8

Molecular dynamics simulations

8.1 Introduction

In the previous chapter we saw that the experimental values of physical quantities of a many-particle system can be found as an ensemble average. Experimental systems are so large that it is impossible to determine this ensemble average by summing over all the accessible states in a computer. There exist essentially two methods for determining these physical quantities as statistical averages over a restricted set of states: the molecular dynamics and Monte Carlo methods. Imagine that we have a random sample of, say, 10^7 configurations of the system which are all compatible with the values of the system parameters. For such a large number we expect averages of physical quantities over the sample to be rather close to the ensemble average. It is unfortunately impossible to generate such a random sample; however, we can generate a sample consisting of a large number of configurations which are determined successively from each other and are hence correlated. This is done in the molecular dynamics and Monte Carlo methods. The latter will be described in Chapter 10.

Molecular dynamics is a widely used method for studying classical many-particle systems. It consists essentially of integrating the equations of motion of the system numerically. It can therefore be viewed as a simulation of the system as it develops over a period of time. The system moves in phase space along its physical trajectory as determined by the equations of motion, whereas in the Monte Carlo method it follows a (directed) random walk. The great advantage of the MD method is that it not only provides a way to evaluate expectation values of static physical quantities; dynamical phenomena, such as transport of heat or charge, or relaxation of systems far from equilibrium can also be studied.

In this section we discuss the general principles of the molecular dynamics method. In the following sections more details will be given and special techniques will be discussed. There exists a vast research literature on this subject and there are some review papers and books [1–5].

Consider a collection of N classical particles in a rectangular volume $L_1 \times L_2 \times L_3$. The particles interact with each other, and for simplicity we shall assume that the interaction force can be written as a sum over pair forces, $\mathbf{F}(r)$, whose magnitude depends only on the distance, r, between the particle pairs and which is directed between them (see also the previous chapter). In that case the internal force (i.e. the force due to interactions between the particles) acting on particle number i is given as

$$\mathbf{F}_i(R) = \sum_{\substack{j=1,N;\\ j\neq i}} F(|\mathbf{r}_i - \mathbf{r}_j|)\hat{\mathbf{r}}_{ij}. \tag{8.1}$$

R denotes the position coordinates $\mathbf{r}_i$ of all particles in the notation introduced in Section 7.2.1 (P denotes the momenta); $\hat{\mathbf{r}}_{ij}$ is a unit vector directed along $\mathbf{r}_j - \mathbf{r}_i$, pointing from particle i to particle j. In experimental situations there will be external forces in addition to the internal ones – examples are gravitational forces and forces due to the presence of boundaries. Neglecting these forces for the moment, we can use (8.1) in the equations of motion:

$$\frac{\mathrm{d}^2\mathbf{r}_i(t)}{\mathrm{d}t^2} = \frac{\mathbf{F}_i(R)}{m_i} \tag{8.2}$$

in which m_i is the mass of particle i. In this chapter we take the particles identical unless stated otherwise. Molecular dynamics is the simulation technique in which the equations (8.2) are solved numerically for a large collection of particles.

The solutions of the equations of motion describe the time evolution of a real system although obviously the molecular dynamics approach is approximate for the following reasons.

- First of all, instead of a quantum mechanical treatment we restrict ourselves to a classical description for the sake of simplicity. In Chapter 9, we shall describe a method in which ideas of the density functional description for quantum many-particle systems (Chapter 5) are combined with the classical molecular dynamics approach. The importance of the quantum effects depends strongly on the particular type of system considered and on the physical parameters (temperature, density …).
- The forces between the particles are not known exactly: quantum mechanical calculations from which they can be determined are subject to systematic errors as a result of the neglect of correlation effects, as we have seen in previous chapters. Usually these forces are given in a parametrised form, and the parameters are determined either by *ab initio* calculations or by fitting the results of simulations to experimental data. There exist systems for which the forces are known to high precision, such as systems consisting of stars and galaxies at large mutual distances and at nonrelativistic velocities where the interaction is largely dominated by Newton's gravitational $1/r^2$ force.

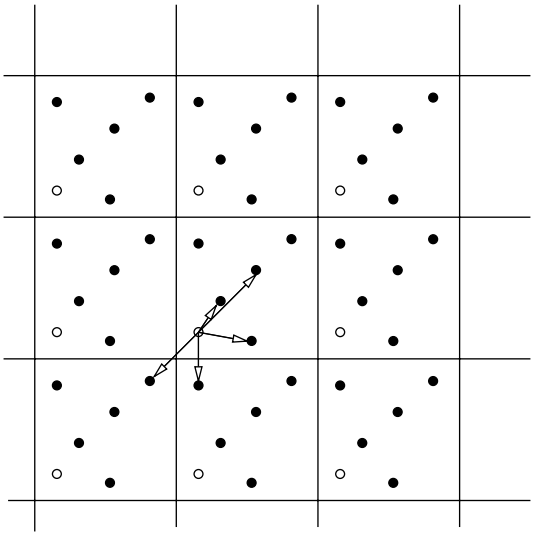

Figure 8.1. Periodic boundary conditions for molecular dynamics. Each particle interacts not only with every other particle in the system but also with all other particles in the copies of the system. The arrows from the white particle point to the nearest copies of the other particles in the system.

- Another approximation is inherent to most computer simulations aiming at a description of the real world: the system sizes in such simulations are much smaller than those of experimental systems. In the limit where the correlation length is much smaller than the system size this does not matter too much, and in the opposite regime, in which the correlation length exceeds the system size we can use the finite-size scaling methods discussed in Chapter 5 in order to extrapolate results for physical quantities in the finite system to those of the infinite system (although second order transitions are seldom studied in molecular dynamics because of the heavy demands on computing resources). The finiteness of the system size is felt through the presence of the boundary. The convention adopted in the vast majority of molecular simulations is to use periodic boundary conditions (PBC) as it is assumed that for these boundary conditions the behaviour of the system is most similar to that of a system of the same size embedded in an infinite system. In fact, with periodic boundary conditions the system of interest is surrounded by similar systems with exactly the same configuration of particles at any time (see Figure 8.1). The interaction between two particles i and j is then given by the following expression:

$$\mathbf{F}_{\mathrm{PBC}}(\mathbf{r}_i - \mathbf{r}_j) = \sum_{\mathbf{n}} \mathbf{F}\left(\left|\mathbf{r}_i - \mathbf{r}_j + \sum_{\mu=1}^{3} \mathbf{L}_\mu n_\mu\right|\right) \tag{8.3}$$

where $\mathbf{L}_\mu$ are vectors along the edges of the rectangular system volume and the first sum on the right hand side is over all vectors $\mathbf{n}$ with integer coefficients n_μ. The force $\mathbf{F}$ is directed along the line connecting particle i and the image particle $\mathbf{r}_j - \sum_{\mu=1}^{3} \mathbf{L}_\mu n_\mu$ according to the convention of Eq. (8.1). Of course, calculating terms of this infinite sum until convergence is achieved is a time-consuming procedure, and in the next section we shall consider techniques for approximating this sum efficiently.

- The time average must obviously be evaluated over a finite time. For liquid argon, which is the most widely studied system in molecular dynamics because simple Lennard–Jones pair forces yield results which are in very good agreement with experiment, the typical time step used in the numerical integration of the equations of motion is about 10^{-14} seconds, which means that for the $\sim 10^5$ integration steps which can usually be carried out in a reasonable amount of computer time, the total simulation is restricted to about 10^{-9} seconds. The correlation time of the system should therefore be much smaller than this. There is also a limitation in time because of the finite size of the system. This might in principle become noticeable when the particles have travelled on average more than half the linear system size, but in practice such effects occur at much longer time scales, of the order of the *recurrence time*, the time after which the system returns to the initial configuration (in continuum mechanics, this is called the *Poincaré time*).
- The numerical integration algorithm is not infinitely accurate. This forces us to make some optimum choice between speed and accuracy: the larger the integration time step, the more inaccurate the results of the simulation. In fact, the system will follow a trajectory in phase space which deviates from the trajectory it would follow in reality. The effect on the physical quantities as measured in the simulation is of course related to this deviation in the course of time.

We may summarise by saying that MD is – in principle – a direct simulation of a many-particle system but we have seen that, just as with any computational technique in physics, MD simulations must be carried out with considerable care. It is furthermore advisable to carry out reference tests for systems for which exact results exist or for which there is an extensive literature for comparison.

8.2 Molecular dynamics at constant energy

In the previous section we sketched the molecular dynamics method briefly for the simplest case in which the equations of motion for a collection of particles are solved for forces depending on the relative positions of the particles only. In that case energy and momentum are conserved.[1] Trivially, the particle number and system volume are

[1] The angular momentum is not conserved because of the periodic boundary conditions breaking the spherical symmetry of the interactions.

conserved too, so the time averages of physical quantities obtained by this type of simulation are equivalent to averages in the microcanonical or (*NVE*) ensemble. In this section we describe the microcanonical MD method in more detail.

The algorithm of a standard MD simulation for studying systems in equilibrium is the following:

- Initialise;
- Start simulation and let the system reach equilibrium;
- Continue simulation and store results.

We will now describe these main steps in more detail.

Initialise: The number of particles and the form of the interaction are specified. The temperature is usually of greater interest than the total energy of the system and is therefore usually specified as an input parameter. We shall see below how the system can be pushed toward the desired temperature.

The particles are assigned positions and momenta. If a Lennard–Jones potential is used, the positions are usually chosen as the sites of a Bravais-fcc lattice, which is the ground state configuration of the noble gases like argon (although the Lennard–Jones system is hexagonal close-packed in the ground state [6]). The fcc lattice contains four particles per unit cell, and for a cubic volume the system contains therefore $4M^3$ particles, $M = 1, 2, \ldots$ This is the reason why MD simulations with Lennard–Jones interactions are often carried out with particle numbers 108, 256, 500, 864, ….

The velocities are drawn from a Maxwell distribution with the specified temperature. This is done by drawing the x, y and z velocity components for each particle from a Gaussian distribution; for the x-component of the velocity this distribution is $\exp[-mv_x^2/(2k_\mathrm{B}T)]$. In Appendix B3 it is described how random numbers with a Gaussian distribution can be generated. After generating the momenta, the total momentum is made equal to zero by calculating the average momentum $\bar{\mathbf{p}}$ per particle, and then subtracting an amount $\bar{\mathbf{p}}$ from all the individual momenta $\mathbf{p}_i$.

Start simulation and let the system reach equilibrium: The particles being released from fcc lattice positions, the system is generally not in equilibrium and during the initial phase of the simulation it is given the opportunity to relax. We now describe how the integration of the equations of motion is carried out and how the forces are evaluated. Finally we shall explain how in this initial phase the desired temperature is arrived at.

Numerical algorithms for molecular dynamics will be considered in detail in Section 8.4. Suffice it here to mention briefly the most widely used algorithm which is simple and reliable at the same time – the Verlet algorithm (see also Appendix A7.1). The standard form of the Verlet algorithm for the integration of the equation of motion of a single particle subject to a force $\mathbf{F}$ depending only on the position of the particle reads

$$\mathbf{r}(t+h) = 2\mathbf{r}(t) - \mathbf{r}(t-h) + h^2\mathbf{F}[\mathbf{r}(t)]/m \tag{8.4}$$

where $\mathbf{r}(t)$ is the position of the particle at time $t = nh$ (h is the time step; n is an integer). From now on we choose units such that $m = 1$. The error per time step is of order h^4 and a worst case estimate for the error over a fixed time interval containing many time steps is of order h^2 (see Problem A3). To start up the algorithm we need the positions of the particles at two subsequent time steps. As we have only the initial ($t = 0$) positions and velocity at our disposal, the positions at $t = h$ are calculated as

$$\mathbf{r}(h) = \mathbf{r}(0) + h\mathbf{v}(0) + \frac{h^2}{2}\mathbf{F}[\mathbf{r}(t=0)] \quad (m \equiv 1), \tag{8.5}$$

with an error of order h^3.

During the integration, the velocities can be calculated as

$$\mathbf{v}(t) = \frac{\mathbf{r}(t+h) - \mathbf{r}(t-h)}{2h} + \mathcal{O}(h^2). \tag{8.6}$$

When using periodic boundary conditions in the simulation, we must check for each particle whether it has left the simulation cell in the last integration step. If this is the case, the particle is translated back over a lattice vector $\mathbf{L}_\mu$ to keep it inside the cell (we shall see below that this procedure facilitates the common procedure for evaluating the forces with periodic boundary conditions). The velocity must obviously be determined before such a translation.

There exist two alternative formulations of the Verlet algorithm, which are exactly equivalent to it in exact arithmetic but which are less susceptible to errors resulting from finite numerical precision in the computer than the original version. The first of these, the *leap-frog* form, introduces the velocities at time steps precisely in between those at which the positions are evaluated:

$$\mathbf{v}(t + h/2) = \mathbf{v}(t - h/2) + h\mathbf{F}[\mathbf{r}(t)], \tag{8.7a}$$

$$\mathbf{r}(t + h) = \mathbf{r}(t) + h\mathbf{v}(t + h/2). \tag{8.7b}$$

These steps are then repeated over and over. Note that they must always be applied in the given order: the second step uses $\mathbf{v}(t + h/2)$ which is calculated in the first step.

Another form is the so-called velocity-Verlet algorithm [7] which is also more stable than the original Verlet form and which, via the definition

$$\mathbf{v}(t) = \frac{\mathbf{r}(t+h) - \mathbf{r}(t-h)}{2h} \tag{8.8}$$

evaluates velocities and positions at the same time instances:

$$\mathbf{r}(t + h) = \mathbf{r}(t) + h\mathbf{v}(t) + h^2\mathbf{F}(t)/2, \tag{8.9a}$$

$$\mathbf{v}(t + h) = \mathbf{v}(t) + h[\mathbf{F}(t + h) + \mathbf{F}(t)]/2. \tag{8.9b}$$

This form is most convenient because it is very stable with respect to errors due to finite precision arithmetic, and it does not require additional calculations in order to find the velocities. It should be noted that all formulations have essentially the same memory requirements. It may seem that, as this algorithm needs *two* forces the second step, we need two arrays for these, one containing $\mathbf{F}(t)$ and the other $\mathbf{F}(t+h)$. However, the following form of the algorithm is exactly equivalent and avoids the need for two force arrays:

$$\tilde{\mathbf{v}}(t) = \mathbf{v}(t) + h\mathbf{F}(t)/2, \tag{8.10a}$$

$$\mathbf{r}(t+h) = \mathbf{r}(t) + h\tilde{\mathbf{v}}(t), \tag{8.10b}$$

$$\mathbf{v}(t+h) = \tilde{\mathbf{v}}(t) + h\mathbf{F}(t+h)/2. \tag{8.10c}$$

The new force $\mathbf{F}(t+h)$ is calculated between the second and third step.

The force acting on particle i results from the interaction forces between this particle and all the other particles in the system – usually pair-wise interactions are used. The calculation of the forces therefore takes a relatively long time as this requires $\mathcal{O}(N^2)$ steps. A problem in the evaluation of the force arises from the assumption of periodic boundary conditions. These imply that the system is surrounded by an infinite number of copies with exactly the same configuration as in Figure 8.1. A particle therefore interacts not only with each partner j in the system cell we are considering but also with the images of particle j in all the copies of the system. This means that in principle an infinite number of interactions has to be summed over. In many cases, the force decays rapidly with distance, and in that case remote particle copies will not contribute significantly to the force. If the force between the particles can safely be neglected beyond separations of half the linear system size, the force evaluation can be carried out efficiently by taking into account, for each particle in the system, only the interactions with the nearest copy of each of the remaining particles (see Figure 8.1): each infinite sum over all the copies is replaced by a single term! This is the *minimum image convention*. In formula, for a cubic system cell the minimum image convention reads

$$r_{ij}^{\min} = \min_{\mathbf{n}} |\mathbf{r}_i - \mathbf{r}_j + n_\mu \mathbf{L}_\mu| \tag{8.11}$$

with the same notation as in Eq. (8.3), but where the components of n_μ assume the values $0, \pm 1$, provided all the particles are kept within the system cell, by translating them back if they leave this cell. The potential is no longer analytic in this convention, but discontinuities will obviously be unimportant if the potential is small beyond half the linear system size.

Often it is possible to cut the interactions off at a distance $r_{\text{cut-off}}$ smaller than half the linear system size without introducing significant errors. In that case the forces do not have to be calculated for all pairs. However, all pairs must be considered to

check whether their separation is larger than $r_{\text{cut-off}}$. In the same paper in which he introduced the midpoint integration algorithm into MD, Verlet [8] proposed keeping a list of particle pairs whose separation lies within some maximum distance $r_{\max}$ and updating this list at intervals of a fixed number of steps – this number lies typically between 10 and 20. The radius $r_{\max}$ is taken larger than $r_{\text{cut-off}}$ and must be chosen such that between two table updates it is unlikely for a pair not in the list to come closer than $r_{\text{cut-off}}$. If both distances are chosen carefully, the accuracy can remain very high and the increase in efficiency is of the order of a factor of 10 (the typical relative accuracy in macroscopic quantities in a MD simulation is of order 10^{-4}).

There exists another method for keeping track of which pairs are within a certain distance of each other: the *linked-cell method*. In this method, the system is divided up into (rectangular) cells. Each cell is characterized by its integer coordinates IX,IY,IZ in the grid of cells. The cell size is chosen larger than the interaction range which is about the size of $r_{\max} > r_{\text{cut-off}}$ in the Verlet method. If we wanted a list of particles for each cell, we could simply restrict the interactions to particle pairs in the same, or in neighbouring cells. However, as particles will leave and enter the cells, the bookkeeping of these lists becomes a bit cumbersome. This bookkeeping can however be done very efficient by using a list of particle indices. The procedure is reminiscent of the use of pointers in a linked list. We need two ingredients: we must have a routine which generates a sort of table containing information about which particle is in what cell, and we need to organise the force calculation such that it uses this information.

To be specific, let us assume that there are $M \times M \times M$ cells. The particles are numbered 1 through N, so each particle has a definite index. We use an integer array called 'Header' which is of size $M \times M \times M$: Header(IX,IY,IZ) tells us the *highest* particle index to be found in cell IX,IY,IZ. We also introduce an integer array 'Link' which is of size N. The arrays Header and Link are filled in the following code: dimension header(M,M,M), link(N)

```
Set Header (IX,IY,IZ) to 0
Set Link(I) to 0
FOR  I = 1,N DO
    IX = int(M*x(I)/L)+1
    IY = int(M*y(I)/L)+1
    IZ = int(M*z(I)/L)+1
    link(i) = header(IX,IY,IZ)
    header(IX,IY,IZ) = I
END FOR
```

Now, Header contains the highest index present in all cells. Furthermore, for particle I, Link(I) is another particle *in the same cell*. To find all particles in cell IX, IY, IZ,

we look at Header(IX,IY,IZ) and then move down from particle I to the following by taking for the next particle the value Link(I). Using this in the force calculation leads to the pseudocode:

```
FOR all cells with indices (IX,IY,IZ) DO
{Fill the list xt, yt and zt with the particles of the central cell}
    icnt = 0;
    j = Header(IX,IY,IZ);
    WHILE (j>0) DO
        j = link(j);
        icnt = icnt + 1;
        xt(icnt) = x(j); yt(icnt) = y(j); zt(icnt) = z(j);
        LocNum = icnt;
    END WHILE
{Now, LocNum is the number of particles in the central cell}
    FOR half of the neighbouring cells DO
        Find particles in the same way as central cell
        and append them to the list xt, yt, zt;
    END FOR
    Calculate Lennard–Jones forces between all particles in the central cell;
    Calculate Lennard–Jones forces between particles in central and
      neighbouring cells;
END FOR
```

Note that we loop over only *half* the number of neighbouring cells in order to avoid double counting of particle pairs. The cell method is less efficient than the neighbour list method as the blocks containing possible interaction candidates for each particle substantially bigger than the spheres of the neighbour list. The advantage of the present method lies in its suitability for parallel computing (see Chapter 16).

Cutting off the force violates energy conservation although the effect is small if the cut-off radius is chosen suitably. To avoid this violation, the pair potential $U(r)$ can be shifted so that it becomes continuous at $r_{\text{cut-off}}$. The shifted potential can be written in terms of the original one as

$$U_{\text{shift}}(r) = U(r) - U(r_{\text{cut-off}}). \tag{8.12}$$

The force is not affected by this shift; it remains discontinuous at the cut-off and this gives rise to inaccuracies in the integration. Applying a shift in the force in addition to the shift in the potential yields [9, 10]

$$U_{\text{force shift}}(r) = U(r) - U(r_{\text{cut-off}}) - \frac{\mathrm{d}}{\mathrm{d}r}U(r_{\text{cut-off}})(r - r_{\text{cut-off}}) \tag{8.13}$$

and now the force and the potential are continuous. These adjustments to the potential can be compensated for by thermodynamic perturbation theory (see Ref. [11]).

Electric and gravitational forces decay as $1/r$ and cannot be truncated beyond a finite range without introducing important errors. These systems will be treated in Section 8.7.

The time needed to reach equilibrium depends on how far the initial configuration was from equilibrium, and on the relaxation time (see Section 7.4). To check whether equilibrium has been reached, it is best to monitor several physical quantities such as kinetic energy and pressure, and see whether they have levelled down. This can be judged after completing the simulation by plotting out the values of these physical quantities as a function of time. It is therefore convenient to save all these values on disk during the simulation and analyse the results afterwards. It is also possible to measure correlation times along the lines of Section 7.4, and let the system relax for a period of, for example, twice the longest correlation time measured.

