_{\mathrm{el}}\Phi(\mathbf{r}_n; \mathbf{R}_i) = V(\mathbf{R}_i)\Phi(\mathbf{r}_n; \mathbf{R}_i) \tag{4.5}$$

where

$$H_{\mathrm{el}} = \sum_n \frac{p_n^2}{2m} + \frac{1}{2}\sum_{ij}\frac{Z_i Z_j e^2}{|\mathbf{R}_i - \mathbf{R}_j|} + \frac{1}{2}\sum_{nn'}\frac{e^2}{|\mathbf{r}_n - \mathbf{r}_{n'}|} - \sum_{in}\frac{Z_i e^2}{|\mathbf{R}_i - \mathbf{r}_n|}, \tag{4.6}$$

and

$$\left[\sum_i \frac{P_i^2}{2M_i} + V(\mathbf{R}_i)\right]\Xi(\mathbf{R}_i) = E\Xi(\mathbf{R}_i). \tag{4.7}$$

Eq. (4.5) is the equation for the electronic problem, considering the nuclei as fixed. The eigenvalue of the energy $V(\mathbf{R}_i)$ will depend parametrically on the coordinates of the nuclei; we call this quantity the *interatomic potential.* Once found, this quantity enters (4.7) which will give the motion of the nuclei. Note how in this equation there are no electronic degrees of freedom: all the electronic effects are incorporated in $V(\mathbf{R}_i)$. It is customary to replace this Schrödinger equation with a Newton equation, that is, to move the nuclei classically. We have already discussed in §1.5.1 the conditions under which this can be done safely.

4.2 The design of potentials

The above section defines rigorously a potential, but it is not of much practical help: solving (4.5) is still a formidable task and constitutes in fact a large fraction of the activity done in condensed matter physics.

One possibility is to adopt some approximation, but *really* solve this equation: we then have *first-principles molecular dynamics*, briefly mentioned later in §4.7. While this is feasible, it requires massive computer resources, and poses severe limits on the maximum size of the system and on the simulation time. In these notes I will rather focus on the traditional approach where we get rid completely of electronic degrees of freedom and move our nuclei (atoms) according to some potential function $V(\mathbf{R}_i)$ whose analytical form we specify in advance. Our goal is to select functional forms which mimic the behavior of the "true" potential in realistic ways for specific materials.

Constructing a potential involves two steps:

1. Selecting an analytical form for the potential. In the past this was almost invariably a sum of pairwise terms, with the energy of a pair depending on their relative distance. Today novel many-body forms are tried in

the attempt to capture as much as possible the physics and chemistry of the bonding. A typical analytical form is constituted by a number of functions, depending on geometrical quantities such as distances or angles, or on intermediate variables such as atom coordinations.

2. Finding an actual parametrization for the functions that constitute the analytical form we have chosen. This step is very important and can be technically elaborate.

A variety of techniques have been utilized over the years to this end. At one extreme, some groups tried to start from a *first principle* description (that is, where the electronic structure is kept into account), and obtain an expression for the energy as a function of the nuclei position by means of successive approximations. At the other extreme, other groups chose to fit the potential (that is, to parametrize the functions which appear in its analytical form) on experimental data, giving maximum priority to realism rather than to connections with first-principles.

In all cases, potentials are designed with a "range of applicability" in mind. Due to the vast differences in the electronic structure, it would probably be too ambitious to try modeling a bulk metal and a diatomic molecule of the same element with the same potential: the environment is dramatically different. However, it could be feasible to model simultaneously a bulk and a surface environment, where the environment differs (due to the reduced coordination of the atoms at the surface) but not as dramatically as in the previous example. The ability of a potential to work properly in different environments is called *transferability*. When using a potential, the simulator should always be familiar with its transferability properties, and look critically at results obtained in unusual conditions—for example, very low coordinations, or very high temperature, or very high pressure.

4.3 The problems with two-body potentials

In §2.1.1 I presented the Lennard-Jones potential, which is a typical pairwise potential and probably the most commonly used one. In spite of that, the class of materials which can be realistically modeled using this approach is in practice limited to rare gases, where no electrons are available for bonding and atoms are attracted with each other only through the weak van der Waals forces.

