Molecular Dynamics of Simple Systems

Programming Exercises for the Course Statistical Physics and Computer Simulation

Originally developed by

Prof. Dr. W. F. van Gunsteren

Adapted by

Prof. Dr. M. Reiher

Current version by

Prof. Dr. S. Riniker

Laboratory of Physical Chemistry, ETH Zurich

April 20, 2020

Contents

1 Introduction

This script will guide you through the programming exercises of the course Statistical Physics and Computer Simulation. You will learn to execute and understand a given computer program for numerical simulations, to program a physical or chemical problem, to test and debug your programs and to interpret the results obtained from simulations.

1.1 About the programming exercises

The exercises are based on a simple molecular dynamics program for atoms. Since you will have to make changes to the program and implement new features, basic knowledge in the C++ programming language is required.

The source code can in principle be compiled on MS Windows, Mac OS or Linux systems. The instructions given in this script are, however, for a Linux environment. But all tasks can also be solved on MS Windows or Mac OS.

The script starts with the basic theory necessary for performing molecular dynamics simulations of atomic liquids in Section 2. A more detailed discussion of the theory and the algorithms is provided by the accompanying lecture course Statistical Physics and Computer Simulation. Hence, we refer to the literature list of the lecture for more details.

Section 3 is about the usage and the structure of the MD program. There, you find detailed information about the input files and how to compile and run the program. In Section 4 example calculations are given. They are followed in Section 5 by some exercises involving only small changes to the code. In Section 6 a variety of topics are presented for new features of the MD program.

The book Computer Simulation of Liquids by Allen & Tildesley [1] is in general a good reference for many of the algorithms covered in this script.

1.2 What to do

1.2.1 Step 1: Familiarize yourself with the MD program

Make sure that you understand the structure of the program and that your source code is in a clean state, i.e., that it reproduces the results correctly. The examples and exercises in Sections 3 and 4 guide you through the program. They cover

- Structure and usage of the program.
- Checking integrity of the program.
- Reproduction simulation data from the literature.
- Studying the properties of simple liquids.

1.2.2 Step 2: Extend the capabilities of the MD program

Select one of the topics in Section 6. Before you start with the implementation, make sure that you use the correct equations and that you understand the algorithm. Then, try to answer the following questions, *before* you write new code. You may have to reformulate the equations to implement them into the program.

- Which parts of the program are to be changed?
- Which new parameters are to be read in?
- Which new quantities are to be calculated and printed?
- Which simulations, using which parameters, have to be performed in order to verify the new code?

1.2.3 Step 3: Present your results

At the end of the semester you will have to present your results to your fellow students in a short presentation (**not more** than three slides). Focus on the problems you encountered during the project and how you solved them.

- Slide 1: Provide a motivation for your extension.
  What is the benefit of your extension? Which problem has been solved?
- Slide 2: Present the key equations that you implemented.
  Which issues did you encounter during the implementation?
- Slide 3: Briefly explain how you implemented the extension.
  What are possible restrictions of your implementation? Show results that demonstrate that your extension is correct.

1.2.4 Step 4: Optional tasks

When you plan your implementation, you can think about how you could restructure the code so that it better fits your needs. Present your decisions and how you restructured the code.

It might be also helpful to have some more post-processing tools or smarter input generation tools. You can even think about implementing an easy-to-use graphical front end.

2 Computer Simulation of Simple Liquids

2.1 The Lennard-Jones interaction

The Lennard-Jones interaction between two atoms at a distance r is given by

$$V(r) = 4\varepsilon\left(\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right) \tag{1}$$

The graph of the potential is depicted in Fig. 1.