A complication is that we want to study the system at a predefined temperature rather than at a predefined total energy because temperature is easily measurable and controllable in experimental situations. Unfortunately, we can hardly forecast the final temperature of the system from the initial configuration. To arrive at the desired value of the temperature, we rescale the velocities of the particles a number of times during the equilibration phase with a uniform scaling factor λ according to

$$\mathbf{v}_i(t) \to \lambda \mathbf{v}_i(t) \tag{8.14}$$

for all the particles $i = 1, \ldots, N$. The scaling factor λ is chosen such as to arrive at the desired temperature T_D after rescaling:

$$\lambda = \sqrt{\frac{(N-1)3k_\mathrm{B}T_\mathrm{D}}{\sum_{i=1}^{N} m v_i^2}}. \tag{8.15}$$

Note the factor $N - 1$ in the numerator of the square root: the kinetic energy is composed of the kinetic energies associated with the *independent* velocities, but as for interparticle interactions with PBC the total force vanishes, the total momentum is conserved and hence the number of independent velocity components is reduced by 3. This argument is rather heuristic and not entirely correct. We shall give a more rigorous treatment of the temperature calculation in Section 10.7.

After a rescaling the temperature of the system will drift away but this drift will become less and less important when the system approaches equilibrium. After a number of rescalings, the temperature then fluctuates around an equilibrium value. Now the 'production phase', during which data can be extracted from the simulation, begins.

Continue simulation and determine physical quantities: Integration of the equations of motion proceeds as described above. In this part of the simulation, the actual determination of the static and dynamic physical quantities takes place. We determine the expectation value of a static physical quantity as a time average according to

$$\overline{A} = \frac{1}{n - n_0} \sum_{\nu > n_0}^{n} A_\nu. \tag{8.16}$$

The indices ν label the n time steps of the numerical integration, and the first n_0 steps have been carried out during the equilibration. For determination of errors in the measured physical quantities, see the discussion in Section 7.4.

Difficulties in the determination of physical quantities may arise when the parameters are such that the system is close to a first or second order phase transition (see the previous chapter): in the first order case, the system might be 'trapped' in a metastable state and in the second order case, the correlation time might diverge for large system sizes.

In the previous chapter we have already considered some of the quantities of interest. In the case of a microcanonical simulation, we are usually interested in the temperature and pressure. Determination of these quantities enables us to determine the *equation of state*, a relation between pressure and temperature, and the system parameters – particle number, volume and energy (*NVE*). This relation is hard to establish analytically, although various approximate analytical techniques for this purpose exist: cluster expansions, Percus–Yevick approximation, etc. [11].

The pair correlation function is useful not only for studying the details of the system but also to obtain accurate values for the macrosopic quantities such as the potential energy and pressure, as we shall see below. The correlation function is determined by keeping a histogram which contains for every interval $[i\Delta r, (i + 1)\Delta r]$ the number of pairs $n(r)$ with separation within that range. The list can be updated when the pair list for the force evaluation is updated. The correlation function is found in terms of $n(r)$ as

$$g(r) = \frac{2V}{N(N-1)} \left[\frac{\langle n(r) \rangle}{4\pi r^2 \Delta r} \right]. \tag{8.17}$$

Similar expressions can be found for time-dependent correlation functions – see Refs. [2] and [11].

If the force has been cut off during the simulation, the calculation of average values involving the potential U requires some care. Consider for example the potential energy itself. This is calculated at each step taking only the pairs with separation within the minimum cut-off distance into account; taking all pairs into account would imply losing the efficiency gained by cutting off the potential. The neglect of the tail of the potential can be corrected for by using the pair correlation

function beyond $r_{\text{cut-off}}$:

$$\langle U \rangle = \langle U \rangle_{\text{cut-off}} + 2\pi \frac{N(N-1)}{V} \int_{r_{\text{cut-off}}}^{\infty} r^2 \mathrm{d}r \; U(r) g(r) \tag{8.18}$$

where $\langle \cdots \rangle_{\text{cut-off}}$ is the average restricted to pairs with separation smaller than $r_{\text{cut-off}}$. Of course, we can determine the correlation function for r up to half the linear system size only because of periodic boundary conditions. Verlet [12] has used the Percus–Yevick approximation to extrapolate g beyond this range. Often g is simply approximated by its asymptotic value $g(r) \equiv 1$ for large r.

Similarly, the virial equation is corrected for the potential tail:

$$\frac{P}{nk_{\mathrm{B}}T} = 1 - \frac{1}{3Nk_{\mathrm{B}}T} \left\langle \sum_i \sum_{j>i} r_{ij} \frac{\partial U(R)}{\partial r_{ij}} \right\rangle_{\text{cut-off}} - \frac{2\pi N}{3k_{\mathrm{B}}TV} \int_{r_{\text{cut-off}}}^{\infty} r^3 \frac{\partial U(r)}{\partial r} g(r) \mathrm{d}r, \tag{8.19}$$

where $g(r)$ can also be replaced by 1.

The specific heat can be calculated from Lebowitz's formula, see Eq. (7.37).

8.3 A molecular dynamics simulation program for argon

In the previous section we described the structure of a MD program and here we give some further details related to the actual implementation. The program simulates the behaviour of argon. In 1964, Rahman [13] published a paper on the properties of liquid argon – the first MD simulation involving particles with smoothly varying potentials. Previous work by Alder and Wainwright [14] was on hard sphere fluids. Rahman's work was later refined and extended by Verlet [8] who introduced several features that are still used, as we have seen in the previous section.

The Lennard–Jones pair potential turns out to give excellent results for argon:

$$U(r) = 4\varepsilon \left[\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^{6} \right]. \tag{8.20}$$

The optimal values for the parameters ε and σ are $\varepsilon/k_{\mathrm{B}} = 119.8$ K and $\sigma = 3.405$ Å respectively.

In the initialisation routine, the positions of a face centred cubic lattice are generated. For an $L \times L \times L$ system containing $4M^3$ particles, the fcc lattice constant a is $a = L/M$. It may be safer to put the particles not exactly on the boundary facets of the system because as a result of rounding errors it might not always be clear whether they belong to the system under consideration or a neighbouring copy.

The procedure in Appendix B3 for generating random numbers with a Gaussian distribution should be used in order to generate momenta according to a Maxwell distribution. First generate all the momenta with some arbitrary distribution width. Then calculate the total momentum $\mathbf{p}_{\text{tot}}$ and subtract a momentum $\bar{\mathbf{p}} = \mathbf{p}_{\text{tot}}/N$ from

each of the momenta in order to make the total momentum zero. Now the kinetic energy is calculated and then all momenta are rescaled to arrive at the desired kinetic energy.

When calculating the forces, the minimum image convention should be adopted. It is advisable to start without using a neighbour list. For the minimum image convention it should be checked for each pair (i,j) whether the difference of the x-components $x_i - x_j$ is larger or smaller than $L/2$ in absolute value. If it is larger, then an amount L should be added to or subtracted from this difference to transform it to a value which is smaller than $L/2$ (in absolute value). In many codes, this translation is implemented as follows:

$$x \to x - [x/L] * L, \tag{8.21}$$

where [ ] denotes the integer part. This procedure is then repeated for the y- and z-components. Potential and force may be adjusted according to Eqs (8.12) and (8.13).

The equations of motion are solved using the leap-frog or the velocity form of the Verlet algorithm. A good value for the time step is 10^{-14} s which in units of $(m\sigma^2/\varepsilon)^{1/2}$ is equal to about 0.004. Using the argon mass as the unit of mass, σ as the unit of distance and $\tau = (m\sigma^2/\varepsilon)^{1/2}$ as the unit of time, the x-component of the force acting on particle i resulting from the interaction with particle j is given by

$$F_x^{ij} = (x_i - x_j)(48r_{ij}^{-14} - 24r_{ij}^{-8}) \tag{8.22}$$

with similar expressions for the y- and z-components.

After each step in the Verlet/leap-frog algorithm, each particle should be checked to see whether it has left the volume. If this is the case, it should be translated over a distance $\pm L$ along one or more of the Cartesian axes in order to bring it back into the system in accordance with the periodic boundary conditions.

During equilibration, the velocities (momenta) should be rescaled at regular intervals. The user might specify the duration of this phase and the interval between momentum rescalings.

During the production phase, the following quantities should be stored in a file at each time step: the kinetic energy, potential energy, and the virial

$$\sum_{ij} r_{ij} F(r_{ij}). \tag{8.23}$$

Furthermore, the program should keep a histogram-array containing the numbers of pairs found with a separation between r and $r + \Delta$ for, say, $\Delta = L/200$ from which in the end the correlation function can be read off.

Table 8.1. *Molecular dynamics data for thermodynamic quantities of the Lennard–Jones liquid.*

$\rho(1/\sigma^3)$	$T_0(\varepsilon/k_B)$	T	$\beta P/\rho$	$U(\varepsilon)$
0.88	1.0	0.990 (2)	2.98 (2)	−5.704 (1)
0.80	1.0	1.010 (2)	1.31 (2)	−5.271 (1)
0.70	1.0	1.014 (2)	1.06 (4) (5)	−4.662 (1)

T_0 is the desired temperature; T is the temperature as determined from the simulation; ρ is the density: $\rho = N/V$. All values are in reduced units.

PROGRAMMING EXERCISE

Write a program that simulates the behaviour of a Lennard–Jones liquid with the proper argon parameters given above.

Check 1 To check the program, you can use small particle numbers, such as 32 or 108. Check whether the program is time-reversible by integrating for some time (without rescaling) and then reversing velocities. The system should then return to its initial configuration (graphical display of the system might be helpful).

Check 2 The definite check is to compare your results for argon with literature. A good value for the equilibration time is 10.0 τ and rescalings could take place after every 10 or 20 time steps. A sufficiently long simulation time to obtain accurate results is 20.0 τ (remember the time step is 0.004 τ). In Table 8.1 you can find a few values for the potential energy and pressure for different temperatures. Note that the average temperature in your simulation will not be precisely equal to the desired value. In Figure 7.1, the pair correlation function for $\rho = N/V = 1.06$ and $T = 0.827$ is shown.

It is interesting to study the specific heat (Eq. (7.37)) in the solid and in the gas phase. You may compare the behaviour with that of an ideal gas, $c_V = 3k_B/T$ per particle, and for a harmonic solid, $c_v = 3k_BT$ per particle (this is the Dulong–Petit law).

Note that phase transitions are difficult to locate, as there is a strong hysteresis in the physical quantities there. It is however interesting to obtain information about the different phases. For $T = 1$, $\rho = 0.8$ the argon Lennard–Jones system is found in the liquid phase, and for $\rho = 1.2$ and $T = 0.5$ in the solid phase. The gas phase is found for example with $\rho = 0.3$ and and $T = 3.0$. It is very instructive to plot the correlation function for the three phases and explain how they look. Another interesting exercise is to calculate the diffusion constant by plotting the displacement as a function of time averaged over all particles. For times smaller

than the typical collision time (time of free flight), you should find

$$\langle x^2 \rangle \propto t^2, \tag{8.24}$$

and this crosses over to diffusive behaviour

$$\langle x^2 \rangle = Dt, \tag{8.25}$$

with D the diffusion constant. In the solid phase, the diffusion constant is 0. In the gas phase, the diffusive behaviour sets in at later times than in the fluid.

If the program works properly, keeping a Verlet neighbour list as discussed in the previous section can be implemented. Verlet [8] used $r_{\text{cut-off}} = 2.5\sigma$ and $r_{\text{max}} = 3.3\sigma$. A more detailed analysis of the increase in efficiency for various values of r_{max} with $r_{\text{cut-off}} = 2.5\sigma$ shows that $r_{\text{max}} = 3.0\sigma$ with the neighbour list updated once every 25 integration steps is indeed most efficient [2, 15].

PROGRAMMING EXERCISE

Implement the neighbourlist in your program and check whether the results remain essentially the same. Determine the increase in efficiency.

8.4 Integration methods: symplectic integrators

There exist many algorithms for integrating ordinary differential equations, and a few of these are described in Appendix A. In this section, we consider the particular case of numerically integrating the equations of motion for a dynamical system described by a time-independent Hamiltonian, of which the classical many-particle system at constant energy is an example. Throughout this section we consider the equation of motion for a single particle in one dimension. The discussion is easily generalised to more particles in more dimensions.

The Verlet algorithm is the most popular algorithm for molecular dynamics and we shall consider it in more detail in the next subsection. Before doing so, we describe a few criteria which were formulated by Berendsen and van Gunsteren [16] for integration methods for molecular dynamics. First of all, *accuracy* is an important criterion: it tells us to which power of the time step the numerical trajectory will deviate from the exact one after one integration step (see also Appendix A). Note that the prefactor of this may diverge if the algorithm is unstable (e.g. close to a singularity of the trajectory). The accuracy is the criterion that is usually considered in numerical analysis in connection with integration methods.

Two further criteria are related to the behaviour of the energy and other conserved quantities of a mechanical system which are related to symmetries of the interactions. Along the exact trajectory, energy is conserved as a result of the time-translation invariance of the Hamiltonian, but the energy of the numerical trajectory will deviate from the initial value and this deviation can be characterised by its *drift*,

a steady increase or decrease, and the *noise*, fluctuations on top of the drift. Drift is obviously most undesirable. In microcanonical MD we want to sample the points in phase space with a given energy; these points form a hypersurface in phase space – the so-called *energy surface*. If the system drifts away steadily from this plane it is obviously not in equilibrium.

It is very important to distinguish in all these cases between two sources of error: those resulting from the numerical integration method as opposed to those resulting from finite precision arithmetic, inherent to computers. For example, we shall see below that the Verlet algorithm is not susceptible to energy drift in exact arithmetic. Drift will however occur in practice as a result of finite precision of computer arithmetic, and although different formulations of the Verlet algorithm have different susceptibility to this kind of drift, this depends also on the particular way in which numbers are rounded off in the computer.

Recently, there has been much interest in *symplectic integrators*. After considering the Verlet algorithm in some detail, we shall describe the concept of symplecticity[2] and its relevance to numerical integration methods.

8.4.1 The Verlet algorithm revisited

Properties of the Verlet algorithm

In this section we treat the Verlet algorithm

$$x(t+h) = 2x(t) - x(t-h) + h^2 F[x(t)] \tag{8.26}$$

in more detail with emphasis on issues which are relevant to MD. A derivation of this algorithm can be found in Appendix A7.1. The error per integration step is of the order h^4. Note that we take the mass of the particle(s) involved equal to 1. Unless stated otherwise, we analyse the one-dimensional single-particle version of the algorithm. The momenta are usually determined as

$$p(t) = [x(t+h) - x(t-h)]/(2h) + \mathcal{O}(h^2). \tag{8.27}$$

Note that there is no need for a more accurate formula, as the accumulated error in the positions after many steps is also of order h^2. We shall check this below, using also a more accurate expression for the momenta [16]:

$$p(t) = [x(t+h) - x(t-h)]/(2h) - \frac{h}{12}\{F[x(t+h)] - F[x(t-h)]\} + \mathcal{O}(h^3). \tag{8.28}$$

This form can be derived by subtracting the Taylor expansions for $x(t+h)$ and $x(t-h)$ about t, and approximating $\mathrm{d}F[x(t)]/\mathrm{d}t$ by $\{F[x(t+h)] - F[x(t-h)]\}/h$.

[2] Some authors use the term 'symplecticness' instead of 'symplecticity'.

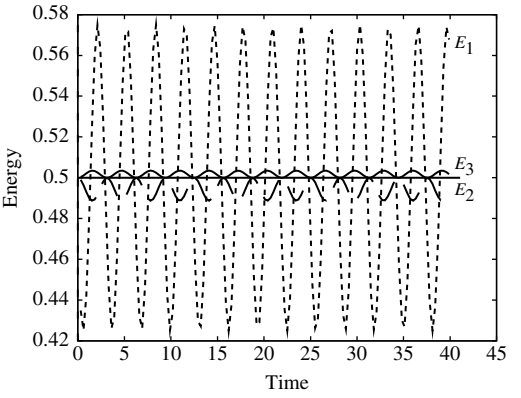

Figure 8.2. The energy of the harmonic oscillator determined using the various velocity estimators described in the text. E_1 is the energy using (8.29), E_2 uses (8.27) and E_3 was calculated using (8.28).

In the leap-frog version, we have the velocities at our disposal for times halfway between those at which the positions are given:

$$p(t + h/2) = [x(t + h) - x(t)]/h + \mathcal{O}(h^2). \tag{8.29}$$

Each of the expressions (8.27–8.29) for the momentum gives rise to a different expression for the energy.

We first analyse the different ways of calculating the total energy for the simple case of the one-dimensional harmonic oscillator

$$\mathcal{H} = (p^2 + x^2)/2 \tag{8.30}$$

and we can use any of the formulae (8.27–8.29) for the momentum. In Figure 8.2 the different energy estimators are shown as a function of time for the harmonic oscillator which is integrated using the Verlet algorithm with a time step $h = 0.3$. This is to be compared with the period $T = 2\pi$ of the motion $x(t) = \cos(t)$ (for appropriate initial conditions). It is seen that the leap-frog energy estimator is an order of magnitude worse than the other two. This is not surprising, since the fact that the velocity is not calculated at the same time instants as the position results in deviation of the energy from the continuum value of order h instead of h^2 when using (8.27). The energy estimator using third order momenta according to (8.28) is better than the second order form. Note that the error in the position accumulates in time to give $\mathcal{O}(h^2)$ (see Problem A3), so that there is no point in calculating the momenta with a higher order of accuracy, as this will not yield an order of magnitude improvement. The fact that the error for the third order estimator is about a factor of 3 better than that of the second order one for the harmonic oscillator does not therefore indicate a systematic trend. More importantly, the error in *both* estimators

(8.27) and (8.28) does indeed scale as h^2. In the following we determine momenta according to Eq. (8.27). In the leap-frog version the momentum estimator is

$$p(t) = [p(t + h/2) + p(t - h/2)]/2 + \mathcal{O}(h^2). \tag{8.31}$$

The results for the various energy estimators can be obtained by solving the harmonic oscillator in the Verlet algorithm analytically. The 'Verlet harmonic oscillator' reads

$$x(t + h) = 2x(t) - x(t - h) - h^2 x(t). \tag{8.32}$$

If we substitute $x(t) = \exp(\mathrm{i}\omega t)$ into the last equation, we obtain

$$\cos(\omega h) = 1 - h^2/2 \tag{8.33}$$

and this defines a frequency ω differing by an amount of order h^2 from the angular frequency $\omega = 1$ of the exact solution. The difference between the numerical and the exact solution will therefore show a slow beat.

A striking property of the energy determined from the Verlet/leap-frog solution is that it does not show any drift in the total energy (in exact arithmetic). This stability follows directly from the fact that the Verlet algorithm is time-reversible, which excludes steady increase or decrease of the energy for periodic motion. In a molecular dynamics simulation, however, the integration time, which is the duration of the simulation, is much smaller than the period of the system, which is the *Poincaré time*, that is the time after which the system returns to its starting configuration. The error in the energy might therefore grow steadily during the simulation. It turns out, however, that the deviation of the energy remains bounded in this case also, as the Verlet algorithm possesses an additional symmetry, called *symplecticity*. Symplecticity will be described in detail in Section 8.4.2. Here we briefly describe what the consequences of symplecticity are for an integration algorithm. Symplecticity gives rise to conserved quantities, and in particular, it can be shown that a discrete analogue of the total energy is rigorously conserved (in exact arithmetic) [17]. It turns out that this discrete energy deviates from the continuum energy at most an amount of order h^k, for some positive integer k. Therefore, the energy cannot drift away arbitrarily and it follows that the noise remains bounded.

To illustrate this point we return to the harmonic oscillator. In this particular case we can actually determine the conserved discrete energy. In the leap-frog formulation:

$$p(t + h/2) = p(t - h/2) - hx(t); \tag{8.34a}$$

$$x(t + h) = x(t) + hp(t + h/2), \tag{8.34b}$$

it is equal to [18]

$$\mathcal{H}_{\mathrm{D}} = \tfrac{1}{2}[p(t - h/2)^2 + x(t)^2 - hp(t - h/2)x(t)]. \tag{8.35}$$

The fact that this quantity is conserved can also be checked directly using (8.34b). This energy is equal to $1/2-h^2/8$ for the solution $\cos(\omega t)$ with ω given in Eq. (8.33). For general potentials, the discrete energy is not known.

As mentioned before, the absence of drift in the energy in the case of the harmonic oscillator can be explained by the time-reversibility of the Verlet algorithm, and comparisons with Runge–Kutta integrators for example, which are in general not time-reversible for potentials such as the harmonic oscillator, do not convincingly demonstrate the necessity for using a symplectic algorithm. Symplecticity does however impose a restriction on the noise, but time-reversibility does not.

Symplectic integrators are generally recommended for integrating dynamical systems because they generate solutions with the same geometric properties in phase space as the solutions of the continuum dynamical system. The fact that the deviation of the energy is always bounded is a pleasant property of symplectic integrators. Symplectic integrators are considered in more detail in Section 8.4.2.

Finite precision of computer arithmetic obviously does not respect the symplectic geometry in phase space. Hockney and Eastwood observed that when numbers are rounded off properly in the computer, the system tends to heat up because the rounding effects can be viewed as small random forces acting on the particles [19]. If real numbers are systematically truncated to finite precision numbers, the system cools down slowly. Both effects are clearly signs of nonsymplectic behaviour.