Systems of more practical interest such as metals and semiconductors cannot be modeled with pairwise forces. Considering for instance noble metals, one can easily identify a few indicators of many-atom effects, summarized in the following table, where experimental data for a few metals are compared with Lennard-Jones data (but other pair potentials yield similar results):

Property	Cu	Ag	Pt	Au	LJ
E_c/k_BT_m	30	28	33	33	13
E_v/E_c	0.33	0.36	0.26	0.25	~ 1
C_{12}/C_{44}	1.5	1.9	3.3	3.7	1

E_c/k_BT_m is the ratio between the cohesive energy and the melting temperature. This ratio is about 30 in metals, and about 10 in two-body systems. This result indicates that metals exhibit some "extra cohesion" with respect to pairwise systems, which is less effective than two-body forces in keeping the system in the crystalline state.

E_v/E_c is the ratio between the vacancy formation energy and the cohesive energy. This number is between 1/4 and 1/3 in metals, but about 1 in two-body systems (exactly 1 if relaxations are neglected, actually). When forming a vacancy in a crystal structure with coordination number Z, one has to pay the energy E_v to decrease coordination from Z to $Z-1$ for Z atoms. In contrast, E_c is the energy to pay to decrease the coordination of a single atom from Z to 0. In a two-body model, where a fixed energy contribution is attached to bonds between pairs of atoms, these two energies are the same as they are both associated to the breaking of Z bonds. But this is not what happens in metals!

C_{12}/C_{44} is the ratio between two elastic constants of a cubic crystal (all the systems in the table are fcc). This ratio is *exactly 1* in two-body systems: this is a so-called *Cauchy relation* that can be demonstrated analytically. But deviations in metals are very common. The high value in Au is to be related with the well-known high ductility and malleability of gold.

If one consider semiconductors, deviations from a two-body behavior are even worse. For instance, silicon undergoes a series of structural phase transitions (from tetrahedral to β-tin to simple cubic to fcc) under pressure: this indicates that the energy difference between this structures is not too large. In other words, the cohesive energy is nearly independent upon coordination, while a two-body model should favor the more packed structures, which have more bonds.

Due to these and other [25] shortcomings, since the mid '80s researchers started to figure out how to improve realism by incorporating many-atom effects in potentials.

4.4 Many-body potentials for metals

A significant progress was made during the '80s by the development of many-atom potentials for metals based on the concept of *density*, or *coordination*, as the key variable to consider. The main physical point to model is that bonds become weaker when the local environment becomes more crowded: another consequence of the Pauli principle. So, a plot of the cohesive energy as a function of coordination should not be decreasing linearly as in two-body systems, but should exhibit a positive curvature: decreasing faster when coordination is low, and more slowly as coordination increases.

A possible form for the attractive part of the potential (the repulsive one being still conveniently treated by a pairwise law) can be qualitatively obtained by working out a connection with the tight-binding formalism [26]. The result of this (non-rigorous) argument suggests a form for the energy of an atom i

$$E_i \propto \sqrt{\sum_j h_{ij}^2} \propto \sqrt{Z_i} \tag{4.8}$$

where $h_{ij} = h(r_{ij}) = \langle i|H|j\rangle$ are the overlap integrals between i and its neighbors. In the tight-binding formalism the orbitals are localized and these functions vanish beyond a certain cutoff distance.

The key result contained in (4.8) is that the energy is proportional to *the square root* of the coordination, rather than to the coordination itself as in two-body models. One can easily verify that such a scheme goes in the right direction in solving the problems associated with pairwise potentials in metals. For instance,

$$\frac{E_v}{E_c} \sim \frac{Z[\sqrt{Z} - \sqrt{Z-1}]}{\sqrt{Z}} \sim \frac{1}{2}, \tag{4.9}$$

and one can also verify that the Cauchy relation $C_{12} = C_{44}$ no longer holds.