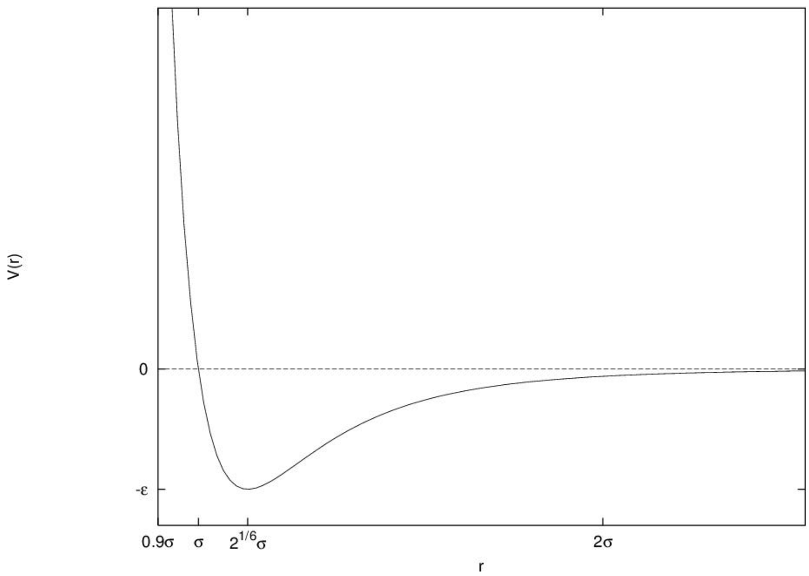


Figure 1: The Lennard-Jones interaction potential $V(r)$ as a function of the interatomic distance r.

For a system of N atoms the potential energy is then

$$E_{\text{pot}} = V(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N) = \sum_{i<j}^{N} V(r_{ij}) \tag{2}$$

where the vector $\mathbf{r}_{ij}$ is defined by the Cartesian position vectors $\mathbf{r}_i$ and $\mathbf{r}_j$ of atoms i and j:

$$\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j \tag{3}$$

The distance r_{ij} between atoms i and j is

$$r_{ij} = |\mathbf{r}_{ij}| = \left(x_{ij}^2 + y_{ij}^2 + z_{ij}^2\right)^{\frac{1}{2}} \tag{4}$$

The partial derivative of $V(r_{ij})$ with respect to $\mathbf{r}_i$ is:

$$\frac{\partial V(\mathbf{r}_{ij})}{\partial \mathbf{r}_i} = 4\varepsilon\left(-12\left(\frac{\sigma}{r_{ij}}\right)^{12} + 6\left(\frac{\sigma}{r_{ij}}\right)^{6}\right)\left(\frac{\mathbf{r}_{ij}}{r_{ij}^2}\right) \tag{5}$$

The number of atom pairs in the summation in (2) can be reduced by applying a *cut-off radius* R_c beyond which no interactions $V(r_{ij})$ are computed. Due to the r^{-6} distance dependence of $V(r)$, rather short cut-off radii of $2.5\sigma - 3.3\sigma$ can be used (Verlet, 1968).

2.2 Periodic boundary conditions

The classical way to minimize edge effects in a finite system is to apply *periodic boundary conditions*. The atoms of the system that is to be simulated are put into a cubic, or more generally into any periodically space filling shaped box. In case of a rectangular box, one box is then surrounded by 26 translated images of itself. When calculating the forces on an atom i in the central box, only the interaction with the nearest image of atom j is taken into account. The vector $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ connecting *nearest images* can be obtained from

$$\begin{aligned} x_{ij} &= x_{ij} - \mathrm{nint}\left(\frac{x_{ij}}{a}\right) a \\ y_{ij} &= y_{ij} - \mathrm{nint}\left(\frac{y_{ij}}{b}\right) b \\ z_{ij} &= z_{ij} - \mathrm{nint}\left(\frac{z_{ij}}{c}\right) c \end{aligned} \tag{6}$$

where the lengths of the edges of the periodic box are denoted by a, b, and c, and the function $\mathrm{nint}(x)$ delivers the integer number that is nearest to x. When an atom leaves the central box at one side, it enters it with identical velocity at the opposite side at the translated image position. The atom i can be kept in the central box that lies in the positive quadrant with respect to an origin at $\mathbf{r}_0$, by applying

$$\begin{aligned} x_i &= x_i - \mathrm{nint}\left(\frac{x_i - x_0 - \frac{a}{2}}{a}\right) a \\ y_i &= y_i - \mathrm{nint}\left(\frac{y_i - y_0 - \frac{b}{2}}{b}\right) b \\ z_i &= z_i - \mathrm{nint}\left(\frac{z_i - z_0 - \frac{c}{2}}{c}\right) c \end{aligned} \tag{7}$$