Several classes of symplectic integrators with explicit formulas for different orders of accuracy have been found. Runge–Kutta–Nystrom integrators (not to be confused with ordinary Runge–Kutta algorithms) have been studied by Okunbor and Skeel [20]. Yoshida [21] and Forest [22] have considered Lie-integrators. Their approach follows rather naturally from the structure of the symplectic group, as we shall see in Section 8.4.2.[3]

Let us make an inventory of relevant symmetry properties of integrators. First of all, time-reversibility is important. If it is present in the equations of motion, as is usually the case in MD, it is natural to require it in the integration method. Another symmetry is phase space conservation. This is a property of the trajectories of the continuum equations of motion – this property is given by Liouville's theorem – and it is useful to have our numerical trajectories obeying this condition too (note that time-reversibility by itself does not guarantee phase space conservation). The most detailed symmetry requirement is symplecticity, which will be considered in greater detail below (Section 8.4.2). This incorporates phase space conservation and conservation of a number of conserved quantities, the so-called *Poincaré invariants*. The symplectic symmetry properties can also be formulated in geometrical terms

[3] Gear algorithms [16, 23, 24] have been fashionable for MD simulations. These are predictor–corrector algorithms requiring only one force evaluation per time step. Gear algorithms are not symplectic and they are becoming less popular for that reason.

as we shall see below. Most important for molecular dynamics is the property that the total energy fluctuates within a narrow range around the exact one. Some comparison has been carried out between nonsymplectic phase space conserving and symplectic integrators [25], and this gave no indication of the superiority of symplectic integrators above merely phase-space conserving ones. As symplectic integrators are not more expensive to use than nonsymplectic time-reversible ones, their use is recommended as the safest option. Investigating the merits of the various classes of integration methods for microcanonical MD is a fruitful area for future research.

Frictional forces

Later we shall encounter extensions of the standard MD method where a frictional force is acting on the particles along the direction of the velocity. The Verlet algorithm can be generalised to include such frictional forces and we describe this extension for the one-dimensional case which can easily be generalised to more dimensions. The continuum equation of motion is

$$\ddot{x} = F(x) - \gamma \dot{x}, \tag{8.36}$$

and expanding $x(h)$ and $x(-h)$ around $t = 0$ in the usual way (see Appendix A7.1) gives

$$x(h) = x(0) + h\dot{x}(0) + h^2[-\gamma \dot{x}(0) + F(0)]/2 + h^3 \dddot{x}(0)/6 + \mathcal{O}(h^4) \tag{8.37a}$$

$$x(-h) = x(0) - h\dot{x}(0) + h^2[-\gamma \dot{x}(0) + F(0)]/2 - h^3 \dddot{x}(0)/6 + \mathcal{O}(h^4). \tag{8.37b}$$

Addition then leads to

$$x(h) = 2x(0) - x(-h) + h^2[-\gamma \dot{x}(0) + F(0)] + \mathcal{O}(h^4) \tag{8.38}$$

where $\dot{x}(0)$ remains to be evaluated. If we write

$$\dot{x}(0) = [x(h) - x(-h)]/(2h) + \mathcal{O}(h^2), \tag{8.39}$$

and substitute this into (8.38), we obtain

$$(1 + \gamma h/2)x(h) = 2x(0) - (1 - \gamma h/2)x(-h) + h^2 F(0) + \mathcal{O}(h^4). \tag{8.40}$$

A leap-frog version of the same algorithm is

$$x(h) = x(0) + hp(h/2); \tag{8.41a}$$

$$p(h/2) = \frac{(1 - \gamma h/2)p(-h/2) + hF(0)}{1 + \gamma h/2}. \tag{8.41b}$$

If the mass m is not equal to unity, the factors $1 \pm \gamma h/2$ are replaced by $1 \pm \gamma h/(2m)$.

It is often useful to simulate the system with a prescribed temperature rather than at constant energy. In Section 8.5 we shall discuss a constant-temperature MD

method in which a time-dependent friction parameter occurs, obeying a first order differential equation:

$$\ddot{x}(t) = -\gamma(t)\dot{x}(t) + F[x(t)] \tag{8.42a}$$

$$\dot{\gamma}(t) = g[\dot{x}(t)]. \tag{8.42b}$$

The solution can conveniently be presented in the leap-frog formulation. As the momentum is given at half-integer time steps in this formulation, we can solve for γ in the following way:

$$\gamma(h) = \gamma(0) + hg[p(h/2)] + \mathcal{O}(h^2), \tag{8.43}$$

and this is to be combined with Eqs. (8.41). Velocity-Verlet formulations (Eqs. (8.9)) for equations of motions including friction terms can be found straightforwardly. This is left as an exercise to the reader – see also Ref. [26].

*8.4.2 Symplectic geometry; symplectic integrators

In recent years, major improvement has been achieved in understanding the merits of the various methods for integrating equations of motion which can be derived from a Hamiltonian. This development started in the early 1980s with the observations made independently by Ruth [27] and Feng [28] that methods for solving Hamiltonian equations of motion should preserve the geometrical structure of the continuum solution in phase space. This geometry is the so-called *symplectic geometry*. Below we shall explain what this geometry is about, and what the properties of symplectic integrators are. In Section 8.4.3 we shall see how symplectic integrators can be constructed. We restrict ourselves again to a two-dimensional phase space (one particle moving in one dimension) spanned by the coordinates p and x, but it should be realised that the analysis is trivially generalised to arbitrary numbers of particles in higher dimensional space with phase space points $(\mathbf{p}_1, \ldots, \mathbf{p}_m, \mathbf{r}_1, \ldots, \mathbf{r}_m)$.[4] The equations of motion for the particle are derived from a Hamiltonian which for a particle moving in a potential (in the absence of constraints) reads

$$\mathcal{H}(p, x) = \frac{p^2}{2} + V(x). \tag{8.44}$$

The Hamilton equations of motion are then given as

$$\dot{p} = -\frac{\partial \mathcal{H}(p, x)}{\partial x} \tag{8.45a}$$

$$\dot{x} = \frac{\partial \mathcal{H}(p, x)}{\partial p} \tag{8.45b}$$

[4] Although we use the notation $\mathbf{r}_i$ for the coordinates, they may be generalised coordinates.

It is convenient to introduce the combined momentum–position coordinate $z = (p, x)$, in terms of which the equations of motion read

$$\dot{z} = J\nabla\mathcal{H}(z) \tag{8.46}$$

where J is the matrix

$$J = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \tag{8.47}$$

and $\nabla\mathcal{H}(z) = (\partial\mathcal{H}(z)/\partial p, \partial\mathcal{H}(z)/\partial x)$.[5]

Expanding the equation of motion (8.46) to first order, we obtain the time evolution of the point z to a new point in phase space:

$$z(t+h) = z(t) + hJ\nabla_z\mathcal{H}[z(t)]. \tag{8.49}$$

The exact solution of the equations of motion can formally be written as

$$z(t) = \exp(tJ\nabla_z\mathcal{H})[z(0)] \tag{8.50}$$

where the exponent is to be read as a series expansion of the operator $tJ\nabla_z\mathcal{H}$. This can be verified by substituting Eq. (8.50) into (8.46). This is a one-parameter family of mappings with the time t as the continuous parameter. The first order approximation to (8.50) coincides with (8.49).

Now consider a small region in phase space located at $z = (p, x)$ and spanned by the infinitesimal vectors δz^a and δz^b. The area δA of this region can be evaluated as the cross product of δz^a and δz^b which can be rewritten as[6]

$$\delta A = \delta z^a \times \delta z^b = \delta z^a \cdot (J\delta z^b). \tag{8.51}$$

It is now easy to see that the mapping (8.50) preserves the area δA. It is sufficient to show that its time derivative vanishes for $t = 0$, as for later times the analysis can be translated to this case. We have

$$\begin{aligned} \left.\frac{\mathrm{d}\delta A}{\mathrm{d}t}\right|_{t=0} &= \frac{\mathrm{d}}{\mathrm{d}t}\{[\mathrm{e}^{tJ\nabla_z\mathcal{H}}(\delta z^a)] \cdot [J\mathrm{e}^{tJ\nabla_z\mathcal{H}}(\delta z^b)]\}_{t=0} \\ &= [J\nabla_z\mathcal{H}(\delta z^a)] \cdot (J\delta z^b) + (\delta z^a) \cdot [JJ\nabla_z\mathcal{H}(\delta z^b)]. \end{aligned} \tag{8.52}$$

We can find $\mathcal{H}(\delta z^{a,b})$ using a first order Taylor expansion:

$$\mathcal{H}(\delta z^a) = \mathcal{H}(z + \delta z^a) - \mathcal{H}(z) = \delta z^a \cdot \nabla_z\mathcal{H}(z), \tag{8.53}$$

[5] In more than one dimension, the vector z is defined as $(p_1, \ldots, p_N, x_1, \ldots, x_N)$, and the matrix J reads in that case

$$J = \begin{pmatrix} 0 & -I \\ I & 0 \end{pmatrix} \tag{8.48}$$

where I is the $N \times N$ unit matrix.

[6] Note that the area can be negative: it is an *oriented* area. In the language of differential geometry this area is called a *two-form*.

and similar for $\mathcal{H}(\delta z^b)$. This leads to the form

$$\left.\frac{d\delta A}{dt}\right|_{t=0} = -(L^T \delta z^a) \cdot (J\delta z^b) - (\delta z^a) \cdot (JL^T \delta z^b) \tag{8.54}$$

where L is the Jacobian matrix of the operator $J\nabla_z \mathcal{H}$:

$$L_{ij} = \sum_k J_{ik}[\partial^2 \mathcal{H}(z)/\partial z_k \partial z_j] = \begin{pmatrix} -\mathcal{H}_{px} & -\mathcal{H}_{xx} \\ \mathcal{H}_{pp} & \mathcal{H}_{px} \end{pmatrix}. \tag{8.55}$$

Here $\mathcal{H}_{xx}$ denotes the second partial derivative with respect to x etcetera. It is easy to see that the matrix L satisfies

$$L^{\mathrm{T}} J + JL = 0, \tag{8.56}$$

where L^{T} is the transpose of L, and hence from (8.54) the area δA is indeed conserved.

We can now define symplecticity in mathematical terms. The Jacobi matrix S of the mapping $\exp(tJ\nabla H)$ is given as $S = \exp(tL)$. This matrix satisfies the relation:

$$S^{\mathrm{T}} J S = J. \tag{8.57}$$

Matrices satisfying this requirement are called *symplectic*. They form a Lie group whose Lie algebra is formed by the matrices L satisfying (8.56). General nonlinear operators are symplectic if their Jacobi matrix is symplectic.

In more than two dimensions the above analysis can be generalised for *any* pair of canonical variables p_i, x_i – we say that phase space area is conserved for any pair of one-dimensional conjugate variables p_i, x_i. The conservation law can be formulated in an integral form [29]; this is depicted in Figure 8.3. In this picture the three axes correspond to p, x and t. If we consider the time evolution of the points lying on a closed loop in the p, x plane, we obtain a tube which represents the flow in phase space. The area conservation theorem says that *any* loop around the tube encloses the same area $\oint p\mathrm{d}x$. In fact, there exists a similar conservation law for volumes enclosed by the areas of pairs of canonical variables: these volumes are called the *Poincaré invariants*. For the particular case of the volume enclosed by areas of *all* the pairs of canonical variables, we recover Liouville's theorem which says that the volume in phase space is conserved. Phase space volume conservation is equivalent to the Jacobi determinant of the time evolution operator in phase space being equal to 1 (or -1 if the orientation is not preserved). For two-dimensional matrices, the Jacobi determinant being equal to 1 is equivalent to symplecticity as can easily be checked from (8.57). This is also obvious from the geometric representation in Figure 8.3. For systems with a higher-dimensional phase space, however, the symplectic symmetry is a more detailed requirement than mere phase space conservation.

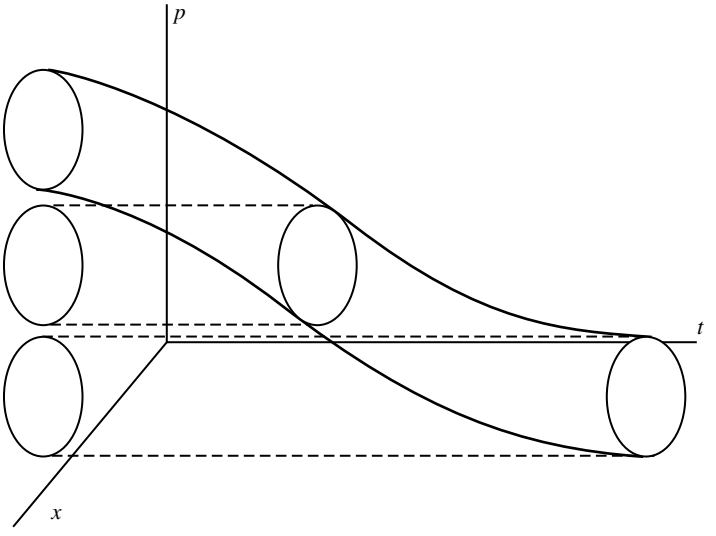

Figure 8.3. The area conservation law for a symplectic flow. The integral $\oint p\mathrm{d}x$ for any loop around the tube representing the flow of a closed loop in the p, x plane remains constant. This integral represents the area of the projection of the loop onto the xp plane. Note that the loops do not necessarily lie on a plane of constant time.

We have seen that symplecticity is a symmetry of Hamiltonian mechanics in continuum time; now we consider numerical integration methods for Hamiltonian systems (discrete time). As mentioned above, it is not clear whether full symplecticity is necessary for a reliable description of the dynamics of a system by a numerical integration. However, it will be clear that the preservation of the symmetries present in continuum time mechanics is the most reliable option. The fact mentioned above, that symplecticity implies conservation of the discrete version of the total energy, is an additional feature in favour of symplectic integrators for studying dynamical systems.

It should be noted that symplecticity does not guarantee time reversibility or vice versa. Time reversibility shows up as the Hamiltonian being invariant when replacing p by $-p$, and a Hamiltonian containing odd powers of p might still be symplectic.

*8.4.3 Derivation of symplectic integrators

The first symplectic integrators were found by requiring that an integrator of some particular form be symplectic. The complexity of the resulting algebraic equations for the parameters in the integration scheme was found to increase dramatically with

increasing order of the integrator. Later Yoshida [21] and Forest [22] developed a different scheme for finding symplectic integrators, and in this section we follow their analysis.

Consider a Hamiltonian of the simple form:

$$\mathcal{H} = T(p) + U(x) \tag{8.58}$$

(we still restrict ourselves to a particle in one dimension – results are easily generalised). In terms of the variable $z = (p, x)$ the equations of motion read

$$\begin{aligned}\frac{\mathrm{d}z}{\mathrm{d}t} &= \left(-\frac{\partial \mathcal{H}}{\partial x}, \frac{\partial \mathcal{H}}{\partial p}\right) = \left(-\frac{\partial U(x)}{\partial x}, \frac{\partial T(p)}{\partial p}\right) \\ &= J\nabla\mathcal{H}(z) \equiv \tilde{T}(z) + \tilde{U}(z),\end{aligned} \tag{8.59}$$

where in the last expression the operator $J\nabla\mathcal{H}$, which acts on $z = (p, x)$, is split into the contributions from the kinetic and potential energy respectively:

$$\tilde{T}(z) = \left(0, \frac{\partial T(p)}{\partial p}\right) \tag{8.60a}$$

$$\tilde{U}(z) = \left(-\frac{\partial U(x)}{\partial x}, 0\right). \tag{8.60b}$$

$\tilde{T}$ and $\tilde{U}$ are therefore also operators which map a point $z = (p, x)$ in phase space onto another point in phase space.

As we have seen in the previous section, the exact solution of (8.59) is given as

$$z(t) = \exp(tJ\nabla H)[z(0)] = \exp[t(\tilde{T} + \tilde{U})][z(0)]. \tag{8.61}$$

The term $\exp(tJ\nabla H)$ is a time evolution operator. It is a symplectic operator, as are $\exp(t\hat{T})$ and $\exp(t\hat{U})$ since these can both be derived from a Hamiltonian (for a free particle and a particle with infinite mass respectively).

An n*th order integrator* for time step h is now defined by a set of numbers a_k, b_k, $k = 1, \ldots, m$, such that

$$\prod_{k=1}^{m} \exp(a_k h\tilde{T}) \exp(b_k h\tilde{U}) = \exp(hJ\nabla H) + \mathcal{O}(h^{n+1}). \tag{8.62}$$

Since the operators $\exp(a_k h\tilde{T})$ and $\exp(b_k h\tilde{U})$ are symplectic, the integrator (8.62) is symplectic too. The difference between the integrator and the exact evolution operator can be expressed in Campbell–Baker–Hausdorff (CBH) commutators: if $\mathrm{e}^C = \mathrm{e}^A\mathrm{e}^B$ then

$$C = A + B + [A, B]/2 + ([A, [A, B]] + [B, [B, A]])/12 + \cdots \tag{8.63}$$

where the dots represent higher order commutators. This formula can be derived by writing $\exp(tA)\exp(tB) = \exp[t(A + B) + \Delta]$, expanding the operator Δ in

powers of t and equating equal powers of t on the left and right hand sides of the equality [30]. Applying this formula with $A = h\tilde{T}$ and $B = h\tilde{U}$ to increasing orders of commutators, we find

$$\exp(hJ\nabla H) = \exp(h\tilde{T})\exp(h\tilde{U}) + \mathcal{O}(h^2) \tag{8.64a}$$

$$\exp(hJ\nabla H) = \exp(h\tilde{T}/2)\exp(h\tilde{U})\exp(h\tilde{T}/2) + \mathcal{O}(h^3) \tag{8.64b}$$

etc.,

but the extra terms are often tedious to find. As $\tilde{T}$ and $\tilde{U}$ appear in the exponent, these expressions do not seem very useful. However, as it follows directly from Eq. (8.60) that applying $\tilde{T}$ and $\tilde{U}$ more than once gives zero, we have simply

$$\exp(ah\tilde{T}) = 1 + ah\tilde{T} \tag{8.65}$$

and similarly for $\exp(bh\tilde{U})$. Therefore, the first order integrator is

$$p(t+h) = p(t) - h\{\partial U[x(t)]/\partial x\}; \tag{8.66a}$$

$$x(t+h) = x(t) + h\{\partial T[p(t+h)]/\partial p\} \tag{8.66b}$$

which is recognised as the Verlet algorithm (although with a less accurate definition of the momentum).

The second order integrator is given by

$$p(t+h/2) = p(t) - h\{\partial \tilde{U}[x(t)]/\partial x\}/2; \tag{8.67a}$$

$$x(t+h) = x(t) + h\{\partial \tilde{T}[p(t+h/2)]/\partial p\}; \tag{8.67b}$$

$$p(t+h) = p(t+h/2) - h\{\partial \tilde{U}[x(t+h)]/\partial x\}/2. \tag{8.67c}$$

Applying this algorithm successively, the first and third step can be merged into one, and we obtain precisely the Verlet algorithm in leap-frog form with a third order error in the time step h. This error seems puzzling since we know that the Verlet algorithm gives positions with an error of order h^4 and momenta with an error of order h^2. The solution to this paradox lies in the interpretation of the variable p. If at time t, $p(t)$ is the continuous time derivative of the continuum solution $x(t)$, the above algorithm gives us $x(t+h)$ and $p(t+h)$ both with error h^3. If however $p(t)$ is defined as $[x(t+h) - x(t-h)]/(2h)$, the algorithm is equivalent to the velocity-Verlet algorithm and hence gives the positions $x(t+h)$ with an error of order h^4 and $p(t+h)$ is according to its definition given with a h^2 error. The way in which initial conditions are given defines which case we are in.

Finding higher order algorithms is nontrivial as we do not know the form of the higher order expansion terms of the operators $\exp(h\tilde{T})$ and $\exp(h\tilde{U})$. However, Yoshida [21] proposed writing the fourth order integrator in the following form:

$$S_2(\alpha h)S_2(\beta h)S_2(\alpha h) \tag{8.68}$$

where S_2 is the second order integrator, and he fixed α and β by the requirement that the resulting expression is equal to the continuum operator to fourth order. Higher order integrators were found similarly. The general result can be written as

$$\begin{aligned} &\text{for } k = 1 \text{ to } n \text{ do} \\ &\quad x^{(k)} = x^{(k-1)} - h a_k \partial T[p^{(k-1)}]/\partial p \\ &\quad p^{(k)} = p^{(k-1)} - h b_k \partial U[x^{(k)}]/\partial x \\ &\text{end} \end{aligned} \tag{8.69}$$

and the numbers a_k and b_k can be found in Yoshida's paper. For the fourth order case, they read

$$a_1 = a_4 = 1/[2(2 - 2^{1/3})]; \quad a_2 = a_3 = (1 - 2^{1/3})a_1 \tag{8.70a}$$

$$b_1 = b_3 = 2a_2; \quad b_2 = -2^{1/3} b_1; \quad b_4 = 0. \tag{8.70b}$$

From Yoshida's derivation it follows that there exists a conserved quantity which acts as the analog of the energy. The integrator is certainly not the same as the exact time evolution operator, but it deviates from the latter only by a small amount. Writing the integrator $S(h)$ as

$$S(h) = \exp(h A_{\mathrm{D}}) \tag{8.71}$$

we have a new operator A_{D} which deviates from the continuum operator A only by an amount of order h^{n+1}, as the difference can be written as a sum of higher order CBH commutators. It will be shown in Problem 8.9 that for an operator of the form $\exp(tA_{\mathrm{D}})$ which is symplectic for all t, there exists a Hamiltonian $\mathcal{H}_{\mathrm{D}}$ which is the analogue of the Hamiltonian in the continuum time evolution. This means that, if we know $\mathcal{H}_{\mathrm{D}}$ (which is usually impossible to find, except for the trivial case of the harmonic oscillator), we could either use the integrator (8.71) to give us the image at time h, or the continuum solution of the dynamical system with Hamiltonian $\mathcal{H}_{\mathrm{D}}$ for $t = h$: both mappings would give identical results. The Hamiltonian $\mathcal{H}_{\mathrm{D}}(z)$ is therefore a conserved quantity of the integrator, and it differs from the energy by an amount of order h^{n+1}. The existence of such a conserved quantity is also discussed in Refs. [17, 18, 31].