Following this and other considerations, several schemes to construct many-atom potentials for metals were developed, all essentially based on an analytical form

$$V = \frac{1}{2} \sum_{\substack{i,j=1 \\ (j\neq i)}}^{N} \phi(r_{ij}) + \sum_{i=1}^{N} U(n_i) \tag{4.10}$$

where $\phi(r)$ is a two-body part, and $U(n)$ is a function giving the energy of an atom as a function of a "generalized coordination" n. n for a given atom is in turn constructed as a superpositions of contributions from neighboring atoms:

$$n_i = \sum_{\substack{j=1 \\ (j\neq i)}}^{N} \rho(r_{ij}) \tag{4.11}$$

where $\rho(r)$ is a short-ranged, decreasing function of distance. Belonging to this scheme are the *glue model* [25], the *embedded atom method* [27] and the *Finnis-Sinclair potentials* [28]. Also similar is the *effective medium theory* [29]. Even if sharing the analytical form, these scheme differ vastly in the procedure used to build the three functions $\phi(r)$, $\rho(r)$, $U(n)$ constituting the model, often resulting in rather different parametrizations for the same material. Fitting is in fact the true crux of the matter of building potentials, and some groups use molecular dynamics simulation to test trial potentials (for example, to evaluate thermal expansion and melting point) during the construction stage.

It is interesting to note that eq. (4.10) is invariant with respect to the transformation

$$\begin{aligned} \hat{\phi}(r) &= \phi(r) + 2\lambda\rho(r) \\ \hat{U}(n) &= U(n) - \lambda n \end{aligned} \tag{4.12}$$

for any value of λ. This implies that there is no unique choice for $\phi(r)$ and $U(n)$, and one can arbitrarily impose a condition such as $U'(n_\circ) = 0$ (where $n_\circ$ is, for instance, the coordination of a bulk atom at equilibrium at $T = 0$). It also shows that if $U(n)$ is a linear function of n, then the whole scheme reduces to a two-body potential: many-body effects are related to *the curvature* of $U(n)$. In fact, $U''(n)$ is invariant with respect to (4.12).

To implement these potentials in a molecular dynamics program, one has to evaluate the forces

$$\mathbf{F}_i = -\sum_{j\neq i}\Big(\phi'(r_{ij}) + [U'(n_i) + U'(n_j)]\rho'(r_{ij})\Big)\frac{\mathbf{r}_{ij}}{r_{ij}} \tag{4.13}$$

This calculation is only slightly more complex than that required for a two-body system. Energy and forces can still be obtained by using pair distances as the only input: no angular term or other 3-body or 4-body terms are present. This allows to write very fast MD programs. On the other hand, the lack of true angular forces makes it difficult to use these schemes to model metals where covalent effects in the bonding are important, for instance transition metals. New, more powerful schemes are being developed to handle these materials.

The interested reader is invited to check [26] for further details.

4.5 Many-body potentials for semiconductors

Semiconductors are even more challenging than metals. Take for instance silicon, and consider that

- the most stable phase in the absence of pressure has the diamond structure: very open, with a coordination number of only 4;
- when a pressure is applied, new structures with increasing coordinations appear in sequence (β-tin, simple cubic, fcc), indicating that they are not so far in energy;
- the liquid is a metal, and is more dense than the solid (anomalous).

These features alone guarantee that constructing a potential for Si is not a trivial task. Nevertheless, due to its technological importance, many groups have challenged themselves in this territory, with various degrees of success.

4.5.1 The Stillinger-Weber potential

The Stillinger-Weber potential [30] is one of the first attempts to model a semiconductor with a classical model. It is based on a two-body term and a three-body term:

$$V = \frac{1}{2}\sum_{ij}\phi(r_{ij}) + \sum_{ijk} g(r_{ij})g(r_{ik})\left(\cos\theta_{jik} + \frac{1}{3}\right)^2 \tag{4.14}$$

where θ_{jik} is the angle formed by the ij bond and the ik bond, and $g(r)$ is a decaying function with a cutoff between the first- and the second-neighbor shell. The intent is obvious: favor those configuration where $\cos\theta_{jik} = -1/3$, that is, where angles are as close as possible to those found in the diamond-like tetrahedral structure, and make this structure more stable than compact structure, as it should.

This potential gives a fairly realistic description of crystalline silicon. However, its built-in tetrahedral bias creates problems in other situations: it cannot predict the right energies of the non-tetrahedral polytypes found under pressure; the coordination of the liquid is too low; surface structures are not correct. These are called *transferability problems*: it is difficult to use the potential under conditions different from those it was designed for. Recent research [26] shows that, to build more realistic models, one should consider analytic forms which take into account the concept of *local environment*, with bond strengths depending on it.