Application of periodic boundary conditions means that in fact a crystal is simulated. For a liquid or solution the periodicity is an artifact of the computation. Thus, one has to make sure that the effects of periodicity on the forces on the atoms are not be significant. This means that an atom should not simultaneously interact with another atom *and* its images or with its own images. Only interactions between nearest images should be evaluated, that is, the cut-off radius R_c for computation of the interaction should be smaller than half the smallest box edge:

$$R_c < \frac{1}{2}\min(a, b, c) \tag{8}$$

2.3 Newton's equations of motion

Newton's equations of motion for a system of N atoms with Cartesian position vectors $\mathbf{r}_i$ and masses m_i are:

$$\frac{\mathrm{d}^2\mathbf{r}_i(t)}{\mathrm{d}t^2} = \frac{\mathbf{f}_i(t)}{m_i} \qquad\qquad i = 1, 2, \ldots, N \tag{9}$$

where the force $\mathbf{f}_i(t)$ on atom i is equal to the negative gradient of the potential energy function (2) with respect to the coordinates of atom i

$$\mathbf{f}_i(t) = -\frac{\partial}{\partial \mathbf{r}_i} V(\mathbf{r}_1(t), \mathbf{r}_2(t), \ldots, \mathbf{r}_N(t)) \tag{10}$$

These equations can be numerically integrated using small ($\ll$ than the shortest oscillation period in the system) MD time steps Δt producing a *trajectory* (atomic positions as a function of time t) of the system. Equations (9) and (10) conserve the total energy of the system

$$E_{\text{tot}} = E_{\text{pot}} + E_{\text{kin}} \tag{11}$$

and the total translational momentum of the system

$$\mathbf{p}_{\text{tot}} = \sum_{i=1}^{N} m_i \mathbf{v}_i \tag{12}$$

where the total kinetic energy of the system is given by

$$E_{\text{kin}} = \sum_{i=1}^{N} \frac{1}{2} m_i \mathbf{v}_i^2 \tag{13}$$

Monitoring E_{tot} and $\mathbf{p}_{\text{tot}}$ (or $|\mathbf{p}_{\text{tot}}|$) is very useful in order to check the conservation of these quantities and thus the integrity of a MD program.

2.4 The leap-frog integration scheme

A simple algorithm for the integration of (9) can be obtained by writing a Taylor expansion of the velocity $\mathbf{v}_i(t)$ at a time point t_n for $t + \frac{\Delta t}{2}$ and one for $t - \frac{\Delta t}{2}$

$$\mathbf{v}_i\left(t + \frac{\Delta t}{2}\right) = \mathbf{v}_i(t_n) + \left.\frac{\mathrm{d}\mathbf{v}_i}{\mathrm{d}t}\right|_{t_n} \frac{\Delta t}{2} + \left.\frac{\mathrm{d}^2\mathbf{v}_i}{\mathrm{d}t^2}\right|_{t_n} \frac{\left(\frac{\Delta t}{2}\right)^2}{2!} + \ldots \tag{14}$$

$$\mathbf{v}_i\left(t - \frac{\Delta t}{2}\right) = \mathbf{v}_i(t_n) - \left.\frac{\mathrm{d}\mathbf{v}_i}{\mathrm{d}t}\right|_{t_n} \frac{\Delta t}{2} + \left.\frac{\mathrm{d}^2\mathbf{v}_i}{\mathrm{d}t^2}\right|_{t_n} \frac{\left(\frac{\Delta t}{2}\right)^2}{2!} - \ldots \tag{15}$$

Subtracting (15) from (14), rearranging the terms, and employing

$$\frac{\mathrm{d}\mathbf{v}_i(t)}{\mathrm{d}t} = \frac{\mathbf{f}_i(t)}{m_i} \tag{16}$$

one obtains

$$\mathbf{v}_i\left(t_n + \frac{\Delta t}{2}\right) = \mathbf{v}_i\left(t_n - \frac{\Delta t}{2}\right) + \frac{\mathbf{f}_i(t_n)}{m_i}\Delta t \tag{17}$$