8.5 Molecular dynamics methods for different ensembles

8.5.1 Constant temperature

In experimental situations the total energy is often not a control variable as usually the temperature of the system is kept constant. We know that in the infinite system the temperature is proportional to the average kinetic energy per degree of freedom

with proportionality constant $k_B/2$, and therefore this quantity is used in MD to calculate the temperature, even though the system is finite (see Section 10.7 for a discussion on temperature for a finite system). As the total energy remains constant in the straightforward implementation of the molecular dynamics paradigm as presented in the previous sections, the question arises how we can perform MD simulations at constant temperature or pressure. We start with a brief overview of the various techniques which have been developed for keeping the temperature constant. Then we shall discuss the most successful one, the Nosé–Hoover method, in greater detail.

Overview of constant temperature methods

Experience from real life is a useful guide to understanding procedures for keeping the temperature at a constant value. In real systems, the temperature is usually kept constant by letting the system under consideration exchange heat with a much larger system in equilibrium – the heat bath. The latter has a definite temperature (it is in equilibrium) and the smaller system that we consider will assume the same temperature, as it has a negligible influence on the heat bath. Microscopically the heat exchange takes place through collisions of the particles in the system with the particles of the wall that separates the system from the heat bath. If, for example, the temperature of the heat bath is much higher than that of the system under consideration, the system particles will on average increase their kinetic energy considerably in each such collision. Through collisions with their partners in the system, the extra kinetic energy spreads through the system, and this process continues until the system has attained the temperature of the heat bath.

In a simulation we must therefore allow for heat flow from and to the system in order to keep it at the desired temperature. Ideally, such a heat exchange leads to a distribution ρ of configurations according to the canonical ensemble, irrespective of the number of particles:

$$\rho(R, P) = \mathrm{e}^{-\mathcal{H}(R,P)/(k_B T)}, \tag{8.72}$$

but some of the methods described below yield distributions differing from this by a correction of order $1/N^k$, $k > 0$. In comparison with the experimental situation, we are not confined to allowing heat exchange only with particles at the boundary: any particle in the system can be coupled to the heat bath.

Several canonical MD methods have been developed in the past. In 1980 Andersen [32] devised a method in which the temperature is kept constant by replacing every so often the velocity of a randomly chosen particle by a velocity drawn from a Maxwell distribution with the desired temperature. This method is closest to the experimental situation: the velocity alterations mimic particle collisions with the walls. The rate at which particles should undergo these changes

in velocity influences the equilibration time and the kinetic energy fluctuations. If the rate is high, equilibration will proceed quickly, but as the velocity updates are uncorrelated, they will destroy the long time tail of the velocity autocorrelation function. Moreover, the system will then essentially perform a random walk through phase space, which means that it moves relatively slowly. If on the other hand the rate is low, the equilibration will be very slow. The rate $R_{\text{collisions}}$ for which wall collisions are best mimicked by Andersen's procedure is of the order of

$$R_{\text{collisions}} \sim \frac{\kappa}{k_{\text{B}} n^{1/3} N^{2/3}} \tag{8.73}$$

where κ is the thermal conductivity of the system, and n, N the particle density and number respectively [32] (see Problem 8.9). Andersen's method leads to a canonical distribution for all N. The proof of this statement needs some theory concerning Markov chains and is therefore postponed to Section 15.4.3, where we consider the application of this method to lattice field theories.

For evaluating equilibrium expectation values for time- and momentum-independent quantities, the full canonical distribution (8.72) is not required: a canonical distribution in the positional coordinates

$$\rho(R) = \mathrm{e}^{-U(R)/(k_{\text{B}}T)} \tag{8.74}$$

is sufficient since the momentum part can be integrated out for momentum-independent expectation values. For a sufficiently large system the total kinetic energy of a canonical system will evolve towards its equilibrium value $3Nk_{\text{B}}T/2$ and fluctuations around this value are very small. We might therefore force the kinetic energy to have a value exactly equal to the one corresponding to the desired temperature. This means that we replace the narrow distribution of the kinetic energy by a delta-function

$$\rho(E_{\text{kin}}) \to \delta[E_{\text{kin}} - 3(N-1)k_{\text{B}}T/2]. \tag{8.75}$$

The simplest way of achieving this is by applying a simple velocity rescaling procedure as described in the previous section (Eqs. (8.14) and (8.15)) after *every* integration step rather than occasionally:

$$p_i \to p_i \sqrt{\frac{3/2(N-1)k_{\text{B}}T}{E_{\text{kin}}}}. \tag{8.76}$$

This method can also be derived by imposing a constant kinetic energy via a Lagrange multiplier term added to the Lagrangian of the isolated system [33]. It turns out [34] that this velocity rescaling procedure induces deviations from the canonical distribution of order $1/\sqrt{N}$, where N is the number of particles.

Apart from the rescaling method, which is rather *ad hoc*, there have been attempts to introduce the coupling via an extra force acting on the particles with the purpose

of keeping the temperature constant. This force assumes the form of a friction proportional to the velocity of the particles, as this is the most direct way to affect velocities and hence the kinetic energy:

$$m\ddot{\mathbf{r}}_i = \mathbf{F}_i(R) - \zeta(R,\dot{R})\dot{\mathbf{r}}_i. \tag{8.77}$$

The parameter ζ acts as a friction parameter which is the same for all particles and which will be negative if heat is to be added and positive if heat must be drained from the system. Various forms for ζ have been used, and as a first example we consider [33, 35]

$$\zeta(R,\dot{R}) = \frac{\mathrm{d}V(R)/\mathrm{d}t}{\sum_i \dot{\mathbf{r}}_i^2}. \tag{8.78}$$

This force keeps the kinetic energy $K = m\sum_i v_i^2/2$ constant as can be seen using (8.77). From this equation, we obtain

$$\frac{\partial K}{\partial t} \propto \sum_i \mathbf{v}_i\dot{\mathbf{v}}_i = -\sum_i \mathbf{v}_i[\nabla_i V(R) - \zeta(R,\dot{R})\mathbf{v}_i] = \frac{\mathrm{d}V}{\mathrm{d}t} - \sum_i \dot{\mathbf{r}}_i^2\zeta(R,\dot{R}) = 0. \tag{8.79}$$

It can be shown [34] that for finite systems the resulting distribution is purely canonical (without $1/N^k$ corrections) in the restricted sense, i.e. in the coordinate part only.

Another form of the friction parameter ζ was proposed by Berendsen *et al.* [36] This now has the form $\zeta = \gamma(1 - T_\mathrm{D}/T)$ with constant γ, T is the actual temperature $T = \sum_i mv_i^2/(3k_\mathrm{B})$, and T_D is the desired temperature. It can be shown that the temperature decays to the desired temperature exponentially with time at rate given by the coefficient γ. However, this method is not time reversible; moreover, it can be shown that the Nosé method (see below) is the only method with a single friction parameter which gives a full canonical distribution [37], so Berendsen's method cannot have this property. Berendsen's method can be related to a Langevin description of thermal coupling, in the sense that the time evolution of the temperature for a Langevin system (see Section 8.8) can be shown to be equivalent to that of a system with a coupling via ζ as given by Berendsen.

Nosé's method in the formulation by Hoover [37] uses yet another friction parameter ζ which is now determined by a differential equation:

$$\frac{\mathrm{d}\zeta}{\mathrm{d}t} = \Big(\sum_i v_i^2 - 3Nk_\mathrm{B}T_\mathrm{D}\Big)/Q \tag{8.80}$$

where Q is a parameter which has to be chosen with some care (see below) [38]. This way of keeping the temperature constant yields the canonical distribution for positions and momenta, as will be shown in the next subsection.

The Nosé and the Andersen methods yield precise canonical distributions for position and momentum coordinates. They still have important disadvantages, however.

In the Andersen method, it is not always clear at which rate the velocities are to be altered and it has been found [39, 40] that the temperature sometimes levels down at the wrong value. The Nosé–Hoover thermostat suffers from similar problems. In this method, the coupling constant Q in Eq. (8.80) between the heat bath and the system must be chosen – this coupling constant is the analogue of the velocity alteration rate in the Andersen method. It turns out [38] that for a Lennard–Jones fluid at high temperatures, the canonical distribution comes out well, but if the temperature is lowered [26], the temperature starts oscillating with an amplitude much larger than the standard deviation expected in the canonical ensemble. It can also occur that such oscillations are much smaller than the expected standard deviation, but in this case the fluctuations on top of this oscillatory behaviour are much smaller than in the canonical ensemble. Martyna *et al.* [41] have devised a variant of the Nosé–Hoover thermostat which is believed to alleviate these problems to some extent. Although the difficulties with these constant temperature approaches are very serious, they have received rather little attention to date. It should be clear that it must always be checked explicitly whether the temperature shows unusual behaviour; in particular, it should not exhibit systematic oscillations, and the standard deviation for N particles in D dimensions should satisfy

$$\overline{\Delta T} = \sqrt{\frac{2}{ND}}\overline{T} \tag{8.81}$$

where $\overline{\Delta T}$ is the width of the temperature distribution and $\overline{T}$ is the mean value [26]. This equation follows directly from the Boltzmann distribution.

*Derivation of the Nosé–Hoover thermostat

In this section we shall discuss Nosé's approach [34, 42], in which the heat bath is explicitly introduced into the system in the form of a single degree of freedom s. The Hamiltonian of the total (extended) system is given as

$$\mathcal{H}(P, R, p_s, s) = \sum_i \frac{\mathbf{p}_i^2}{2ms^2} + \frac{1}{2}\sum_{ij, i\neq j} U(\mathbf{r}_i - \mathbf{r}_j) + \frac{p_s^2}{2Q} + gkT\ln(s). \tag{8.82}$$

g is the number of independent momentum-degrees of freedom of the system (see below), and R and P represent all the coordinates $\mathbf{r}_i$ and $\mathbf{p}_i$ as usual. The physical quantities R, P and t (time) are virtual variables – they are related to real variables R', P' and t' via $R' = R$, $P' = P/s$ and $t' = \int^t \mathrm{d}\tau/s$. With these definitions we have for the real variables $P' = \mathrm{d}Q'/\mathrm{d}t'$.

First we derive the equations of motion in the usual way:

$$\frac{\mathrm{d}\mathbf{r}_i}{\mathrm{d}t} = \frac{\partial \mathcal{H}}{\partial \mathbf{p}_i} = \frac{\mathbf{p}_i}{ms^2} \tag{8.83a}$$

$$\frac{\mathrm{d}s}{\mathrm{d}t} = \frac{\partial \mathcal{H}}{\partial p_s} = \frac{p_s}{Q} \tag{8.83b}$$

$$\frac{\mathrm{d}\mathbf{p}_i}{\mathrm{d}t} = -\frac{\partial \mathcal{H}}{\partial \mathbf{r}_i} = -\nabla_i U(R) = -\sum_{i<j} \nabla_i U(\mathbf{r}_i - \mathbf{r}_j) \tag{8.83c}$$

$$\frac{\mathrm{d}p_s}{\mathrm{d}t} = -\frac{\partial \mathcal{H}}{\partial s} = \left(\frac{\sum_i p_i^2}{ms^2} - g k_\mathrm{B} T \right) \Big/ s. \tag{8.83d}$$

We have used the notation $\partial \mathcal{H}/\partial \mathbf{p}_i = \nabla_{\mathbf{p}_i} \mathcal{H}$, etc. The partition function of the total system (i.e. including heat bath degree of freedom s) is given by the expression:

$$Z = \frac{1}{N!} \int \mathrm{d}p_s \int \mathrm{d}s \int \mathrm{d}P \int \mathrm{d}R \times \delta \left(\sum_i \frac{p_i^2}{2ms^2} + \frac{1}{2} \sum_{ij, i \neq j} U(r_{ij}) + \frac{p_s^2}{2Q} + g k_\mathrm{B} T \ln(s) - E \right). \tag{8.84}$$

Integrations $\int \mathrm{d}R$ and $\int \mathrm{d}P$ are over all position and momentum degrees of freedom. We now rescale the momenta $\mathbf{p}_i$:

$$\frac{\mathbf{p}_i}{s} = \mathbf{p}_i', \tag{8.85}$$

so that we can rewrite the partition function as

$$Z = \frac{1}{N!} \int \mathrm{d}p_s \int \mathrm{d}s \int \mathrm{d}P' \int \mathrm{d}R \times s^{3N} \delta \left(\sum_i \frac{p_i'^2}{2m} + \frac{1}{2} \sum_{ij, i \neq j} U(r_{ij}) + \frac{p_s^2}{2Q} + g k_\mathrm{B} T \ln(s) - E \right). \tag{8.86}$$

We define the Hamiltonian $\mathcal{H}_0$ in terms of R and P' as

$$\mathcal{H}_0 = \sum_i \frac{p_i'^2}{2m} + \frac{1}{2} \sum_{ij, i \neq j} U(r_{ij}). \tag{8.87}$$

Furthermore we use the relation $\delta[f(s)] = \delta(s - s_0)/f'(s)$ with $f(s_0) = 0$ and set $g = 3N + 1$, so that we can rewrite Eq. (8.86) as

$$\begin{aligned} Z &= \frac{1}{N!}\int \mathrm{d}p_s \int \mathrm{d}s \int \mathrm{d}P' \int \mathrm{d}R \frac{s^{3N+1}}{(3N+1)k_\mathrm{B}T} \\ &\quad \times \delta\left(s - \exp\left[-\frac{\mathcal{H}_0(P',R) + p_s^2/2Q - E}{(3N+1)k_\mathrm{B}T}\right]\right) \\ &= \frac{1}{(3N+1)k_\mathrm{B}T}\frac{1}{N!}\int \mathrm{d}p_s \int \mathrm{d}P' \int \mathrm{d}R \exp\left[-\frac{\mathcal{H}_0(P',R) + p_s^2/2Q - E}{k_\mathrm{B}T}\right]. \end{aligned} \tag{8.88}$$

The dependence on p_s is simply Gaussian and integrating over this coordinate we obtain

$$Z = \frac{1}{3N+1}\left(\frac{2\pi Q}{k_\mathrm{B}T}\right)^{1/2} \exp(E/k_\mathrm{B}T) Z_\mathrm{c} \tag{8.89}$$

where Z_c is the canonical partition function:

$$Z_\mathrm{c} = \frac{1}{N!}\int \mathrm{d}P' \int \mathrm{d}R \exp[-\mathcal{H}_0(P',R)/k_\mathrm{B}T]. \tag{8.90}$$

It follows that the expectation value of a quantity A which depends on R and P is given by

$$\langle A(P/s,R)\rangle = \langle A(P',R)\rangle_\mathrm{c} \tag{8.91}$$

where $\langle\cdots\rangle_\mathrm{c}$ denotes an average in the canonical ensemble. The ergodic hypothesis relates this ensemble average to a virtual-time average.

The Lagrangian equations of motion for the $\mathbf{r}_i$ can be obtained by eliminating the momenta from (8.83a):

$$\frac{\mathrm{d}^2\mathbf{r}_i}{\mathrm{d}t^2} = -\frac{1}{ms^2}\nabla_i V(R) - \frac{2}{s}\frac{\mathrm{d}\mathbf{r}_i}{\mathrm{d}t}\frac{\mathrm{d}s}{\mathrm{d}t}. \tag{8.92}$$

In this equation the ordinary force term is recognised with a factor $1/s^2$ in front and with an additional friction term describing the coupling to the heat bath. The factor $1/s^2$ is consistent with the relation between real and virtual-time $\mathrm{d}t' = \mathrm{d}t/s$ given above. Together with the definitions $P' = P/s$ and $p_s' = p_s/s$, this leads to

the equations of motion in real variables:

$$\frac{\mathrm{d}\mathbf{r}_i'}{\mathrm{d}t'} = \frac{\mathbf{p}_i'}{m} \tag{8.93a}$$

$$\frac{\mathrm{d}\mathbf{p}_i'}{\mathrm{d}t'} = -\nabla_i V(R) - sp_s'\mathbf{p}_i'/Q \tag{8.93b}$$

$$\frac{\mathrm{d}s}{\mathrm{d}t'} = s'^2 p_s'/Q \tag{8.93c}$$

$$\frac{\mathrm{d}p_s'}{\mathrm{d}t'} = \Big(\sum_i p_i'^2/m - gk_\mathrm{B}T\Big)/s - s^2 p_s'^2/Q. \tag{8.93d}$$

Although these equations are equivalent to the equations for the virtual variables, there is a slight complication in the evaluation of averages. The point is that we have used the ergodic theorem for the canonical Hamiltonian expressed in virtual variables (P, R, t, s, p_s) in order to relate *virtual-time* averages to ensemble averages. The real time steps however are not equidistant and time averaging in real time is therefore not equivalent to averaging in virtual-time. Fortunately the two can be related. Expressing the real time t' as an integral over virtual-time τ according to $t' = \int_0^t \mathrm{d}\tau/s$ we obtain

$$\begin{aligned}\lim_{t'\to\infty}\frac{1}{t'}\int_0^{t'} A(P/s,R)\mathrm{d}\tau' &= \lim_{t'\to\infty}\frac{t}{t'}\frac{1}{t}\int_0^t A(P/s,R)\mathrm{d}\tau/s \\ &= \left[\lim_{t'\to\infty}\frac{1}{t}\int_0^t A(P/s,R)\mathrm{d}\tau/s\right] \Big/ \left(\lim_{t'\to\infty}\frac{1}{t}\int_0^t \mathrm{d}\tau/s\right) \\ &= \langle A(P/s,R)/s\rangle/\langle 1/s\rangle. \end{aligned} \tag{8.94}$$

It can be verified (see Problem 8.9) that the latter expression coincides with the canonical ensemble average if we put g equal to $3N$ instead of $3N+1$. This means that if we carry out the simulation using Eqs. (8.93) with $g = 3N$, real-time averages are equivalent to canonical averages.

Hoover [37] showed that by defining $\zeta = sp_s'/Q$, Eqs. (8.93) can be reduced to the simpler form

$$\frac{\mathrm{d}\mathbf{r}_i'}{\mathrm{d}t'} = \frac{\mathbf{p}_i'}{m}; \quad \frac{\mathrm{d}\mathbf{p}_i'}{\mathrm{d}t'} = \mathbf{F}_i - \zeta\mathbf{p}_i'; \tag{8.95}$$

$$\frac{\mathrm{d}\zeta}{\mathrm{d}t'} = \left(\frac{\sum_i p_i'^2}{m} - gk_\mathrm{B}T\right)/Q, \tag{8.96}$$

and taking g equal to the number of degrees of freedom, i.e. $3N$, he was able to show that the distribution $f(P, R, \zeta)$ is phase space conserving, i.e. it satisfies Liouville's equation.

The disadvantage of the real-time equations is that they are not Hamiltonian, in the sense that they cannot be derived from a Hamiltonian. Although this might not seem to be a problem, we prefer Hamiltonian equations of motion as they allow for stable (symplectic) integration methods as discussed in Section 8.4. Winkler *et al.* [43] have formulated canonical equations of motion in real-time but these are subject to severe numerical problems when integrating the equations of motion for large systems.

8.5.2 Keeping the pressure constant

In experimental situations not only the temperature is kept under control but also the pressure. The partition function for the (NpT)-ensemble is given as

$$Q(N,p,T) = \int \mathrm{d}V\, \mathrm{e}^{-\beta pV} \frac{1}{N!} \int \mathrm{d}R\, \mathrm{d}P\, \mathrm{e}^{-\beta \mathcal{H}(R,P)} = \int \mathrm{d}V\, \mathrm{e}^{-\beta pV} Z_{\mathrm{c}}(N,V,T) \tag{8.97}$$

(see Chapter 7). We use a lower-case p for the pressure in order to avoid confusion with the total momentum coordinate P. We now describe the scheme which is commonly adopted for keeping the pressure constant but do not go into too much detail as the analysis follows the same lines as the Nosé–Hoover thermostat, and refer to the literature for details [32, 34, 37].

Andersen first presented this scheme. He proposed incorporating the volume into the equations of motion as a dynamical variable and scaled the spatial coordinates back to a unit volume:

$$\mathbf{r}_i' = \mathbf{r}_i V^{1/3}, \tag{8.98}$$

where again the prime denotes the real coordinate – unprimed coordinates are those of the virtual system. Moreover

$$\mathbf{p}_i' = \mathbf{p}_i/(sV^{1/3}). \tag{8.99}$$

The canonical Hamiltonian is extended by *two* variables, the volume V and the canonical momentum p_V which can be thought of as the momentum of a piston closing the system.[7] The Hamiltonian has an extra 'potential energy' term pV and a 'kinetic' term $p_V^2/2W$ (W is the 'mass' of the piston, and p_V its momentum):

$$\begin{aligned}\mathcal{H}(P,R,p_s,s,p_V,V) = &\sum_i \frac{\mathbf{p}_i^2}{2mV^{2/3}s^2} + \frac{1}{2}\sum_{ij,i\neq j} U[V^{1/3}R] \\ &+ \frac{p_s^2}{2Q} + gkT\ln(s) + pV + p_V^2/2W.\end{aligned} \tag{8.100}$$

[7] Note that the system expands and contracts isotropically, so instead of a piston, the whole system boundary moves.