Nevertheless, potentials based on similar concepts (using only geometrical quantities such as distances and bond angles as variables) are also currently used in simulations of organic systems, where they are usually called *force fields*. In spite of the limited transferability, which makes it difficult to change the topology of a molecule by breaking and reforming bonds, or to work at high temperatures or pressures, they are capable of modeling with a very high precision the structural and dynamical properties of a large variety of molecules, and are therefore vastly used in research and industry.

4.5.2 The Tersoff potential

The family of potentials developed by Tersoff [31], are based on the concept of *bond order*: the strength of a bond between two atoms is not constant, but depends on the local environment. This idea is similar to that of the "glue model" for metals, to use the coordination of an atom as the variable controlling the energy. In semiconductors, however, the focus is on bonds rather than atoms: that is where the electronic charge is sitting in covalent bonding.

At first sight, a Tersoff potential has the appearance of a pair potential:

$$V = \frac{1}{2}\sum_{ij} \phi_R(r_{ij}) + \frac{1}{2}\sum_{ij} B_{ij}\phi_A(r_{ij}) \tag{4.15}$$

where R and A mean "repulsive" and "attractive". However, it is not a pair potential because B_{ij} is not a constant. In fact, it is the *bond order* for the bond joining i and j, and it is a decreasing function of a "coordination" G_{ij} assigned to the bond:

$$B_{ij} = B(G_{ij}). \tag{4.16}$$

G_{ij} is in turn defined as

$$G_{ij} = \sum_{k} f_c(r_{ik}) g(\theta_{jik}) f(r_{ij} - r_{ik}) \tag{4.17}$$

where $f_c(r)$, $f(r)$ and $g(\theta)$ are suitable functions. The basic idea is that the bond ij is weakened by the presence of other bonds ik involving atom i. The amount of weakening is determined by where these other bonds are placed. Angular terms appear necessary to construct a realistic model.

This scheme works in a broader spectrum of situations than the Stillinger-Weber potential, however it is not exempt from problems. One of the biggest problems is perhaps that the fit is difficult: with 6 functions to fit and angular terms, finding a good parametrization is not an easy task.

These potentials have also been applied to hydrocarbons [32].

4.6 Long-range forces

So far it has always been assumed that interactions are short ranged, that is, that the effect of atom j on the force perceived by atom i vanishes when their separation r_{ij} is larger than a cutoff distance R_c. While this is a reasonable assumption in many cases, there are circumstances where long-range forces are crucial and must be taken into account. For instance:

- ions are charged, as, say, in a simulation of a salt like NaCl. In this case there is a Coulomb interaction ($\sim 1/r$) between ions
- individual particles are neutral but polarized, such as in water. There will be dipole-dipole interactions ($\sim 1/r^3$)
- van der Waals attractions between atoms ($\sim 1/r^6$), arising from induced dipole moments and often neglected in simulations, become extremely important under particular geometric conditions. For instance, the tip of a scanning tunneling microscope is attracted towards a surface tens of Angstroms underneath by van der Waals attraction.

In these cases, there is no cutoff distance and, more importantly, each particle interacts with *all* the images of the other particles in the nearby MD boxes: the "minimum image rule" introduced in 2.2.1 breaks down. A particle would also interact with its own images.

A known method to deal with long-range forces is that of Ewald sums. Briefly, and assuming we are dealing with charged ions, one assumes that each ion is surrounded by a spherically symmetric charge distribution of opposite sign which neutralizes the ion. This distribution is localized, that is, it is contained within a cutoff distance. This effectively screens ion-ion interactions, which are then treated with conventional short-range techniques.

To restore the original system, one then considers the Coulomb interaction of similar charge distributions centered on the ions, but now with the *same sign* as the original ions. These distributions exactly cancel those that we have considered before, so that the total interactions that we shall obtain will be those of the original system made of point charges only. The interaction of the cancelling distributions is computed in reciprocal space. The required Fourier transforms are particularly simple when the (arbitrary) shape of the distributions is chosen to be a gaussian.

The interested reader is referred to ref. [3], §5.5, for more details. An extension of Finnis-Sinclair potentials for metals to incorporate van der Waals forces has been made by Sutton and Chen [33].