By an analoguous procedure using a Taylor expansion for the position $\mathbf{r}_i(t)$ at time $t_n + \frac{\Delta t}{2}$ (forward: $\frac{\Delta t}{2}$, backward: $-\frac{\Delta t}{2}$) one obtains

$$\mathbf{r}_i(t_n + \Delta t) = \mathbf{r}_i(t_n) + \mathbf{v}_i\left(t_n + \frac{\Delta t}{2}\right)\Delta t \tag{18}$$

Equations (17) and (18) are called the *leap-frog scheme* for integration of the classical equations of motion (9).

2.5 Coupling to a temperature bath

When solving the classical equations of motion (9) and (10) the total energy E_{tot} is a constant of motion, and the volume V is constant too, yielding a microcanonical ensemble. For various reasons this is not very convenient and many approaches have appeared in the literature to yield a type of dynamics in which temperature is the independent variable rather than a derived property. A simple way to control the temperature T of the system is to couple it to a heat or temperature bath of reference temperature T_0. The basic idea [4] is to modify the equations of motion (9) such that the net result on the system is a first-order relaxation of temperature T to a given reference value T_0:

$$\frac{\mathrm{d}T(t)}{\mathrm{d}t} = \frac{T_0 - T(t)}{\tau_{\mathrm{T}}} \tag{19}$$

The modified equations of motion are

$$\frac{d\mathbf{v}_i(t)}{dt} = \frac{\mathbf{f}_i(t)}{m_i} - \frac{c_{\mathrm{v}}^{\mathrm{df}}}{k_{\mathrm{B}}\tau_{\mathrm{T}}}\left(\frac{T_0}{T(t)} - 1\right)\mathbf{v}_i \tag{20}$$

where the heat capacity per degree of freedom (number is N_{df}) of the system, $c_{\mathrm{v}}^{\mathrm{df}}$, is approximated using

$$\Delta E_{\mathrm{kin}} = N_{\mathrm{df}} c_{\mathrm{v}}^{\mathrm{df}} \Delta T \tag{21}$$

Here we neglect the contribution of the change in E_{pot}, ΔE_{pot}, to the total energy change ΔE_{tot}, which should be used in (21) to define the heat capacity. The temperature relaxation time or coupling time τ_{T} controls the strength of the coupling to the heat bath. This temperature coupling can be easily incorporated into the leap-frog scheme. For a system of N atoms the temperature can be obtained from the velocities by the relation (equipartition)

$$\frac{3N}{2} k_{\mathrm{B}} T(t) = E_{\mathrm{kin}}(t) = \sum_{i=1}^{N} \frac{1}{2} m_i \mathbf{v}_i^2(t) \tag{22}$$

where k_{B} is Boltzmann's constant. If the system's energies are evaluated on a per mole (mol^{-1}) basis, one should use R, the gas constant, in (22) instead of k_{B}. After applying (17) the obtained velocities $\mathbf{v}_i\left(t_n + \frac{\Delta t}{2}\right)$ are to be scaled by a factor

$$\lambda\left(t_n + \frac{\Delta t}{2}\right) = \left(1 + \frac{2c_{\mathrm{v}}^{\mathrm{df}}}{k_{\mathrm{B}}}\frac{\Delta t}{\tau_{\mathrm{T}}}\left(\frac{T_0}{T\left(t - \frac{\Delta t}{2}\right)} - 1\right)\right)^{\frac{1}{2}} \tag{23}$$

where the temperature at the previous time point $t_n - \frac{\Delta t}{2}$ is used since the temperature $T\left(t_n + \frac{\Delta t}{2}\right)$ is still to be calculated from the velocities $\mathbf{v}_i\left(t_n + \frac{\Delta t}{2}\right)$ obtained after multiplication by λ. The coupling constant τ_{T} can be arbitrarily chosen, but should exceed 10 time steps Δt to ensure stability of the algorithm. The equations of motion (20) no longer conserve the total energy E_{tot} or the total momentum $\mathbf{p}_{\mathrm{tot}}$ of the system, as when solving (9) and (10).