The equations of motion now read:

$$\frac{\mathrm{d}\mathbf{r}}{\mathrm{d}t} = \frac{\partial \mathcal{H}}{\partial \mathbf{p}_i} = \frac{\mathbf{p}_i}{mV^{2/3}s^2} \tag{8.101a}$$

$$\frac{\mathrm{d}s}{\mathrm{d}t} = \frac{\partial \mathcal{H}}{\partial p_s} = \frac{p_s}{Q} \tag{8.101b}$$

$$\frac{\mathrm{d}\mathbf{p}_i}{\mathrm{d}t} = -\frac{\partial \mathcal{H}}{\partial \mathbf{r}_i} = -\nabla_i U(V^{1/3}R) \tag{8.101c}$$

$$\frac{\mathrm{d}p_s}{\mathrm{d}t} = -\frac{\partial \mathcal{H}}{\partial s} = \left(\frac{\sum_i p_i^2}{mV^{2/3}s^2} - gk_\mathrm{B}T \right) /s \tag{8.101d}$$

$$\frac{\mathrm{d}V}{\mathrm{d}t} = \frac{\partial \mathcal{H}}{\partial p_V} = \frac{p_V}{W} \tag{8.101e}$$

$$\frac{\mathrm{d}p_V}{\mathrm{d}t} = -\frac{\partial \mathcal{H}}{\partial V} = \left(\frac{\sum_i p_i^2}{mV^{2/3}s^2} - \sum_i \nabla_i U(V^{1/3}R) \cdot \mathbf{r}_i \right) /(3V) - p. \tag{8.101f}$$

It can be shown in the same way as in the thermostat case that the distribution of configurations corresponds to that of the (NpT) ensemble:

$$\rho(P', R', V) = V^N \exp\{-[\mathcal{H}_0(P', R') + pV]/k_\mathrm{B}T\}. \tag{8.102}$$

Hoover [37] proposed similar equations of motion which conserve phase space, but they differ from this distribution by an extra factor V in front of the exponent [44].

The method described is restricted to isotropic volume changes and can therefore not be used for studying structural phase transitions in solids. A method which allows for anisotropic volume changes while keeping the pressure constant was developed by Parrinello and Rahman [45].

8.6 Molecular systems

8.6.1 Molecular degrees of freedom

Interactions in molecular systems can be divided into intramolecular and intermolecular ones. The latter are often taken to be atom-pair interactions similar to those considered in the previous sections. The intramolecular interactions (i.e. the interactions between the atoms of one molecule) are determined by chemical bonds, so not only are they strong compared with the intermolecular interactions (between atoms of different molecules) but they also include orientational dependencies. We now briefly describe the intramolecular degrees of freedom and interactions (see also Figure 8.4).

First of all, the chemical bonds can stretch. The interaction associated with this degree of freedom is usually described in the form of a harmonic potential for the

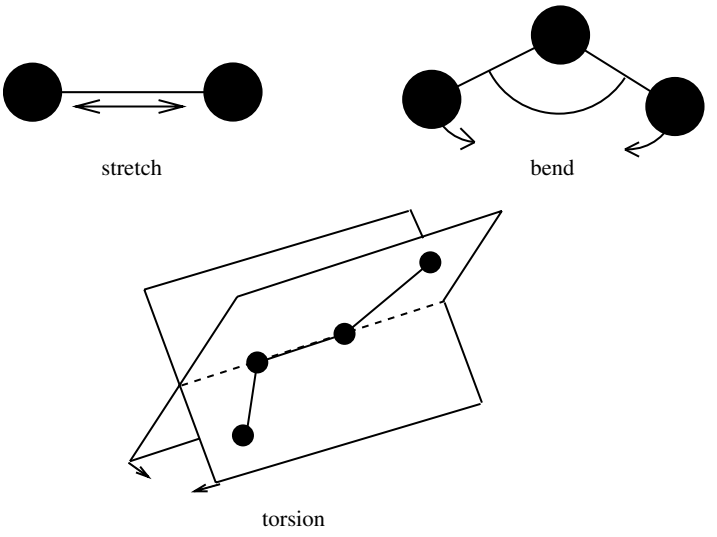

Figure 8.4. Different types of motion of atoms within a molecule.

bond length l:

$$V_{\text{stretch}}(l) = \tfrac{1}{2}\alpha_{\text{S}}(l - l_0)^2 \tag{8.103}$$

where l_0 is the equilibrium bond length.

Now consider three atoms bonded in a chain-like configuration A–B–C. This chain is characterised by a bending or *valence angle* φ which varies around an equilibrium value φ_0 and the potential is described in terms of a cosine:

$$V_{\text{valence}}(\varphi) = -\alpha_{\text{B}}[\cos(\varphi - \varphi_0) + \cos(\varphi + \varphi_0)] \tag{8.104}$$

where the equivalence of the angles φ_0 and $-\varphi_0$ is taken into account. Often, a harmonic approximation $\cos(\varphi) \approx 1 - \varphi^2/2$, valid for small angles, is used.

Finally there is an interaction associated with chain configurations of four atoms A–B–C–D. The plane through the first three atoms A, B, C does not coincide in general with that through B, C and D. The *torsion* interaction is similar to the bend interaction, but the angle (called *dihedral* angle), denoted by ϑ, is now that between these two planes:

$$V_{\text{torsion}}(\vartheta) = -\alpha_{\text{T}}[\cos(\vartheta - \vartheta_0) + \cos(\vartheta + \vartheta_0)]. \tag{8.105}$$

This interaction is also often replaced by its harmonic approximation. Other interactions and more complicated forms of these potentials can be used – we have only listed the simplest ones.

Characteristic vibrations associated with the different degrees of freedom distinguished here can be derived from the harmonic interactions – the frequencies

vary as the square root of the α-coefficients. In general, the bond length vibrations are the most rapid, followed by the bending vibrations. For an MD integration to be accurate, the time step should be chosen smaller than the fastest degree of freedom. But as this degree of freedom will vibrate with a small amplitude, because of the strong potential, we are using most of the computer time for those parts of the motion that are not expected to contribute strongly to the physical properties of the system. Moreover, if there is a clear separation between the time scales of the various degrees of freedom of the system, energy transfer between the fast and slow modes is extremely slow, so that it is difficult, if not impossible, to reach equilibrium within a reasonable amount of time. In such a case it is advisable to 'freeze' the fast modes by keeping them rigorously fixed in time. In practice this means that lengths of chemical bonds can safely be kept fixed, and perhaps some bending angles. In a more approximate description it is also possible to consider entire molecules as being rigid. In the next subsections we shall describe how to deal with rigid and partly rigid molecules.

8.6.2 Rigid molecules

We consider molecules which can be treated as rigid bodies whose motion consists of translations of the centre of mass and rotations around this point. The forces acting between two rigid molecules are usually composed of atomic pair interactions between atoms belonging to the two different molecules.[8] The total force acting on a molecule determines the translational motion and the torque determines the rotational motion. In the next subsection, we shall describe a direct formulation of the equations of motion of a simple rigid molecule – the nitrogen molecule. In the following subsection we shall then describe a different approach in which rigidity is enforced through constraints added to the Lagrangian.

Direct approach for the rigid nitrogen molecule

As a simple example consider the nitrogen molecule, N_2. This consists of two nitrogen atoms, each of mass $m \approx 14$ atomic mass units (a.m.u.) and whose separation d is kept fixed in the rigid approximation. The coordinates of the molecule are the three coordinates of the centre of mass and the two coordinates defining its orientation. The latter can be polar angles but here we shall characterise the orientation of the molecule by a unit direction vector $\hat{\mathbf{n}}$, pointing from atom 1 to atom 2 (see Figure 8.5).

The motion of the centre of mass of the molecule is determined by the total force $\mathbf{F}_{\text{tot}}$ acting on a particular molecule. This force is the sum of all the forces between

[8] Sometimes, off-centre interactions (i.e. not centred on the atomic positions) are taken into account too but we shall not consider these.

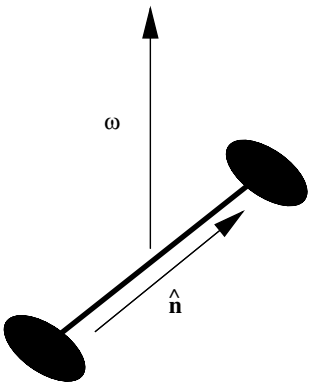

Figure 8.5. The nitrogen molecule. $\hat{\mathbf{n}}$ is a unit vector, $\boldsymbol{\omega}$ is the rotation vector.

each of the two atoms in the molecule and the atoms of the remaining molecules. The atomic forces can be modelled by a Lennard–Jones interaction with the appropriate atomic nitrogen parameters $\sigma = 3.31$ Å, $\varepsilon/k_{\mathrm{B}} = 37.3$ K and $d = 0.3296\sigma$ [46]. The equation of motion for the centre of mass $\mathbf{R}_{\mathrm{CM}}$ is then

$$\ddot{\mathbf{R}}_{\mathrm{CM}} = \mathbf{F}_{\mathrm{tot}}, \tag{8.106}$$

which can be solved in exactly the same way as in an ordinary MD simulation.

The motion of the orientation vector $\hat{\mathbf{n}}$ is determined by the torque $\boldsymbol{\mathcal{N}}$ with respect to the centre of the molecule, which is given in terms of the forces $\mathbf{F}^{(1)}$ and $\mathbf{F}^{(2)}$ acting on atoms 1 and 2 respectively:

$$\boldsymbol{\mathcal{N}} = (d/2)\hat{\mathbf{n}} \times (\mathbf{F}^{(1)} - \mathbf{F}^{(2)}). \tag{8.107}$$

The torque changes the angular momentum $\mathbf{L}$ of the molecule. This is equal to $I\boldsymbol{\omega}$, where I is the moment of inertia $md^2/2$ and $\boldsymbol{\omega}$ is the angular frequency vector whose norm is the angular frequency and whose direction is the axis around which the rotation takes place (see Figure 8.5). Note that $\mathcal{N}$ is not necessarily parallel to $\boldsymbol{\omega}$. The equation of motion for the angular momentum is $\dot{\mathbf{L}} = \boldsymbol{\mathcal{N}}$ or

$$I\dot{\boldsymbol{\omega}} = \boldsymbol{\mathcal{N}}. \tag{8.108}$$

The angular frequency $\boldsymbol{\omega}$ is in turn related to the time derivative of the direction vector $\hat{\mathbf{n}}$:

$$\dot{\hat{n}} = \boldsymbol{\omega} \times \hat{\mathbf{n}}. \tag{8.109}$$

Combining Eqs. (8.108) and (8.109) leads to

$$\ddot{\hat{n}} = \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \hat{\mathbf{n}}) + \boldsymbol{\mathcal{N}} \times \hat{\mathbf{n}}/I = -\omega^2\hat{\mathbf{n}} + \boldsymbol{\mathcal{N}} \times \hat{\mathbf{n}}/I. \tag{8.110}$$

This equation of motion leaves the norm of the direction vector $\hat{\mathbf{n}}$ invariant, as it should – this follows directly from (8.109). In a numerical integration of the

equations of motion the norm of $\hat{\mathbf{n}}$ is not rigorously conserved; it can suffer from numerical errors which may growing steadily with time. We shall now see how this can be avoided.

Let us write down the leap-frog algorithm for the equation of motion (8.110) for $\hat{\mathbf{n}}$:

$$\mathbf{p}(t + h/2) = \mathbf{p}(t - h/2) + h[-\omega^2 \hat{\mathbf{n}}(t) + \boldsymbol{\mathcal{N}}(t) \times \hat{\mathbf{n}}(t)/I] \tag{8.111a}$$

$$\hat{\mathbf{n}}(t + h) = \hat{\mathbf{n}}(t) + h\mathbf{p}(t + h/2). \tag{8.111b}$$

Here $\mathbf{p}$ represents the time-derivative of $\hat{\mathbf{n}}$ at times $t = (n + 1/2)h$. The problem with this algorithm is that it depends on ω^2 and this depends in turn on the time derivative $\dot{\hat{n}}$. A convenient way of finding ω^2 is to use

$$\mathbf{p}(t - h/2) = \mathbf{p}(t) - \frac{h}{2}(-\omega^2 \hat{\mathbf{n}} + \mathcal{N} \times \hat{\mathbf{n}}/I) + \mathcal{O}(h^2), \tag{8.112}$$

so that, using $\hat{\mathbf{n}}(t) \cdot \mathbf{p}(t) = 0$, we obtain

$$2\mathbf{p}(t - h/2) \cdot \hat{\mathbf{n}}(t) = h\omega^2 + \mathcal{O}(h^2). \tag{8.113}$$

Calling the left hand side of this equation λ, we have [2, 47]

$$\lambda = 2\mathbf{p}(t - h/2) \cdot \hat{\mathbf{n}} \tag{8.114a}$$

$$\mathbf{p}(t + h/2) = \mathbf{p}(t - h/2) + h[\hat{\mathbf{n}}(t) \times \mathcal{N}(t)/I - \lambda \hat{\mathbf{n}}(t)] \tag{8.114b}$$

$$\hat{\mathbf{n}}(t + h) = \hat{\mathbf{n}}(t) + h\mathbf{p}(t + h/2). \tag{8.114c}$$

The continuum equations of motion guaranteed conservation of the norm of the unit vector $\hat{\mathbf{n}}$. The leap-frog algorithm will enforce this normalisation only up to an error of h^3. It is therefore sensible to normalise $\hat{\mathbf{n}}$ after every time step – the parameter λ can then be viewed as the Lagrange multiplier associated with the constraint $|\hat{\mathbf{n}}|^2 = 1$. In the next section we shall discuss a simpler method for simulating liquid nitrogen, using constraints in a different way.

For general molecules, we have an extra degree of freedom: the angle of rotation around a molecular axis. This is the third Euler angle, which is usually denoted as γ. The straightforward procedure for solving the equations of motion is to calculate the principal angular velocity $\boldsymbol{\omega}$ in terms of the time derivatives of the Euler angles. The Euler equation of motion gives the rate of change in $\boldsymbol{\omega}$ in terms of the torque. The time derivatives of the Euler angles can then be found again from $\boldsymbol{\omega}$, and these can be used to calculate the new atomic positions. There is however a problem when the Euler angle $\theta = 0$, as in that case the transformation from $\boldsymbol{\omega}$ to the time derivatives of the Euler angles becomes singular. Evans has discussed this problem and has presented methods to avoid the instability resulting from this singularity [48]. The most efficient one is to use the quaternion representation, in which the orientation of the molecule is defined in terms of a four-dimensional unit vector rather than three Euler angles. This method was implemented by Evans and Murad [49].

Enforcing rigidity via constraints

Another method for treating rigid molecules is by imposing holonomic constraints, i.e. constraints which depend only on positions and not on the velocities, through an extension of the Lagrangian. The Lagrangian of the system without constraints reads

$$L_0(R,\dot{R}) = \int_{t_0}^{t_1} \mathrm{d}t \Big[\sum_i \frac{m_i}{2}\dot{\mathbf{r}}_i^2 - \frac{1}{2}\sum_{i\neq j} U(\mathbf{r}_i - \mathbf{r}_j)\Big]. \tag{8.115}$$

A constraint is introduced as usual through a Lagrange multiplier λ [50]. As the constraint under consideration should hold for all times, λ is a function of t. A simple example of a constraint is the following: particles 1 and 2 have a fixed separation d for all times (this could be the separation of the two atoms in a nitrogen molecule). Such a constraint on the separation is called *bond constraint* – it can formally be written as

$$\sigma[R(t)] = [\mathbf{r}_1(t) - \mathbf{r}_2(t)]^2 - d^2 = 0. \tag{8.116}$$

The Lagrangian for the system with this constraint reads

$$L(R,\dot{R}) = L_0(R,\dot{R}) - \int_{t_0}^{t_1} \mathrm{d}t\ \lambda(t)\{[\mathbf{r}_1(t) - \mathbf{r}_2(t)]^2 - d^2\}. \tag{8.117}$$

The integral over time is needed because the constraint holds for all times between t_0 and t_1. The equations of motion are the Euler–Lagrange equations for this Lagrangian. These equations will depend on the Lagrange parameters, λ, whose values are determined by the requirement that the solution must satisfy the constraint.

A slightly more complicated example is the trimer molecule CS_2 [51]. The linear geometry of this molecule is in principle imposed automatically by the correct bond constraints between the three pairs of atoms. However, the motion of this molecule is described by five positional degrees of freedom: two to define the orientation of the molecule and three for the centre of mass position. The three atoms without constraints have nine degrees of freedom and if three of these are eliminated using the bond constraints, we are left with six degrees of freedom instead of the five required. Therefore one redundant degree of freedom is included in this procedure, which is obviously inefficient. A better procedure is therefore to fix only the distance between the two sulphur atoms:

$$|\mathbf{r}_{S^{(1)}} - \mathbf{r}_{S^{(2)}}|^2 = d^2 \tag{8.118}$$

and to fix the position of the C-atom by a linear vector constraint:

$$(\mathbf{r}_{S^{(1)}} + \mathbf{r}_{S^{(2)}})/2 - \mathbf{r}_C = 0, \tag{8.119}$$

adding up to the four constraints required.

For a molecule, in general a number of atoms forming a 'backbone' set is identified and these are fixed by bond constraints (the two sulphur atoms in our example)

and the remaining ones are fixed by linear constraints of the form (8.119). In the case of a planar structure we take three noncollinear atoms as a backbone. These atoms satisfy three bond constraints and the remaining atoms are constrained linearly. In a three-dimensional molecular structure, four backbone atoms are subject to six bond constraints and the remaining ones to a linear vector constraint each. In the constraint procedure, the degrees of freedom of the nonbackbone atoms are eliminated so that the forces they feel are transferred to the backbone. This elimination is always possible for linear constraints such as those obeyed by the nonbackbone atoms.

Let us now return to our CS_2 example. First we write down the equations of motion for all three atoms, following from the extended Lagrangian (the Lagrange parameter for the bond constraint is called λ, that of the linear vector constraint $\boldsymbol{\mu}$):

$$m_S \ddot{\mathbf{r}}_{S^{(1)}} = \mathbf{F}_1 - 2\lambda(\mathbf{r}_{S^{(1)}} - \mathbf{r}_{S^{(2)}}) - \boldsymbol{\mu}/2 \tag{8.120a}$$

$$m_S \ddot{\mathbf{r}}_{S^{(2)}} = \mathbf{F}_2 + 2\lambda(\mathbf{r}_{S^{(1)}} - \mathbf{r}_{S^{(2)}}) - \boldsymbol{\mu}/2 \tag{8.120b}$$

$$m_C \ddot{\mathbf{r}}_C = \mathbf{F}_C + \boldsymbol{\mu}. \tag{8.120c}$$

The linear constraint (8.119) is now differentiated twice with respect to time, and using the equations of motion we obtain

$$\mathbf{F}_C + \boldsymbol{\mu} = \frac{m_C}{2m_S}(\mathbf{F}_1 + \mathbf{F}_2 - \boldsymbol{\mu}). \tag{8.121}$$

We can thus eliminate $\boldsymbol{\mu}$ in the equations of motion for the S-atoms and obtain, with $M = 2m_S + m_C$:

$$m_S \ddot{\mathbf{r}}_{S^{(1)}} = \left(1 - \frac{m_C}{2M}\right)\mathbf{F}_1 - \frac{m_C}{2M}\mathbf{F}_2 + \frac{m_S}{M}\mathbf{F}_C - 2\lambda(\mathbf{r}_{S^{(1)}} - \mathbf{r}_{S^{(2)}}); \tag{8.122a}$$

$$m_S \ddot{\mathbf{r}}_{S^{(2)}} = \left(1 - \frac{m_C}{2M}\right)\mathbf{F}_2 - \frac{m_C}{2M}\mathbf{F}_1 + \frac{m_S}{M}\mathbf{F}_C + 2\lambda(\mathbf{r}_{S^{(1)}} - \mathbf{r}_{S^{(2)}}). \tag{8.122b}$$

These equations define the algorithm for the positions of the S-atoms, and the position of the C-atom is fixed at any time by the linear constraint.

Note that we still have an unknown parameter λ present in the resulting equations: this parameter is fixed by demanding that the bond constraint must hold for $\mathbf{r}_{S^{(1)}}$ and $\mathbf{r}_{S^{(2)}}$ at all times (note that we have not yet used this constraint). It is not easy to eliminate λ from the equations of motion as we have done with $\boldsymbol{\mu}$, as the bond length constraint is quadratic. Instead, we solve for λ at each time step using the constraint equation. We outline this procedure for our example. Suppose we have the positions $\mathbf{r}_{S^{(1)}}$ and $\mathbf{r}_{S^{(2)}}$ at times t and $t - h$ and that for both these time instances the bond constraint is satisfied. According to the equations of motion (8.122) in the

Verlet scheme, predictions for the positions at $t + h$ are given by

$$\mathbf{r}_{S^{(1)}}(t+h) = 2\mathbf{r}_{S^{(1)}}(t) - \mathbf{r}_{S^{(1)}}(t-h) + h^2\left(1 - \frac{m_C}{M}\right)\mathbf{F}_1(t) - h^2\frac{m_C}{M}\mathbf{F}_2 + h^2\frac{m_S}{M}F_C(t) - 2h^2\lambda(t)[\mathbf{r}_{S^{(1)}}(t) - \mathbf{r}_{S^{(2)}}(t)]; \quad (8.123a)$$

$$\mathbf{r}_{S^{(2)}}(t+h) = 2\mathbf{r}_{S^{(2)}}(t) - \mathbf{r}_{S^{(2)}}(t-h) + h^2\left(1 - \frac{m_C}{M}\right)\mathbf{F}_2(t) - h^2\frac{m_C}{M}\mathbf{F}_1 + h^2\frac{m_S}{M}\mathbf{F}_C(t) + 2h^2\lambda(t)[\mathbf{r}_{S^{(1)}}(t) - \mathbf{r}_{S^{(2)}}(t)]. \quad (8.123b)$$

The predictions for the positions at $t+h$ are linear functions of λ and if we substitute them into the bond constraint (8.118), we obtain a quadratic equation for λ which can be solved trivially. Of the two solutions, we keep the smallest value of λ. This means that the bond constraint is now satisfied to computer precision for all times. It should be noted that the value of λ in this procedure will deviate slightly from its value in the exact solution of the continuum case, but the deviation remains within the overall order h^4 error of the integration scheme [52].