4.7 But do we really need potentials?

About ten years ago, Car and Parrinello [34] developed a powerful method allowing to perform MD simulations where the ions are still moved classically, but under the action of forces obtained by solving the electronic structure problem, thus eliminating the need for empirical potentials at the expense of much larger computational requirements. This important development spawned one of the most important research lines in today's condensed matter physics, known as *first-principles* (or *ab initio*) *molecular dynamics* (FPMD), or simply as Car-Parrinello method.

Will the existence of the certainly more accurate FPMD technique imply the death of classical molecular dynamics based on empirical potentials? It is safe to say that the answer is no. While at present classical MD is being run on systems containing million of particles on the most powerful computers, or for simulation times of the order of nanoseconds (10^{-9} s), the current limit for FPMD is of the order of a thousand atoms, and its typical simulation times are measured in picoseconds (10^{-12} s). This means that several problems requiring large sizes and/or large times—such as many of those mentioned in §1.4—can be attacked only by classical methods. Even if the speed of computers keeps increasing at a breathtaking pace, so does the size of problems of interest, and it is very likely that all the MD techniques that we know today will remain in use for quite some time from now, and new ones may appear.

In fact, classical MD and Car-Parrinello MD already represent the extremes of a spectrum of MD methods, differing in the degree of approximation used to solve the electronic problem in order to obtain atomic forces. In particular, a promising area is that of tight-binding molecular dynamics (TBMD), where systems containing thousand of particles are now being simulated thanks to the recent development of $O(N)$-scaling computer codes.

The reader that wants to be introduced to the fast-growing areas of simulations from first-principles could start from the review articles present in the literature [35, 36, 37].

4.8 Fitting to ab initio data by "force matching"

We have seen in §4.5.1 how analytical forms can become pretty complicated when we strive for realism in modeling. Parametrizing the function appearing in a complex Hamiltonian can then become a tiresome task, and people often resort to very simple parametrizations. For instance, Tersoff potentials have only 12 parameters: an average of 2 parameters to describe each of the 6 functions constituting the model.

A first-principles molecular dynamics program can easily produce huge quantities of useful and reliable data, such as forces acting on atoms in a variety of different geometries and coordinations. These data can complement the experimental quantities normally used to fit potentials, helping to construct more realistic potentials. Recently, a scheme called "force matching method" [38] has been developed. It consists of a numerical optimization procedure which tries to match as closely as possible ab initio-derived forces with a classical potential, using a rich parametrization (10-15 parameters for each function in the potential). By explicitly including different geometries and different temperatures in the data set, one can attack the transferability problem at its very heart. One can says that the potential is constructed by *learning* from first-principles.

Several groups are now working on this front, so that potentials for classes of materials so far unexploited by classical molecular dynamics are likely to appear in the next years.

Appendix A

Choosing hardware and software

A.1 What kind of computer?

Compared with other applications in today's computational condensed matter physics, MD simulations using classical potentials are not particularly demanding, unless one's goal is to treat systems of the order of 10^6 atoms. For a typical system containing 10000 atoms—a size suitable to tackle a large variety of problems—and involving short-range potentials, one can estimate a total memory occupation of the running program between 8 and 16 megabytes (MB). This could fit well in a typical current workstation with a RISC processor[1], including entry-level machines.

Using code which scales linearly with the number of particles (as described later), a RISC CPU can yield a total simulation time of hundred of picoseconds or perhaps a few nanoseconds in a week of production, which is a reasonable rate for productive work.

In fact, a popular choice at many sites running different applications is to dedicate the smallest machines to classical MD, leaving the biggest machines with larger amounts of memory to the more demanding electronic structure codes.

A.1.1 Selecting a computer for molecular dynamics

Technology advances at an astounding rate. Every few months new, more powerful machines appear on the market. The market situation is therefore in continuous evolution, and one should always try to decide what to buy as closely as possible to the day at which the purchase is actually made.

Trying to select the "best computer for MD" is never easy. The choice is always conditioned by various factor, such as

[1]The name RISC (Reduced Instruction Set Computer) denotes an architecture for computer processor units which flourished in the last decade, consisting of reducing the number of different machine instructions, and optimizing their execution time—possibly at the level of one clock cycle per executed instruction.