We have given the CS_2 example here because it illustrates the general procedure involving linear constraints which are all eliminated from the equations of motion, thereby reducing the degrees of freedom to those of the backbone atoms (the two sulphur atoms in our example). These are confined by quadratic bond constraints. The Lagrange multipliers associated with the latter are kept in the problem and fixed by the bond constraints themselves. After solving for the backbone, the linear constraints fix the positions of the remaining atoms uniquely.

The nitrogen molecule which was discussed in the previous subsection using a direct approach can be treated with the method of constraints. It is a simple problem because there are no linear constraints which have to be used to remove redundant degrees of freedom from the equations of motion, and we are left with the following equations:

$$m_1\ddot{\mathbf{r}}_1 = \mathbf{F}_1 + \lambda(\mathbf{r}_1 - \mathbf{r}_2) \quad (8.124a)$$

$$m_2\ddot{\mathbf{r}}_2 = \mathbf{F}_2 - \lambda(\mathbf{r}_1 - \mathbf{r}_2). \quad (8.124b)$$

The Verlet equations lead again to linear predictions for $\mathbf{r}_1$ and $\mathbf{r}_2$ at the next time step and substituting these into the bond constraint leads to a quadratic equation which fixes the Lagrange multiplier λ. For an implementation, see Problem 8.9.

8.6.3 General procedure: partial constraints

In the previous section we have considered systems consisting of completely rigid molecules. Now we discuss partially rigid molecules, consisting of rigid fragments which can move with respect to each other. The motion of two fragments attached

by chemical bonds can be described in terms of stretching, bending and torsion, as described in Section 8.6.2. In general, partial constraints cannot be treated using the methods given previously. Trying to use the constraints to reduce the equations to a smaller set and formulating equations for the rigid fragments in terms of quaternions is quite complicated. Ryckaert *et al.* [51–54] devised a simple and efficient iterative method for treating arbitrary constraints which is now still the most important method for MD with polyatomic molecules. Analogous to the method of constraints for rigid molecules, the rigidity of the fragments can be imposed by constraints, which are all expressed in Cartesian coordinates for simplicity. The Lagrange multipliers are determined after each integration step by substituting the new positions into the constraint equations.

The algorithm, called SHAKE, is formulated within the framework of the Verlet algorithm. The forces experienced by the particles consist of physical and of constraint forces. The constraints are given by $\sigma_k(R) = 0$, where $k = 1, \ldots, M$; M is the number of constraints. We denote the physical force on particle i by $\mathbf{F}_i$ and the constraint force is $\sum_{k=1}^{M} \lambda_k \nabla_i \sigma_k(R)$, where λ_k is the Lagrange multiplier which is to be determined. At time step $t = nh$ we have at our disposal the positions at times t and $t - h$. These positions satisfy the constraint equations $\sigma_k(R) = 0$ to numerical precision. The aim is to find the positions at time $t + h$, satisfying the constraint equation. First we calculate the new positions $\tilde{\mathbf{r}}_i(t + h)$ without taking the constraints into account:

$$\tilde{\mathbf{r}}_i(t + h) = 2\mathbf{r}_i(t) - \mathbf{r}_i(t - h) + h^2 \mathbf{F}_i[\mathbf{r}_i(t)]. \tag{8.125}$$

The final positions $\mathbf{r}_i(t + h)$ can be written as

$$\mathbf{r}_i(t + h) = \tilde{\mathbf{r}}_i(t + h) - \sum_{k=1}^{M} \lambda_k \nabla_i \sigma_k[R(t)]. \tag{8.126}$$

The λ_k are found in an iterative procedure. We number the iterations by an index l. In each iteration, a loop over the constraints k is carried out, and in each step of this loop, the Lagrange parameter λ_k and all the particle positions are updated. The positions are updated according to

$$\mathbf{r}_i^{\text{new}} = \mathbf{r}_i^{\text{old}} - h^2 \lambda_k^{(l)} \nabla_i \sigma_k[R(t)]. \tag{8.127}$$

The parameter $\lambda_k^{(l)}$ is found from a first order expansion of $\sigma_k(R)$, requiring that this vanishes:

$$\sigma_k[R^{\text{new}}] \approx \sigma_k[R^{\text{old}}] - h^2 \lambda_k^{(l)} \sum_i \nabla_i \sigma_k[R^{\text{old}}] \nabla_i \sigma_k[R(t)] = 0, \tag{8.128}$$

leading to

$$\lambda_k^{(l)} = \frac{\sigma_k[R^{\text{old}}]}{h^2 \left\{ \sum_i \nabla_i \sigma_k[R^{\text{old}}] \nabla_i \sigma_k[R(t)] \right\}}. \tag{8.129}$$

Each step will therefore shift the positions more closely to the point where they all satisfy the constraint. The iterative process is stopped when all the constraints are smaller (in absolute value) than some small positive number.

The algorithm can be summarised as follows:

Calculate $\tilde{R}(t+h)$ using (8.125);
Set R^{old} equal to $\tilde{R}(t+h)$;
REPEAT
  Calculate $\lambda_k^{(l)}$ from (8.129);
  FOR $k = 1$ TO M DO
    Set R^{old} equal to R^{new}
    Update R^{old} to R^{new} using (8.127);
  END FOR
UNTIL Constraints are satisfied.

The SHAKE algorithm turns out to be quite efficient: for a system of 48 atoms with 112 constraints, typically 25 iterations are necessary in order to achieve convergence of the constraints within a relative accuracy of 5×10^{-7} [52].

8.7 Long-range interactions

Coulombic and gravitational many-particle systems are of great interest because they describe plasmas, electrolytic solutions, and celestial mechanical systems. The interaction is described by a pair-potential which in three dimensions is proportional to $1/r$ – in two dimensions it is $\ln r$. The long range character of this potential poses problems. First of all, it is not clear whether the potential can be cut off beyond some finite range. One might hope that for a charge-neutral Coulomb system, screening effects could justify this procedure. Unfortunately, for most systems of interest, the screening length exceeds half the linear system size that can be achieved in practice, so we cannot rely on this screening effect to justify cutting off the potential, as this would essentially alter the form of the screening charge cloud. Also, when using the minimum image convention with periodic boundary conditions, equally charged particles tend to occupy opposite ends of a half diagonal of the system unit cell in order to minimise their interaction energy, thus introducing unphysical anisotropies. Therefore, we cannot cut off the potential and all pairs of interacting particles must be taken into account when calculating the forces.

Connected with this is an essential difference in the treatment of periodic or nonperiodic system cells. In the latter case, we simply use the $1/r$ potential (or $\ln r$ in two dimensions), but in the periodic case we must face the problem that in general the sum over the image charges in the periodic replicas does not converge. This can be remedied by subtracting an offset from the potential (note that adding

or subtracting a constant to the potential does not alter the forces and hence the dynamics of the system) leading to the following expression for the total configurational energy for a collection of particles with charge (or mass) q_i located at q_i, $i = 1, \ldots, N$:

$$U = \sum_{\mathbf{R}} \sum_{i<j} \frac{q_i q_j}{|\mathbf{r}_i - \mathbf{r}_j + \mathbf{R}|} - \sum_{i<j} q_i q_j \sum_{\mathbf{R}}{}' \frac{1}{\mathbf{R}}. \tag{8.130}$$

Here $\sum_{i<j}$ denotes a sum over i and j running from 1 to N with the restriction $i < j$; furthermore, $\sum_{\mathbf{R}}$ denotes a sum over the locations $\mathbf{R}$ of the system replicas, the prime with the second sum denoting exclusion of $\mathbf{R} = \mathbf{0}$. From now on we shall restrict ourselves to charge-neutral systems with $\sum_i q_i = 0$, for which the second term in (8.130) vanishes. The system then has a dipole moment and the leading term in computing the total energy in PBC is the result of the dipole–dipole interactions between the replicas. Evaluating the sum over the replicas is a difficult problem even for charge-neutral systems and it will be addressed in the next subsection. In Section 8.7.2 we shall then see how the forces can be evaluated more efficiently than in the conventional MD approach where we must sum over all pairs.

8.7.1 The periodic Coulomb interaction

The total configurational energy of the charge-neutral system is given by

$$U = \sum_{\mathbf{R}} \sum_{i<j} \frac{q_i q_j}{|\mathbf{r}_i - \mathbf{r}_j + \mathbf{R}|}; \quad \sum_i q_i = 0. \tag{8.131}$$

It is assumed here that the particles are point particles, that is, their charge distribution is given by a delta-function $\rho_i(\mathbf{r}) = q_i \delta(\mathbf{r} - \mathbf{r}_i)$. In most realistic cases there will be additional short range interactions preventing the particles from approaching each other too closely. We now apply a Fourier transform as defined in Eqs. (4.104)–(4.105) to (8.131). We have

$$\frac{1}{r} = \int \frac{\mathrm{d}^3 k}{(2\pi)^3} \frac{4\pi}{k^2} \mathrm{e}^{i\mathbf{k}\cdot\mathbf{r}}. \tag{8.132}$$

Substituting this into (8.131) and using

$$\sum_{\mathbf{R}} \mathrm{e}^{\mathrm{i}\mathbf{k}\cdot\mathbf{R}} = \frac{(2\pi)^3}{V} \sum_{\mathbf{K}} \delta^3(\mathbf{k} - \mathbf{K}) \tag{8.133}$$

where V is the volume of the system and $\mathbf{K}$ are reciprocal lattice vectors, we obtain

$$U = \frac{1}{V} \sum_{\mathbf{K}\neq\mathbf{0}} \sum_{i<j} \frac{\mathrm{e}^{\mathrm{i}\mathbf{K}\cdot\mathbf{r}_{ij}}}{K^2} q_i q_j. \tag{8.134}$$

We have not yet made any progress as we have only replaced the infinite sum over $\mathbf{R}$ by another infinite sum over $\mathbf{K}$. It might seem that this sum diverges for small $\mathbf{K}$, but this is not the case for charge-neutral systems: this neutrality is responsible for the exclusion of the $\mathbf{K} = \mathbf{0}$ term, and it ensures convergence of the small-$\mathbf{K}$ terms. Surprisingly, the divergences in the original real-space sum (8.131) were associated with the long range character of the force whereas the divergence in (8.134) is due to the short range (large $\mathbf{K}$) part. In reality, the ions have a finite size, which means that they will repel each other at short distances. This implies that the Coulomb interaction has to be taken into account for ranges beyond some small cut-off r_{core} only, and we can neglect the K-values for $K > 2\pi/r_{\text{core}}$. Of course, this does not yield an exactly spherical cut-off as the reciprocal lattice is cubic, but if the cut-off radius is sufficiently small this will cause no significant errors. Moreover, the core radius can be chosen much smaller than the range of repulsive interaction (which is always present in realistic models) so that this error can be reduced arbitrarily.

In case one insists on having delta-function distributions, or if the cut-off radius is so small that calculating the Fourier sum is still inconveniently demanding, it is possible first to replace the delta-charges by artificial, extended charge distributions with some finite radius and then correcting for this replacement. This is done in the so-called Ewald summation technique. We shall not give a full derivation of the Ewald summation method since it is quite lengthy – it is described elsewhere [55, 56] – but sketch briefly the idea behind this technique. In the Ewald method, the extended charge distribution is taken to be a Gaussian:

$$\rho_i(\mathbf{r}) = q_i \left(\frac{\alpha}{\pi}\right)^{3/2} \exp(-\alpha|\mathbf{r} - \mathbf{r}_i|^2) \tag{8.135}$$

where the normalisation factor is for the three-dimensional case. This charge distribution results in a converging K-sum, and this extension is corrected for by adding the potential resulting from the difference between the point-charge and Gaussian distribution. Since this difference is neutral, it generates a rapidly decaying potential, which can then be treated by the minimum image convention. The total interaction potential for charges q_i located at $\mathbf{r}_i$ is then given as

$$U_{\text{PBC}} = \frac{2\pi}{V} \sum_{\mathbf{K}\neq\mathbf{0}} \left| \sum_i q_i \mathrm{e}^{\mathrm{i}\mathbf{K}\cdot\mathbf{r}_i} \right|^2 \frac{\mathrm{e}^{-K^2/(4\alpha)}}{K^2} + \sum_{i<j} q_i q_j \frac{\operatorname{erfc}(\sqrt{\alpha} r_{ij})}{r_{ij}} - \left(\frac{\alpha}{\pi}\right)^{1/2} \sum_{i=1}^{N} q_i^2 \tag{8.136}$$

where the function erfc is the complementary error function defined in (4.116): $\operatorname{erfc} = 1 - \operatorname{erf}$. The first term of the Ewald sum converges rapidly due to the $\exp[-K^2/(4\alpha)]$ term resulting from the Gaussian charge distribution. The second term in the sum is short ranged, so it can be treated in a minimum image convention. The forces can be found by differentiation. The Ewald sum can also be generalised for dipolar interactions (Ewald–Kornfeld method).

In a careful treatment of the Ewald technique, the sum is carried out formally by taking a large volume of some particular shape (e.g. a sphere) containing the system replicas and then this shape is increased. The reason for this is that the sum over the interactions is conditionally convergent, i.e. it depends on the order in which the various contributions are taken into account. This is explained by the fact that the system replicas all have a dipole moment and will hence build up a surface charge at the boundary of the huge volume. The most natural boundary condition (the one which is arrived at in more pedestrian derivations) is consistent with the sphere being embedded in a perfectly conducting medium. For the sphere embedded in a dielectric, a correction must be included [56]. It is important to be aware of this when calculating (say) dielectric properties of a charged system.

8.7.2 Efficient evaluation of forces and potentials

As a result of the long range of the forces, all interacting pairs must be taken into account in the calculation of forces or potentials. The straightforward implementation, considered in the previous sections of this chapter, also called the *particle–particle method* (PP) because all pairs are considered explicitly, would require $\mathcal{O}(N^2)$ steps, but it turns out possible to reduce this to a more favourable scaling. We shall briefly sketch two other methods, and then consider a third one, the *tree method* in greater detail.

In the *particle-mesh* (PM) method, a (usually cubic) grid in space is defined. A mass (or charge) distribution ρ is then defined by assigning part of each particle's mass (or charge) to its four neighbouring grid points according to some suitable scheme. The potential can then be found by solving Poisson's equation on the grid

$$\nabla_{\mathrm{D}}^2 U(\mathbf{r}) = -4\pi\rho(\mathbf{r}) \tag{8.137}$$

where ∇_{D}^2 is the finite-difference version of the Laplace operator. Using fast Fourier transforms (see Appendix A9), this calculation can be carried out in a number of steps proportional to $M \log M$ where M is the number of grid points. Knowing the potential, the force at any position can be found by taking the finite difference gradient of the potential, after a suitable correction for the self-energy resulting from the inclusion of the interaction of a particle with itself in this procedure. This method obviously becomes less accurate for pairs of particles with a small separation, as in that case the Coulomb/gravitation potential is not sufficiently smooth to be represented accurately on the grid. Therefore it is sensible to treat particles within some small range (for example a range comparable to the grid constant) by the PP method. This can be done by splitting the force into a smooth long range (LR) and a short range (SR) part:

$$\mathbf{F} = \mathbf{F}^{\mathrm{LR}} + \mathbf{F}^{\mathrm{SR}}, \tag{8.138}$$

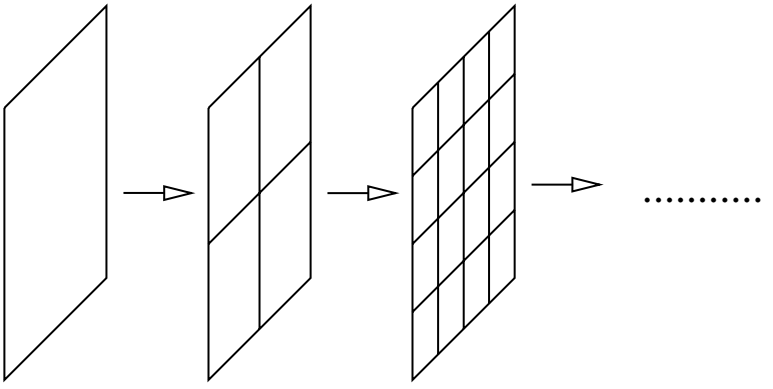

Figure 8.6. Hierarchical subdivision of the full simulation space (a square) into children, grandchildren etc.

such that the short range force vanishes beyond some small range, and the long range force can be calculated accurately on the grid. The splitting can be obtained by considering the long range force as resulting from a particle whose charge (or mass) is distributed over some finite range (homogeneous sphere, Gaussian distribution, …). The short range force is then the potential resulting from the difference between the point charge and the finite-width distribution (cf. the Ewald method). The long range interactions are treated as in the PM method, and the short range ones can be dealt with using the PP scheme. The resulting method is called the *particle–particle/particle–mesh* (PPPM) or P^3M method. For a detailed description of the PM and PPPM methods, see Ref. [19].

We now describe the *tree-code* algorithm in some detail [57–60]. The amount of computer time involved in the evaluation of the forces in this method is reduced to $\mathcal{O}(N \ln N)$ steps. We describe the Barnes–Hut[57] version in the formulation by van Dommelen and Rundensteiner [61, 62] and restrict ourselves to two-dimensional gravitational (or Coulomb) systems, with an interaction $\ln r$ between two particles of unit mass (or charge) and separation r. The idea of the method is that the force which a mass experiences from a distant cluster of particles can be calculated from a multipole expansion of the cluster. The convergence of the multipole expansion depends on the ratio of the distance from the cluster and its linear size.

The total system volume is hierarchically divided up into blocks. We start with a square shape (level 0) which in a first step is divided into four squares of half the linear size (level 1), and at the next step each of these squares is divided up into four smaller ones etc. We speak of parents and children of squares in this hierarchy – see Figure 8.6. Now consider some square S at level n. It is not justified to apply the multipole expansion to nearest neighbour squares as particles in neighbouring

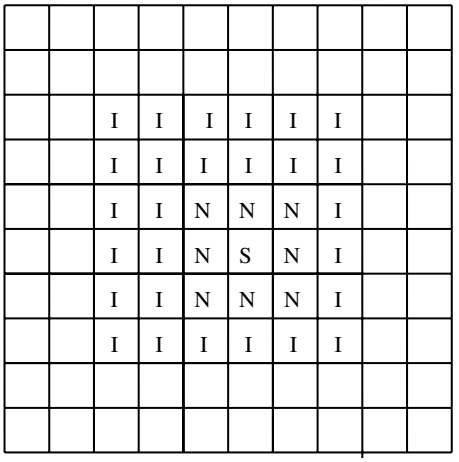

Figure 8.7. Interaction list of a square S at level n. The squares at level n are separated by thin lines, their parents (at level $n-1$) by heavy lines. For the square labelled by S, the squares in the interaction list of a square are denoted by I. The nearest neighbours are labelled by N.

squares might be very close so that the multipole expansion would require far too many terms. These squares will be dealt with at a higher level, so we apply this approximation in each step to at least next nearest squares and skip squares lying in regions that have been treated at previous levels. Therefore, the squares with which the particles in S will interact at the present level are those (1) which are not nearest neighbours of S and (2) whose parent was a nearest neighbour of the parent of S. These squares form the *interaction list* of S. Figure 8.7 shows which squares are in the interaction list of S. It will be clear that all the interactions will be taken into account when proceeding in this way.

More specifically, at level n we carry out two steps.

1. We calculate the multipole moments of each square of the present level.
2. For each particle, we calculate the interactions with the interaction list of the square to which it belongs using the multipole expansion for the particles in the cells.

This process is carried through over $n_{\max} = (\log_2 N)/2$ steps so that for N being an integer power of 4, the squares at the lowest level contain on average one particle. Empty squares are 'pruned' from the tree, that is, they are not divided up any more.

Let us now calculate the number of steps needed in this procedure. We assume that we carry out the multipole expansion up to order M. This number is independent of the number of particles N. At level n, the first step, in which the multipole moments are calculated, requires $N \times M$ steps. The second step requires $N \times M \times K$ steps,

where K is the average size of the interaction list, which is at most 27. K and M are fixed numbers, there are $\mathcal{O}(\ln N)$ levels, so the total number of steps scales as $\mathcal{O}(N \ln N)$.