- Size of physical problems to study
- Will the machine be used to run other kinds of programs?
- Is fast graphics important? (The site may have another machine specifically dedicated to visualization)
- "Friendliness" of operating systems, compilers and development tools
- Good customer and service support given by the company in your area
- Existence of other similar machines at your site, which would decrease the system management duties

Obviously, one has to carefully weight these factors keeping into account the peculiarities of the local situation.

Nevertheless, *performance* remains perhaps the most important point! To evaluate performance, it is common to use *benchmarks*: programs which accomplish a certain task (sometimes very simple), are never (or seldom) modified, and are run in controlled conditions. When running a benchmark on a machine for evaluation, the execution time is measured, and the output is compared with a *reference output* to make sure that the machine produced the correct result.

There are many scientific benchmarks around. The most well known are the Linpack[2] benchmark, and the SPEC[3] program suite. A problem is that different machines behave differently on different programs. In other words, the *ratio* between the execution time of two different benchmarks can vary widely between one machine architecture and another. Linpack, for instance, is a linear algebra benchmark. Machines which are able to perform matrix multiplications very efficiently—perhaps due to a special organization of their floating point unit or to compilers highly optimized for this problem—will obtain a very good Linpack score. While extremely representative of an important class of physical problems (for instance those in which diagonalizations are involved), this is of limited usefulness to assess MD performance, since there is no linear algebra in classical MD!

The other extreme, and by far the best choice, is to *run yourself* a copy of the program you are going to use on the machines you are interested in. This will not only give you an idea of a machine's performance, but also of *how easy* is using it. However, this is often impractical to do, due to the need to find access to these computers, and the time to run the program on each of them and verify the results.

To partially relieve this problem, the author developed in 1988 a molecular dynamics benchmark called MDBNCH[4]. A database of execution times on several machines has been collected over the years, and it keeps growing thanks to the contribution of many people. Keeping in mind that this benchmark may not exactly represent the applications which you are interested to run, it is

[2]http://netlib.bell-labs.com/

[3]http://www.specbench.org/

[4]http://www.sissa.it/furio/mdbnch.html

probably an indicator of *molecular dynamics performance* more accurate than the more general benchmarks available around.

A.1.2 Storage space

On the other hand, the need for disk space should not be underestimated. Storing positions, velocities and accelerations of 10000 atoms in 64-bit precision takes 720 KB. In a typical project, a large number of such configurations are saved for later reference, and additional statistical quantities are often also computed, making it easy to fill hundreds of MB of mass storage. Projects involving dynamical analysis over fine-grained trajectory files, such as calculations of vibrational spectra, are even more demanding. While the use of tapes is always possible, availability of a comfortable amount of online storage is an important ingredient to work smoothly and efficiently.

A.2 What language?

A.2.1 High level languages

A *computer language* is an interface between man and machine.

Machines execute sequences of instructions belonging to their instruction set. Such instructions are rather simple and tightly bound to features of the machine hardware. For instance, "load into register A the contents of memory location 00900752", or "multiply the numbers contained into registers A and B, leaving the result in register C" could be two examples of instructions at the machine level. A program written in machine language (or *assembler*) would not be easily transferable to a computer produced by a different manufacturer. Moreover, such a program would be very hard to decipher for a human examining it.

So called *high level computer languages* are the solution. A language is a machine-independent way to specify the sequence of operations necessary to accomplish a task. A language can be designed to express in a concise way a "common" sequence of operations. A line in a high level language can embody powerful operations, and correspond to tens, or hundreds, of instructions at the machine level. Of course, what is "common" depends on the kind of problem at hand. For this reason, many different languages exist in the computer world, aimed at solving problems in different areas.

The *compiler* is the tool to convert a program written in a high level language into the sequence of machine instructions required by a specific computer to accomplish the task. Users typically control details of the operation of compilers by means of options supplied on the command line, or directives embedded in the program source, but they seldom need to examine the resulting machine language code.

However, one should not forget the fact that programs are also an interface with humans, and *a way to communicate between humans.* A program carries with it the information of *how* a problem is being solved, and it is extremely important for this information to be presented in a clear, complete way to people examining the code. In the scientific world, quite often the source code is also the main documentation for the program. For this reason, it is very important to have in mind people needs—and not only machines needs—when writing a program. In practice, this means writing informative comments in the code, using meaningful names for symbols, avoiding "dirty tricks" unless absolutely necessary. This documenting activity should always be done at the time of code writing, not deferred to a later time that may never come.