For two dimensions, the multipole moment expansion is very simple if the space is viewed as a complex plane with particles at positions $z = x + iy$. The potential is then given as the real part of $\ln(z)$ and this can easily be expanded in a Taylor expansion around the centre of the cell. For a cluster centred at the origin and consisting of particles of charge q_i at positions z_i, the potential at the point z is given by

$$U(z) = \sum_{i=1}^{N_c} q_i \ln(z - z_i) = a_0 \ln z - \sum_{k=1}^{M} \frac{a_k}{z^k} + \mathcal{O}\left(\frac{R}{z}\right)^{M+1} \tag{8.139}$$

where R is the linear size of the cluster containing N_c particles and the moment expansion coefficients a_k are given by

$$a_0 = \sum_i q_i \quad \text{and} \quad a_k = \sum_{i=1}^{N_c} \frac{q_i z_i^k}{k}, k \geq 1. \tag{8.140}$$

For the field written as a complex number E at the point z we have

$$E(z) = \sum_{k=0}^{M} \frac{a_k}{z^{k+1}} + \mathcal{O}(R/z)^{M+1}. \tag{8.141}$$

From Figure 8.7 it can be seen that a worst case estimate for R/z is 2/3. In practice, fewer than 20 multipole coefficients are necessary to obtain machine accuracy (32 bits).

In fact, it turns out to be possible to reduce the amount of work needed for the force evaluation to $\mathcal{O}(N)$. The resulting method is called the fast multipole method (FMM) – see Refs. [62] and [63].

8.8 Langevin dynamics simulation

Most realistic physical systems are tractable only in a *model*, in which the interesting features of the system are highlighted and in which the less relevant parts are either eliminated or treated in an approximate way. In this spirit we have for example eliminated molecular degrees of freedom in Section 8.6 by considering (parts of) molecules to be rigid. Another example of this approach is the Langevin dynamics technique, the subject of the present section. Consider a solution containing polymers or ions which are much heavier than the solvent molecules. As the kinetic energy is on average divided equally over the degrees of freedom, the ions or polymers will move much more slowly than the solvent molecules. Moreover,

because of their large mass, they will change their momenta only after many collisions with the solvent molecules and the picture which emerges is that of the heavy particles forming a system with a much longer time scale than the solvent molecules. This difference in time scale can be employed to eliminate the details of the degrees of freedom of the solvent particles and represent their effect by forces that can be treated in a simple way. This process can be carried out analytically through a projection procedure (see chapter 9 of Ref. [11] and references therein) but here we shall sketch the method in a heuristic way.

How can we model the effect of the solvent particles without taking into account their degrees of freedom explicitly? When a heavy particle is moving through the solvent, it will encounter more solvent particles at the front than at the back. Therefore, the collisions with the solvent particles will *on average* have the effect of a friction force proportional and opposite to the velocity of the heavy particle. This suggests the following equation of motion for the heavy particle:

$$m\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t}(t) = -\gamma\mathbf{v}(t) + \mathbf{F}(t) \tag{8.142}$$

where γ is the friction coefficient and $\mathbf{F}$ the external or systematic force, due to the other heavy particles, walls, gravitation, etc. It has been noted in Section 7.2.1 that the motion of fluid particles exhibits strong time correlations and therefore the effects of their collisions should show time correlation effects. Time correlations affect the form of the friction term which, in Eq. (8.142), has been taken to be dependent on the *instantaneous* velocity but which in a more careful treatment should include contributions from the velocity at previous times through a memory kernel:

$$m\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t}(t) = -\int_{-\infty}^{t} \mathrm{d}t'\ \gamma(t-t')\mathbf{v}(t') + \mathbf{F}(t). \tag{8.143}$$

In order to avoid complications we shall proceed with the simpler form (8.142). In the following we shall restrict ourselves to a particle in one dimension; the analysis for more particles in two or three dimensions is similar.

Equation (8.142) has the unrealistic effect that if the external forces are absent, the heavy particle comes to rest, whereas in reality it executes a Brownian motion. To make the model more realistic we must include the rapid variations in the force due to the frequent collisions with solvent particles on top of the coarse-grained friction force. We then arrive at the following equation:

$$m\frac{\mathrm{d}v}{\mathrm{d}t}(t) = -\gamma v(t) + F(t) + R(t) \tag{8.144}$$

where $R(t)$ is a 'random force'. Again, the time correlations present in the fluid should show up in this force, but they are neglected once more and the force is

subject to the following conditions.

- As the average effect of the collisions is already absorbed in the friction, the expectation value of the random force should vanish:

$$\langle R(t)\rangle = 0. \tag{8.145}$$

- The values of R are taken to be uncorrelated:

$$\langle R(t)R(t+\tau)\rangle = 0 \quad \text{for } \tau > 0. \tag{8.146}$$

- The values of R are distributed according to a Gaussian:

$$P[R(t)] = (2\pi\langle R^2\rangle)^{-1/2}\exp(-R^2/2\langle R^2\rangle). \tag{8.147}$$

All these assumptions can be summarised in the following prescription for the probability for a set of random forces to occur between t_0 and t_1:

$$P[R_i(t)]_{t_0<t<t_1} \propto \exp\left(-\frac{1}{2q}\int_{t_0}^{t_1} \mathrm{d}t\, R_i^2(t)\right) \tag{8.148}$$

with q a constant to be determined.

Below we consider the numerical integration of the equations of motion for the heavy particles, and in that case it is convenient to assume that the random force is constant over each time step: at step n, the value of the random force is R_n. For this case, the correlation function for the R_n reads

$$\langle R_n R_m\rangle = \frac{\int \mathrm{d}R_n \mathrm{d}R_{n+1}\cdots\mathrm{d}R_m \exp(-1/2q\sum_{l=n}^{m} R_l^2\Delta t)R_nR_m}{\int \mathrm{d}R_n \mathrm{d}R_{n+1}\cdots\mathrm{d}R_m \exp(-1/2q\sum_{l=n}^{m} R_l^2\Delta t)} \tag{8.149}$$

which yields the value 0 for $n \neq m$, in accordance with the previous assumptions. For $n = m$ we find the value $q/\Delta t$, so we arrive at

$$\langle R_n R_m\rangle = \frac{q}{\Delta t}\delta_{nm}. \tag{8.150}$$

For the continuum case $\Delta t \to 0$ (8.150) converges to the δ-distribution function

$$\langle R(t)R(t+\tau)\rangle = q\delta(\tau). \tag{8.151}$$

We now return to the continuum form of the Langevin equation (8.144). This can be solved analytically and the result is

$$v(t) = v(0)\exp(-\gamma t/m) + \frac{1}{m}\int_0^t \exp[-(t-\tau)\gamma/m]R(\tau)\mathrm{d}\tau. \tag{8.152}$$

Because the expectation value of R vanishes we obtain

$$\langle v(t)\rangle = v(0)\exp(-\gamma t/m) \tag{8.153}$$

which is to be expected for a particle subject to a friction force proportional and opposite to the velocity.

The expectation value of v^2 is determined in a similar way. Using (8.151) and (8.144) we find

$$\langle [v(t)]^2 \rangle = v_0^2 \exp(-2\gamma t/m) + \frac{q}{2\gamma m}(1 - \mathrm{e}^{-2\gamma t/m}), \tag{8.154}$$

which for large t reduces to

$$\langle [v(\infty)]^2 \rangle = \frac{q}{2\gamma m}. \tag{8.155}$$

According to (8.152), v depends linearly on the random forces $R(t)$ and as the latter are distributed according to a Gaussian, the same will hold for the velocity. The width is given by (8.155), so we have

$$P[v(t)] = \left(\frac{\gamma m}{\pi q}\right)^{1/2} \exp[-mv(t)^2\gamma/q] \tag{8.156}$$

for large t. This is precisely the Maxwell distribution if we write

$$q = 2k_\mathrm{B}T\gamma, \tag{8.157}$$

so this equation defines the value of q necessary to obtain a system with temperature T. In Chapter 12 we shall discuss Langevin types of equations in a more formal way, using the Fokker–Planck equation.

The velocity autocorrelation function can also be obtained from (8.152):

$$\langle v(0)v(t) \rangle = \langle v(0)^2 \rangle \mathrm{e}^{-\gamma t/m}. \tag{8.158}$$

The absence of a long time tail in this correlation function reflects the oversimplifications in the construction of the Langevin equation, in particular the absence of correlations in the random force and the fact that the frictional force does not depend on the 'history' of the system.

The results presented here are easily generalised to more than one dimension. However, including a force acting between the heavy particles causes problems if this force exhibits correlations with the random force, and Eq. (8.157) is no longer valid in that case. Such correlation effects are often neglected and the systematic force is simply added to the friction and the Langevin term.

A further refinement is the inclusion of memory kernels in the forces, similar to the approach in Eq. (8.143). In that case, the random force is no longer uncorrelated – it is constructed with correlations in accordance with the fluctuation-dissipation theorem [64]:

$$\langle R(0)R(t) \rangle = \langle v^2 \rangle \gamma(t). \tag{8.159}$$

However, this equation is again no longer valid if external forces are included. The approach with memory kernels has led to a whole industry of so-called generalised Langevin-dynamics simulations [64–67].

The systematic interaction force between the particles in the solvent will affect the friction which these particles are subject to through hydrodynamic effects. This coupling is usually neglected, but a method including these effects has been proposed and implemented [68]. We mention the dissipative particle dynamics (DPD) method which is based on these ideas [69].

An algorithm for simple Langevin dynamics can be formulated starting from the methods given in Section 8.4.1. Suppose the random force is constant during one integration step. Denoting the force during the interval $[0, h]$ by R_+ and that during the interval $[-h, 0]$ by R_-, the random force can directly be included into (8.40):

$$x(h)[1 + \gamma h/2] + x(-h)[1 - \gamma h/2] = 2x(0) + h^2[F(0) + R_+/2 + R_-/2]. \tag{8.160}$$

Therefore, at each step a new value of the random force during the new interval must be drawn from a Gaussian random generator, and this force is to be used together with the random force generated at the previous step in order to predict the new position. This is, however, not always a satisfactory procedure. Normally, the integration time step h is determined by the requirement that the systematic force $\mathbf{F}$ can be assumed to be reasonably constant over a time interval h. This means that the time over which we take the random force to be constant depends on the smoothness of the systematic force. In fact we would prefer to allow for a rapidly varying random force combined with a large time step allowed by the systematic force. This turns out to be possible. Using the statistical properties of the random force, equations of motion can be obtained which are somewhat similar to the ones given here, but with more complicated correlations between the random contributions at subsequent steps – for details see Ref. [70].

It is straightforward to develop a Langevin program for a molecule in a fluid or a gas, using the simple algorithm presented here. For molecules containing chains of at most three chemically bonded atoms, torsion is absent, which reduces the number of forces considerably. Examples are molecules with a tetrahedron conformation, such as CH_4 (methane) and CF_4, and two-dimensional molecules. In Problem 8.9 the construction of a Langevin molecule for methane is considered.

8.9 Dynamical quantities: nonequilibrium molecular dynamics

In the molecular dynamics method, the equations of motion of a classical many-body system are integrated numerically. There is no reason to restrict the applicability of this method to systems in equilibrium. MD is the method of choice for dynamic phenomena in equilibrium or nonequilibrium systems. We speak of nonequilibrium molecular dynamics (NEMD). We consider two examples very briefly here.

There exists a relation between time correlation functions and transport coefficients via the dynamic fluctuation-dissipation theorem [71, 72]. The physical idea behind this theorem is that, in an equilibrium system, particles diffuse and the dynamics of this diffusion tells us something about their ability to transport for example heat or charge. Therefore we can measure transport coefficients by studying the diffusion of the positions or velocities through the system. A disadvantage of measuring transport quantities in this way is that diffusion is often rather slow in equilibrium so that accurate results for transport coefficients are sometimes hard to obtain. Therefore it is useful to apply a field and measure the response to the action of that field directly by keeping track of the motion of the particles (a thermostat must be used in order to prevent the energy from increasing steadily as a result of the interaction with the external field). A complication may arise in connection with periodic boundary conditions, as in that case surface effects may be induced if the applied force is not compatible with the periodicity. Therefore perturbing forces are often chosen sinusoidal with a period compatible with the PBC. An example is provided by the determination of the shear viscosity, caused by fluid layers moving in parallel directions, with different speed, rubbing against each other. The shear viscosity can be measured [73, 74] by applying a force in the x-direction which varies with the coordinate z according to

$$\mathbf{F}(z) = F_0 \cos(kz)\hat{\mathbf{x}} \tag{8.161}$$

where $k = 2\pi/L$, and L is the linear size of the cubic volume. The shear viscosity η can then be measured via the mean velocity in the x-direction of the particles with a given coordinate z:

$$\overline{v_x}(z) = \rho/(k^2\eta)F_0 \cos(kz) \tag{8.162}$$

and this average velocity can easily be determined. In order to improve the estimate one can determine the shear viscosity with various $k_n = 2\pi n/L$ to extrapolate to $k \to 0$.

A second example of NEMD is the transfer of energy between different degrees of freedom. This is of interest in detonation waves. A detonation which traverses a medium of explosive molecules continuously 'recharges' itself by new unstable molecules falling apart, thereby releasing fragments with high velocities. For an unstable molecule to be disrupted it is necessary for the translational energy imparted by a collision with a fast fragment to be transferred to bond length vibrations. For diatomic molecules, the two different degrees of freedom can easily be separated. Holian *et al.* [39, 40] have carried out MD simulations in which the translational and vibrational degrees of freedom were given different temperatures by coupling them to different heat baths which were then turned off or replaced by a single bath (at the higher temperature). In this way it was possible to determine energy transfer rates between the different modes.

Exercises

8.1 [C] For coding the leap-frog method (Eq. (8.7)) two arrays are needed, one containing the velocities at times $t = (n + 1/2)h$, and one for the positions at $t = nh$. The same holds for the velocity-Verlet algorithm.

At first sight it might seem that the Verlet algorithm would need more memory: arrays containing the positions at times $t = nh$, $t = (n - 1)h$ and $t = (n + 1)h$. However, the value $x_i[(n - 1)h]$ can be overwritten by $x_i[(n + 1)h]$. Use this to code the Verlet algorithm such that only two arrays are needed. Test it for a number of particles moving in one dimension and subject to the harmonic oscillator potential.

8.2 The neighbour list proposed by Verlet [8] needs updating every 10–20 integration steps and this update requires of the order of N^2 steps for a system containing N particles. Another bookkeeping device consists of partitioning the system into cubic volumes and keeping track of which particles are to be found in each of these volumes. Consider a two-dimensional $L \times L$ system for convenience. We split this up into $P \times P$ squares of linear size L/P. P is chosen such that the potential can be cut off safely beyond L/P. Suppose we have for each square a list of particles within that volume. These lists will change whenever a particle leaves a square and moves to a neighbouring one. The force evaluation now includes only particle pairs whose members are either in the same or in neighbouring cells.

(a) How many particles are on average to be found in one square?
(b) How many pair forces are on average taken into account in this 'cell method'?
(c) Calculate the gain in speed with respect to the method in which all pair interactions are taken into account, assuming that the particles are distributed more or less homogeneously over the volume.

8.3 The first molecular dynamics simulations were carried out by Alder and Wainwright for hard spheres [14]. The discontinuity in the potential calls for a different approach than that used for smooth potentials. The state of the system is given by the positions $\mathbf{r}_i$ and velocities $\mathbf{v}_i$ (i labels the particles) at some time t_i which is usually the time of the last collision experienced by i. We must calculate the velocity changes for the next pair undergoing a collision.

We consider the elastic collision between two hard spheres, i and j, which are moving with velocities $\mathbf{v}_i$ and $\mathbf{v}_j$. At time t their positions are $\mathbf{r}_i$ and $\mathbf{r}_j$. After the collision, velocities are $\mathbf{v}_i'$ and $\mathbf{v}_j'$ respectively. The sphere diameter is σ.

(a) Show, using energy and momentum conservation, that the changes in velocities of the two particles are given by

$$\Delta\mathbf{v}_i = \mathbf{v}_i' - \mathbf{v}_i = -\Delta\mathbf{v}_j = \frac{\mathbf{r}_{ij}(\mathbf{v}_{ij} \cdot \mathbf{r}_{ij})}{\sigma^2}$$

where $\mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j$ and $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ at the collision.

For each pair of particles we need to know the time at which they will collide (note that because of PBC each pair will indeed collide at some time unless the

velocities have very peculiar values). The collision time for pair i,j is found by

$$|\mathbf{r}_{ij} + t\mathbf{v}_{ij}| = \sigma.$$

This is a quadratic equation which yields two solutions for the collision time t. The first time after the current time must be chosen and recorded as the collision time of pair ij.

The simulation is now constructed as follows. At the beginning, the particles are released from a lattice with velocities according to a Boltzmann distribution. For all $N(N-1)/2$ pairs, the collision times are calculated and stored in a sorted list. The first element of this list contains the first collision to take place. For this collision we calculate the new velocities and positions. Then each pair containing at least one of the two collision partners is removed from the list. Their new collision times are calculated and added again to the list in such a way that the latter remains sorted with respect to the collision times.

(b) How does the simulation time scale with the number of particles?

(c) Explain why the kinetic energy of the hard sphere system is rigorously constant.

In order to calculate pressures we must adapt the virial theorem to this system. The virial theorem for smooth forces reads

$$\frac{\beta P}{\rho} = 1 + \frac{1}{3Nk_{\mathrm{B}}T}\left\langle \sum_{i=1}^{N} \mathbf{r}_i \cdot \mathbf{F}_i \right\rangle.$$

The problem is that the force acts over an infinitely small time during which it has an infinite value. Show that for this case the virial theorem reads

$$\frac{\beta P}{\rho} = 1 + \frac{1}{N\langle \mathbf{v}^2 \rangle t} \sum_{\text{collisions}} \mathbf{v}_{ij} \cdot \mathbf{r}_{ij},$$

where the sum is over the collisions taking place within the sampling time t.

8.4 (a) Show that the Verlet algorithm can be written in the form:

$$\begin{pmatrix} p(t+h/2) \\ x(t+h) \end{pmatrix} = \begin{pmatrix} p(t-h/2) + hF[x(t)] \\ x(t) + hp(t-h/2) + h^2 F[x(t)] \end{pmatrix}.$$

(b) Find the Jacobian matrix of this map and show that the Verlet algorithm is symplectic.

8.5 Consider a time-evolution operator acting on vectors in two dimensions, which is described by the symplectic operator $\exp(tA_{\mathrm{D}})$:

$$z(t) = \exp(tA_{\mathrm{D}})z(0),$$
$$z = (p, x) = (z_1, z_2).$$

(a) Show that symplecticity implies that

$$\frac{\partial A_1}{\partial p} = -\frac{\partial A_2}{\partial x}.$$

(b) Find a necessary condition to write A_D as $J\nabla_z H_D$. Show that this condition is equivalent to that found in (a).

(c) Show that H_D is a conserved quantity.

8.6 In this problem we consider Andersen's method for keeping the temperature constant during a MD simulation. In particular we want to find the momentum refresh rate R for which the method mimics wall collisions best. The refresh rate is defined such that the average number of velocity updates during a time Δt is equal to $R\Delta t$. Suppose the wall of the system is at temperature T, but the system itself is at a temperature $T + \Delta T$.

(a) Show that the rate at which heat is absorbed by the system is given by

$$\frac{\Delta Q}{\Delta t} \sim \kappa V^{1/3} \Delta T,$$

where κ is the thermal conductivity, defined by $\nabla T = \kappa \mathbf{j}$, where $\mathbf{j}$ is the heat flowing through a unit area per unit time.

(b) Show that the rate at which heat is transferred to a system without walls in Andersen's method is equal to

$$\frac{\Delta Q}{\Delta t} \sim RNk_B \Delta T.$$

(c) Derive from the two equations obtained the optimal rate:

$$R_{\text{opt}} \sim \frac{\kappa}{n^{1/3} k_B N^{2/3}}$$

where $n = N/V$.

8.7 [C] In this problem we consider a program for simulating nitrogen molecules in microcanonical MD using the method of constraints. The equations of motion are given in Section 8.6.2 (Eqs. (8.124)). The Lagrange parameters λ occuring in these equations are determined by requiring the constraint to be satisfied by the positions as predicted in the Verlet algorithm. These positions are given in the form

$$\mathbf{r}_i(t + h) = \mathbf{a}_i + \mathbf{b}_i \lambda.$$

The list of particles is grouped into pairs of atoms forming one nitrogen molecule: atoms $2l - 1$ and $2l$ belong to the same molecule. The integration is carried out in a loop over the pairs l – each pair has its own Lagrange parameter λ_l. For reasonable time step sizes the roots λ_l of the constraint equation are real. The smallest of these (in absolute value) is to be chosen. The forces can be calculated as usual, taking only interactions between atoms belonging to different molecules into account. Parameters for the Lennard–Jones interaction are $\varepsilon = 37.3\,\text{K}$, $\sigma = 3.31\,\text{Å}$ and $d = 0.3296\sigma$.

Periodic boundary conditions are implemented with respect to the centre of mass of the molecules. If a molecule leaves the system cell it is translated back into it (as a whole) according to PBC. Note that determining the momentum from the positions

at $t + h$ and $t - h$ after such a translation can cause severe errors: this should be done *before* moving the molecule back into the cell.

(a) Implement this algorithm for liquid nitrogen.

The program can be checked by verifying whether the kinetic energies associated with translational and vibrational degrees of freedom satisfy equipartition. The total kinetic energy K_{tot} can be determined as in the argon case by taking all *atomic* velocities into account. From this, the temperature can be determined as $Nk_{\text{B}}T = 4/5K_{\text{tot}}$ where N is the number of molecules. The translational kinetic energy K_{trans} can be calculated by taking into account the molecular velocities (sums of velocities of the two atoms) and the temperature can be found from this as $Nk_{\text{B}}T = 3/2K_{\text{trans}}$. The average temperatures should be the same for both procedures.

Check whether this requirement is satisfied.