A.2.2 Fortran or C?

Today, the two more popular languages for numeric computation are *Fortran* and *C*.

Fortran has a long tradition in computational physics, chemistry and engineering. Born at the end of the 50s, during the 60s and the 70s it became the most popular choice for the development of numerically intensive code. An enormous body of knowledge has been embedded in Fortran codes.

C is a more recent language. It was developed in the 70s and provides the foundation for the Unix operating system. The popularity of Unix is a direct consequence of the fact that it is largely written in C, and therefore easily portable between different brands of computers. C is a simple still very powerful language, and its scope is rather broad: the solution to problems in very different areas has been coded in C with success.

Languages evolve much more slowly than computer architectures. The main reason for that is that we can easily upgrade to new computers: what we have to do is to recompile our programs with the compiler supplied with the new machine. Changing language is a much more expensive and painful business: we must rewrite our codes, and make sure that the new version does what it is supposed to do. However, advances in computer systems often create a need for new features in a language.

For this reason, even a stable and well-proven language such as Fortran undergoes periodic revisions in order to keep it up to date. Fortran was revised in 1966 (Fortran 66), then 1977 (Fortran 77), and more recently in 1990 (Fortran 90). At present, there is an immense amount of scientific codes written in Fortran 77. Many of them are of public domain, and among them there are libraries of subroutines for solving a large variety of problems.

Both Fortran 90 and C are perfectly valid choices for a computationally intensive application. Fortran 77 lacks many features which are crucial today, and should be regarded as an obsolescent language.

In favor of C:

- efficient compilers are universally available for all the main architectures in use, and a good public compiler also exists (*gcc*). C compilers often come free with machines, while Fortran 90 compilers must be purchased, and are often expensive

- C is very broad in scope, and is more powerful of Fortran 90 in some areas, such as pointers and manipulation of strings of characters

- acquired coding experience can be directly used outside the scientific world: C is very common in the commercial world.

In favor of Fortran 90:

- language designed with easiness of coding numerical problems as one of the main goals (example: array operations)
- language designed with high performance in numerical applications as one of the main goals
- interfacing with previously developed Fortran code or directly reusing portions of existing Fortran code is easier
- No need to master simultaneously two languages when developing new codes while maintaining old Fortran 77 programs at the same time
- better portability between different architectures (for instance, between 32- and 64-bit machines)

The final choice can be in favor of either one of them, depending to the weight assigned to the various factors.

For this writer, the advantages of Fortran 90 outweigh those of C, and the code examples in this notes are written in Fortran 90. Some will disagree with this choice.

Bibliography

[1] D. W. Heermann, *Computer simulation methods*, Springer, 1986.

[2] *Molecular dynamics simulation of statistical-mechanical systems*, G. Ciccotti and W. G. Hoover (Eds.), North-Holland, 1986. Contains chapters written by experts of the field on many aspects of simulation. A place to dig in into specific topics, with plenty of technical details.

[3] M. P. Allen and D. J. Tildesley, *Computer simulation of liquids*, Oxford, 1987. This is a classic *how to* book, containing a large number of technical details on a vast array of techniques.

[4] R. W. Hockney and J. W. Eastwood, *Computer simulation using particles*, IOP Publishing, 1988. This book mostly addresses simulation methods where particles are immersed in an external field as well as with each other. Such methods are commonly used for instance in plasma physics. A section on MD simulations on liquids and solids is however present, and the discussion of the cell method in section 8-4 is particularly useful.

[5] W. G. Hoover, *Computational statistical mechanics*, Elsevier, 1991. This book focuses on the foundations of simulation from the statistical mechanics point of view, and contains interesting chapters on ensembles, hydrodynamics, non-equilibrium molecular dynamics, with several examples.

[6] J. M. Haile, *Molecular dynamics simulation*, Wiley, 1992. An excellent and recent general introduction to molecular dynamics. Not as thorough as [3] in explaining techniques, but very deep in explaining the fundamentals, including "philosophical" aspects. Probably the best starting point. A good amount of Fortran 77 code is included.

[7] D. C. Rapaport, *The art of molecular dynamics simulation*, Cambridge Univ. Press, 1995. Excellent recent "recipe book", describing in detail a large number of techniques, and containing a large amount of C code. The book has a home page[5].

[8] *Simulation of liquids and solids*, G. Ciccotti, D. Frenkel and I. R. McDonald (Eds.), North-Holland, 1987. A collection of key papers on simulation from the origins to 1986.

[5]http://www.cup.cam.ac.uk/onlinepubs/ArtMolecular/ArtMoleculartop.html

[9] W. W. Wood, in [2], p. 3.

[10] B. J. Alder and T. E. Wainwright, J. Chem. Phys. **27**, 1208 (1957).

[11] J. B. Gibson, A. N. Goland, M. Milgram, and G. H. Vineyard, Phys. Rev. **120**, 1229 (1960).

[12] A. Rahman, Phys. Rev. **136**, A405 (1964).

[13] L. Verlet, Phys. Rev. **159**, 98 (1967); Phys. Rev. **165**, 201 (1967).

[14] J.-P. Hansen and L. Verlet, Phys. Rev. **184**, 151 (1969).

[15] S. Yip, in [2], p. 523.

[16] H. J. C. Berendsen, in [2], p. 496; Comp. Phys. Commun. **44**, 233 (1987).

[17] J. P. Hansen and I. R. McDonald, *Theory of simple liquids*, 2nd Ed., Academic, 1986. A classic book on liquids, with a chapter devoted to computer simulation.

[18] See for instance C. Kittel, *Introduction to solid state physics*, Wiley.

[19] H. J. C. Berendsen and W. F. van Gunsteren, in [2], p. 43.

[20] Interestingly, one can find a description of the leap-frog algorithm in R. P. Feynman, R. B. Leighton and M. Sands, *The Feynman Lectures on Physics*, Vol. 1, Addison-Wesley, 1963, Chapter 9 ("Newton's Laws of Dynamics").

[21] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, Science **220**, 671 (1983).

[22] H. C. Andersen, J. Chem. Phys. **72**, 2384 (1980).

[23] M. Parrinello and A. Rahman, Phys. Rev. Lett. **45**, 1196 (1980); J. Appl. Phys. **52**, 7158 (1981); M. Parrinello, in [2], p. 204.

[24] S. Nose, Molec. Phys. **52**, 255 (1984); W. G. Hoover, Phys. Rev. A **31**, 1695 (1985).

[25] F. Ercolessi, M. Parrinello, and E. Tosatti, Philos. Mag. A **58**, 213 (1988).

[26] A. Carlsson, Solid State Phys. **43**, 1 (1990).

[27] M. S. Daw and M. I. Baskes, Phys. Rev. B **29**, 6443 (1984); S. M. Foiles, M. I. Baskes, and M. S. Daw, Phys. Rev. B **33**, 1986.

[28] M. W. Finnis and J. E. Sinclair, Philos. Mag. A **50**, 45 (1984).

[29] K. W. Jacobsen, J. K. Nørskov and M. J. Puska, Phys. Rev. B **35**, 7423 (1987).

[30] F. Stillinger and T. A. Weber, Phys. Rev. B **31**, 5262 (1985).

[31] J. Tersoff, Phys. Rev. B **37**, 6991 (1988).

[32] D. W. Brenner, Phys. Rev. B **42**, 9458 (1990).

[33] A. P. Sutton and J. Chen, Philos. Mag. Lett. **61**, 139 (1990).

[34] R. Car and M. Parrinello, Phys. Rev. Lett. **55**, 2471 (1985).

[35] D. K. Remler and P. A. Madden, Molec. Phys. **70**, 921 (1990).

[36] M. C. Payne, M. P. Teter, C. C. Allan, T. A. Arias, and J. D. Joannopoulos, Rev. Mod. Phys. **64**, 1045 (1992).

[37] G. Galli and A. Pasquarello, in *Computer simulation in chemical physics*, M. P. Allen and D. J. Tildesley (Eds.), Kluwer, 1993, p. 261.

[38] F. Ercolessi and J. B. Adams, Europhys. Lett. **26**, 583 (1994).