(b) The virial theorem applies as usual: molecular forces should be used and the separation occurring in this theorem is the separation between the centres of mass of the molecules. The correction term is evaluated using $g \equiv 1$ for the correlation function beyond the cut-off distance, where it is assumed that g is independent of distance but also of the angular configuration of the molecular pairs.

(c) Calculate the pressure also using the atomic forces (including the constraint forces), and compare the result with (b).

(d) Calculate the pressure for various temperatures and densities. Cheung and Powles give extensive data on thermodynamic quantities [46]. The table below gives some of the data (in reduced units) obtained by Cheung and Powles.

ρ	T	P	U
0.6964	2.86	8.35	−17.16
0.6964	1.72	1.29	−18.68
0.6220	2.70	2.50	−15.82
0.6220	2.17	0.27	−16.30

8.8 [C] In this problem, we consider the implementation of the Andersen method for simulating a system in the canonical ensemble. Remember that the preferred energy estimator for the Verlet/leap-frog algorithm is

$$E = \sum_i \frac{[\mathbf{p}_i(t + h/2) + \mathbf{p}_i(t - h/2)]^2}{8} + V[R(t)],$$

where R is the combined position coordinate of the system which consists of particles of mass $m = 1$. In view of the form of this estimator, it seems sensible to update the momenta at the same time instances for which we calculate the positions, and it is convenient to define the ith component of the momentum coordinate at time t:

$$\mathbf{p}_i(t) = [\mathbf{p}_i(t + h/2) + \mathbf{p}_i(t - h/2)]/2.$$

(a) Using the leap-frog/Verlet algorithm, show that

$$\mathbf{p}_i(t + h/2) = \mathbf{p}_i(t) + hF_i/2.$$

The refreshed momenta $\mathbf{p}_i(t)$ are drawn from a Maxwell–Boltzmann distribution, and the momenta at time $t + h/2$, which are needed in the Verlet/leap-frog algorithm are then calculated using this last formula.

(b) Implement the Andersen update algorithm for argon and compare the results with the microcanonical program.

(c) Now suppose that the momenta are refreshed at *every* step. Show that in that case we have

$$r_i(t + h) = r_i(t) + h^2F_i/2 + h\zeta_i(t),$$

where $\zeta_i(t)$ is the ith random momentum component generated according to the Maxwell–Boltzmann distribution. This is a kind of Langevin equation. Discuss the difference with the Langevin equation described in Section 8.8.

8.9 [C] In this problem we consider the implementation of the Nosé–Hoover thermostat in the microcanonical MD simulation for Lennard–Jones argon described in Section 8.3. The extension is straightforward – the equations are given in Section 8.5.1. You can verify now that the behaviour of the Nosé–Hoover thermostat is often nonergodic. For $T = 1.5$ and $\rho = 0.8$ the behaviour is as it should be for coupling constant $Q = 1$. You can check that the standard deviation in the temperature is in accordance with Eq. (8.81). For lower temperatures, like $T = 0.85$, $\rho = 1.067$, the temperature exhibits large oscillations. The period of these oscillations depends on Q [26].

8.10 (a) Verify that when we take $g = 3N$ instead of $g = 3N + 1$ in the derivation of the Nosé–Hoover thermostat, the probability density for configurations (P, R) turns out to be:

$$\rho(P, R) = \frac{1}{3N}\left(\frac{2\pi Q}{k_\mathrm{B}T}\right)^{1/2} \exp\left[\frac{-\mathcal{H}_0(P, R)(3N + 1)}{3Nk_\mathrm{B}T}\right].$$

(b) For this choice, verify that quantities sampled in a simulation yield averages as given in Eq. (8.94).

8.11 [C] In this problem, a code for evaluating the potential felt by the particles in a two-dimensional Coulomb (or gravitational) system is developed, using the tree-code method of Section 8.7.2.

Although experienced programmers would be tempted to start building tree structures using pointers and recursive programming for this problem, it can be dealt with using more pedestrian methods. The point is that the squares can be coded by two integers *NX, NY* which are considered as bit-strings. The first of these contains information about the x-coordinate of the square and the second about the y-coordinate. They are ordered linearly: the leftmost column of squares has $NX = 0$, the rightmost column $NX = 2^n - 1$ etc., and a similar coding is adopted for the rows. If squares are neighbours, their respective *NX* and *NY*-codes should differ at most by 1 (and they should not be equal). The codes of the parents can be found

simply by shifting the bits of NX and NY one position to the right (least significant direction) and it can therefore easily be checked in the program whether the parents of the squares under consideration are neighbours or not.

The calculation of the multipole moments in each box (Eq. (8.140)) is best done in a loop over the particles, recording its contribution to all the multipole coefficients of the to which square it belongs. Also, the calculation of the interactions (Eq. (8.139)) can be done in a loop over the particles, by executing for each particle a loop over the interaction list of the square to which it belongs.

Proceeding this way, it is not necessary to keep for each square a list of the particles belonging to it. However, at the finest level, the interactions between particles within the same square and between particles in neighbouring boxes must be calculated directly so only for the last step do we need such a list for each square. If you want to economise on memory, you might create a linked list for each square containing the indices of the particles in it, but for a test you may use static arrays.

Compare the results for the tree code with those of a direct calculation, varying the number of terms in the multipole expansion.

8.12 [C] In this problem we consider a simulation of a methane molecule using the Langevin approach. Methane consists of a carbon atom sitting at the centre of a tetrahedron whose vertices are occupied by four H atoms. The C–H bond has a preferred interatomic distance of $2.104a_0$. The stretch-potential associated with the bond length varies as

$$V_{\text{stretch}} = \tfrac{1}{2}\kappa(l - l_0)^2; \quad l_0 = 2.104a_0.$$

The force constant κ has the value $\kappa = 0.30$ (in atomic units). This force acts on both the carbon and the hydrogen atoms and is directed along the C–H bond.

The preferred H–C–H angle is 109° and the potential for this bending angle is

$$V_{\text{bend}} = -\lambda \cos(\varphi - \varphi_0)^2; \quad \varphi_0 = 109°,$$

with a force constant $\lambda = 0.74$. This force lies in the H–C–H plane, and acts on the two H atoms and on the C atom. The forces on the H-atoms are perpendicular to the C–H bonds, and the bending force on the C atom is directed along the bisecting line of the H–C–H angle.

These two 'force fields', bending and stretching, specify the force on each of the atoms. To find the forces, given the position $\mathbf{r}_C$ of the carbon atom and the four positions $\mathbf{r}_H$ of the hydrogen atoms, you calculate first the forces on the hydrogen atoms only. The stretch forces can easily be found by calculating the vector $\mathbf{r}_{CH} = \mathbf{r}_H - \mathbf{r}_C$. The bending force is slightly more difficult. Denoting the two hydrogen atoms of a H–C–H chain as H1 and H2, calulate $\mathbf{r}_{CH1}$ and $\mathbf{r}_{CH1}$. Then calculate the dot product between these two vectors. From this, the cosine of the bending angle can be found. Moreover, the direction of the force can be found from the cross-product of $\mathbf{r}_{CH1}$ and $\mathbf{r}_{CH1}$: the bending force on H1 is then perpendicular to this cross product *and* to the vector $\mathbf{r}_{CH1}$, and similarly for H2. Knowing the forces on the hydrogen atoms, you can calculate their sum. The force on the carbon atom is

then simply the opposite of this, as the sum of all the interparticle forces adds up to zero.

(a) [C] Write routines for calculating the forces on the atoms and use these in an ordinary (microcanonical) MD simulation of the atom. To check the program, you can put the H-atoms on the vertices of a tetrahedron with the C-atom in the centre. If you release the molecule from this conformation with a CH-distance slightly smaller or larger than the equilibrium distance of $2.104a_0$, the molecule should stretch and contract isotropically in an oscillatory fashion.

(b) [C] Keep the temperature of the molecule constant by rescaling the velocities after each time step. Determine the average total energy of the molecule.

(c) [C] Add a Langevin thermostat to the simulation, for example by rescaling the velocities after every time step. Use the algorithm given in the last section for solving the equations of motion with friction. Add a Langevin random force, drawn from a Gaussian distribution with a width

$$\sigma^2 = q/h$$

to the interparticle force. Check that the temperature is given by

$$T = 1/(2\gamma).$$

The temperature is determined from the kinetic energy – we have

$$T = \frac{15}{2} k_B T.$$

Determine the average total energy and compare the result with the program of (b).

References

[1] D. W. Heermann, *Computer Simulations in Statistical Physics*. Heidelberg, Springer, 1986.

[2] M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*. Oxford, Oxford University Press, 1989.

[3] W. G. Hoover, *Computational Statistical Mechanics*. Amsterdam, Elsevier, 1991.

[4] D. C. Rapaport, *The Art of Molecular Dynamics Simulation*. New York, Cambridge University Press, 1996.

[5] G. Ciccotti and W. G. Hoover, eds., *Molecular Dynamics Simulation of Statistical-Mechanical Systems Proceedings of the International School of Physics "Enrico Fermi", Varenna 1985*, vol. 97. Amsterdam, North-Holland, 1986.

[6] T. Kihara and S. Koba, 'Crystal structures and intermolecular forces of rare gases,' *J. Phys. Soc. Jpn*, **7** (1952), 348–54.

[7] W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson, 'A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: application to small water clusters,' *J. Chem. Phys.*, **76** (1982), 637–49.

[8] L. Verlet, 'Computer 'experiments' on classical fluids. I. Thermodynamical properties of Lennard–Jones molecules,' *Phys. Rev.*, **159** (1967), 98–103.

[9] S. D. Stoddard and J. Ford, 'Numerical experiments on the stochastic behavior of a Lennard–Jones gas system,' *Phys. Rev. A*, **8** (1973), 1504–12.

[10] J. G. Powles, W. A. B. Evans, and N. Quirke, 'Non-destructive molecular dynamics simulation of the chemical potential of a fluid,' *Mol. Phys.*, **38** (1982), 1347–70.
[11] J. P. Hansen and I. R. McDonald, *Theory of Simple Liquids* 2nd edn. New York, Academic Press, 1986.
[12] L. Verlet, 'Computer 'experiments' on classical fluids. II. Equilibrium correlation functions,' *Phys. Rev.*, **165** (1968), 201–14.
[13] A. Rahman, 'Correlations in the motion of atoms in liquid argon,' *Phys. Rev.*, **136A** (1964), 405–11.
[14] B. J. Alder and T. E. Wainwright, 'Phase transition for a hard sphere system,' *J. Chem. Phys.*, **27** (1957), 1208–9.
[15] S. M. Thompson, 'Use of neighbour lists in molecular dynamics,' *CCP5 Quarterly*, **8** (1983), 20–8.
[16] H. J. C. Berendsen and W. F. van Gunsteren, 'Practical algorithms for molecular dynamics simulations,' in *Molecular Dynamics Simulation of Statistical Mechanical Systems* (G. Ciccotti and W. G. Hoover, eds.), *Proceedings of the International School of Physics "Enrico Fermi", Varenna 1985*, vol. 97. Amsterdam, North-Holland, 1986, pp. 43–65.
[17] J. M. Sanz-Serna, 'Symplectic integrators for Hamiltonian problems: an overview,' *Acta Numerica*, **1** (1992), 243–86.
[18] K. Feng and M.-Z. Qin, 'Hamiltonian algorithms for Hamiltonian systems and a comparative numerical study,' *Comput. Phys. Commun.*, **65** (1991), 173–87.
[19] R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles*, 2nd edn. Bristol, Institute of Physics Publishing, 1988.
[20] D. I. Okunbor and R. D. Skeel, 'Explicit canonical methods for Hamiltonian systems,' *Math. Comput.*, **59** (1992), 439–55.
[21] H. Yoshida, 'Construction of higher order symplectic integrators,' *Phys. Lett. A*, **150** (1990), 262–8.
[22] E. Forest, 'Sixth-order Lie group integrators,' *J. Comp. Phys.*, **99** (1992), 209–13.
[23] C. W. Gear, 'The numerical integration of ordinary differential equations of various orders,' report ANL7126, Argonne National Laboratory, 1966.
[24] C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations*. Englewood Cliffs, NJ, Prentice-Hall, 1971.
[25] D. I. Okunbor, 'Energy conserving, Liouville and symplectic integrators,' *J. Comput. Phys.*, **120** (1995), 375–8.
[26] B. L. Holian, A. F. Voter, and R. Ravelo, 'Thermostatted molecular dynamics: How to avoid the Toda demon hidden in Nosé–Hoover dynamics,' *Phys. Rev. E*, **52** (1995), 2338–47.
[27] R. D. Ruth, 'A canonical integration technique,' *IEEE Trans. Nucl. Sci.*, **30** (1983), 2669–71.
[28] K. Feng, 'On difference schemes and symplectic geometry,' in *Beijing Symposium on Differential Geometry and Differential Equations: Computation of Partial Differential Equations* (K. Feng, ed.). Beijing, Science Press, 1985, p. 45.
[29] J. D. Meiss, 'Symplectic maps, variational principles and transport,' *Rev. Mod. Phys.*, **64** (1992), 795–848.
[30] C. Cohen-Tannoudji, B. Diu, and F. Laloë, *Quantum Mechanics*, vols. 1 and 2. New York/Paris, John Wiley/Hermann, 1977.
[31] A. Auerbach and S. P. Friedman, 'Long-time behaviour of numerically computed orbits: small and intermediate timestep analysis of one-dimensional systems,' *J. Comp. Phys.*, **93** (1991), 189–223.
[32] H. C. Andersen, 'Molecular dynamics at constant temperature and/or pressure,' *J. Chem. Phys.*, **72** (1980), 2384–94.
[33] J. M. Haile and S. Gupta, 'Extensions of molecular dynamics simulation method. II. Isothermal systems,' *J. Chem. Phys.*, **79** (1983), 3067–76.

[34] S. Nosé, 'A unified formulation of constant temperature molecular-dynamics methods,' *J. Chem. Phys.*, **81** (1984), 511–19.
[35] W. G. Hoover, A. J. C. Ladd, R. B. Hickman, and B. L. Holian, 'Bulk viscosity via nonequilibrium and equilibrium molecular dynamics,' *Phys. Rev. A*, **21** (1980), 1756–60.
[36] H. J. C. Berendsen, J. P. M. Postma, W. F. van Gunsteren, A. Dinola, and B. Haak, 'Molecular dynamics with coupling to an external bath,' *J. Chem. Phys.*, **81** (1984), 3684–90.
[37] W. G. Hoover, 'Canonical dynamics: equilibrium phase-space distributions,' *Phys. Rev. A*, **31** (1985), 1695–7.
[38] K. Cho and J. D. Joannopoulos, 'Ergodicity and dynamical properties of constant-temperature molecular dynamics,' *Phys. Rev. A*, **45** (1992), 7089–97.
[39] B. L. Holian, 'Simulations of vibrational relaxation in dense molecular fluids,' in *Molecular Dynamics Simulation of Statistical Mechanical systems* (G. Ciccotti and W. G. Hoover, eds.), *Proceedings of the International School of Physics "Enrico Fermi", Varenna 1985*, vol. 97. Amsterdam, North-Holland, 1986, pp. 241–59.
[40] B. L. Holian, 'Simulations of vibrational-relaxation in dense molecular fluids,' *J. Chem. Phys.*, **84** (1986), 3138–46.
[41] G. J. Martyna, M. L. Klein, and M. Tuckerman, 'Nosé–Hoover chains – the canonical ensemble via continuous dynamics,' *J. Chem. Phys.*, **97** (1992), 2635–43.
[42] S. Nosé, 'A molecular dynamics method for simulations in the canonical ensemble,' *Mol. Phys.*, **52** (1984), 255–68.
[43] R. G. Winkler, V. Kraus, and P. Reineker, 'Time reversible and phase-space conserving molecular dynamics at constant temperature,' *J. Chem. Phys.*, **102** (1995), 9018–25.
[44] W. G. Hoover, 'Constant pressure equations of motion,' *Phys. Rev. A*, **34** (1986), 2499–500.
[45] M. Parrinello and A. Rahman, 'Polymorphic transitions in single crystals: a new molecular dynamics method.,' *J. Appl. Phys.*, **52** (1981), 7182–90.
[46] P. S. Y. Cheung and J. G. Powles, 'The properties of liquid nitrogen. IV. A computer simulation,' *Mol. Phys.*, **30** (1975), 921–49.
[47] D. Fincham, 'More on rotational motion of linear molecules,' *CCP5 Quarterly*, **12** (1984), 47–8.
[48] D. J. Evans, 'On the representation of orientation space,' *Mol. Phys.*, **34** (1977), 317–25.
[49] D. J. Evans and S. Murad, 'Singularity-free algorithm for molecular dynamics simulation of rigid polyatomics,' *Mol. Phys.*, **34** (1977), 327–31.
[50] H. Goldstein, *Classical Mechanics*. Reading, Addison-Wesley, 1980.
[51] G. Ciccotti, M. Ferrario, and J. P. Ryckaert, 'Molecular dynamics of rigid systems in cartesian coordinates,' *Mol. Phys.*, **47** (1982), 1253–64.
[52] J. P. Ryckaert, 'The method of constraints in molecular dynamics,' in *Molecular Dynamics Simulation of Statistical Mechanical Systems* (G. Ciccotti and W. G. Hoover, eds.), *Proceedings of the International School of Physics "Enrico Fermi", Varenna 1985*, vol. 97. Amsterdam, North-Holland, 1986, pp. 329–40.
[53] J. P. Ryckaert, G. Ciccotti, and H. J. C. Berendsen, 'Numerical integration of the cartesian equations of motion of a system with constraints: molecular dynamics of n-alkanes,' *J. Comput. Phys.*, **23** (1977), 327–41.
[54] J. P. Ryckaert, 'Special geometrical constraints in the molecular dynamics of chain molecules,' *Mol. Phys.*, **55** (1985), 549–56.
[55] J.-P. Hansen, 'Molecular dynamics simulation of Coulomb systems,' in *Molecular Dynamics Simulation of Statistical Mechanical Systems* (G. Ciccotti and W. G. Hoover, eds.), *Proceedings of the International School of Physics "Enrico Fermi", Varenna 1985*, vol. 97. Amsterdam, North-Holland, 1986, pp. 89–119.
[56] S. W. De Leeuw, J. W. Perram, and E. R. Smith, 'Simulation of electrostatic systems in periodic boundary conditions. I. Lattice sums and dielectric constants,' *Proc. R. Soc. London*, **A373** (1980), 27–56.

[57] J. Barnes and P. Hut, 'A hierarchical $O(N \log N)$ force-calculation algorithm,' *Nature*, **324** (1986), 446–9.
[58] L. Hernquist, 'Performance characteristics of tree codes,' *Astrophysi. J. Suppl.*, **64** (1987), 715–34.
[59] A. W. Appel, 'An efficient program for many-body simulation,' *Siam. J. Sci. Stat. Comput.*, **6** (1985), 85–103.
[60] J. G. Jernigan, 'Direct N-body simulations with a recursive center of mass reduction and regularization,' in *Dynamics of Star Clusters* (J. Goodman and P. Hut, eds.) *IAU Symposium*, vol. 113, Dordrecht, Reidel, 1985, pp. 275–84.
[61] L. van Dommelen and E. A. Rundensteiner, 'Fast, adaptive summation of point forces in the two-dimensional Poisson equation,' *Siam. J. Sci. Stat. Comput.*, **83** (1989), 286–300.
[62] L. Greengard, 'The numerical solution of the N-body problem,' *Comp. Phys.*, **4** (1990), 142–52.
[63] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*. Cambridge, MIT Press, 1988.
[64] G. Ciccotti and J. P. Ryckaert, 'Computer simulation of the generalized Brownian motion. I. The scalar case,' *Mol. Phys.*, **40** (1980), 141–9.
[65] S. Toxvaerd, 'Solution of the generalised Langevin equation,' *J. Chem. Phys.*, **82** (1985), 5658–62.
[66] F. J. Vesely and H. A. Posch, 'Correlated motion of 2 particles in a fluid. 1. Stochastic equation of motion,' *Mol. Phys.*, **64** (1988), 97–109.
[67] L. G. Nillson and J. A. Prado, 'A time-saving algorithm for generalised Langevin-dynamics simulations with arbitrary memory kernels,' *Mol. Phys.*, **71** (1990), 355–67.
[68] D. L. Ermak and J. A. McCammon, 'Brownian dynamics with hydrodynamic interactions,' *J. Chem. Phys.*, **69** (1978), 1352–60.
[69] P. J. Hoogerbrugge and J. M. V. A. Koelman, 'Simulating microscopic hydrodynamic phenomena with dissipative particle dynamics,' *Europhys. Lett.*, **19** (1992), 155–60.
[70] W. F. van Gunsteren and H. J. C. Berendsen, 'Algorithms for Brownian dynamics,' *Mol. Phys.*, **45** (1982), 637–47.
[71] R. Kubo, H. Ichimura, and T. Usui, *Statistical Mechanics, An Advanced Course*. Amsterdam, North-Holland, 1965.
[72] M. Plischke and H. Bergersen, *Equilibrium Statistical Physics*. Englewood Cliffs, NJ, Prentice-Hall, 1989.
[73] E. M. Gosling, I. R. McDonald, and K. Singer, 'On the calculation by molecular dynamics of the shear viscosity,' *Mol. Phys.*, **26** (1973), 1475–84.
[74] G. Ciccotti and G. Jacucci, 'Direct computation of dynamical response by molecular dynamics: the mobility of a charged Lennard–Jones particle,' *Phys. Rev. Lett.*, **35** (1975), 789–92.