2.6 Pressure and the virial

The pressure of a system in equilibrium is given by

$$P = \frac{2}{3V}\left(\langle E_{kin}\rangle - \langle \Xi \rangle\right) \tag{24}$$

where the volume of the sytem is denoted by V, an ensemble (or trajectory) average is denoted by $\langle \ldots \rangle$ and the virial Ξ is defined by

$$\Xi = -\frac{1}{2}\sum_{i<j}^{N} \mathbf{r}_{ij} \cdot \mathbf{f}_{ij} \tag{25}$$

where $\mathbf{f}_{ij}$ is the force on atom i exerted by atom j, and the dot indicates a scalar product [12].

2.7 Center of mass motion

It is often useful to monitor the motion of the center of mass of a system since the translational momentum should be a constant of motion when Newton's equations of motion (9) and (10) are integrated. The position of the center of mass of the system is defined as

$$\mathbf{r}_{\mathrm{cm}} = \frac{1}{m_{\mathrm{tot}}} \sum_{i=1}^{N} m_i \mathbf{r}_i \tag{26}$$

and its velocity as

$$\mathbf{v}_{\mathrm{cm}} = \frac{1}{m_{\mathrm{tot}}} \sum_{i=1}^{N} m_i \mathbf{v}_i \tag{27}$$

with

$$m_{\mathrm{tot}} = \sum_{i=1}^{N} m_i \tag{28}$$

2.8 Gaussian or Maxwellian distributions

The atomic velocity $\mathbf{v}_i$ of a system in equilibrium will follow a *Maxwellian distribution* [15], that is, the probability that the velocity lies between $\mathbf{v}_i$ and $\mathbf{v}_i + \mathbf{dv}_i$ is

$$p(\mathbf{v}_i)\mathrm{d}\mathbf{v}_i = \left(\frac{2\pi k_{\mathrm{B}} T}{m_i}\right)^{-\frac{3}{2}} \exp\left(\frac{-m_i \mathbf{v}_i^2}{2 k_{\mathrm{B}} T}\right) \mathrm{d}\mathbf{v}_i \tag{29}$$

where k_{B} is Boltzmann's constant, T the temperature and m_i the mass of an atom. Mathematically this velocity distribution has the form of a product of three *Gaussian distributions*

$$p(x) = \left(2\pi\sigma^2\right)^{-\frac{1}{2}} \exp\left(-\frac{(x - \langle x \rangle)^2}{2\sigma^2}\right) \tag{30}$$

When starting a MD simulation the atomic velocities are often sampled from a Maxwellian distribution (29) at the desired temperature T. Technically, sampling from a Gaussian distribution is performed by taking the sum of a series of random numbers taken from a uniform distribution. This procedure is based on the *central limit theorem* which states that in the limit of an infinite sum of independent stochastic variables obeying an arbitrary (so also uniform) distribution, the sum is a stochastic variable obeying a *Gaussian* distribution.

2.9 The radial distribution function $g(r)$

The *radial distribution function* $g(r)$ is defined such that the quantity $\rho g(r)\mathrm{d}r$ is the "probability" of observing a second atom in the spherical shell between r and $r+\mathrm{d}r$ given that there is an atom at the origin of $\mathbf{r}$. The quantity

$$\rho = \frac{N}{V} \tag{31}$$

is the number density. This "probability" is not normalized to unity, but we have instead

$$\int_0^\infty \rho g(r) 4\pi r^2 \mathrm{d}r = N - 1 \tag{32}$$

In fact, (32) shows that $\rho g(r) 4\pi r^2 \mathrm{d}r$ is really the number of atoms between r and $r + \mathrm{d}r$ about a central atom. The function $g(r)$ can be thought of as the factor that multiplies the bulk density ρ to give a local density $\rho(r) = \rho g(r)$ about some fixed atom. Of course $g(r) \to 0$ when $r \to 0$, since atoms can not completely overlap. When r becomes large, the central atom "sees" a uniform distribution, so when $r \to \infty$ then $g(r) \to 1$. The radial distribution function contains structural information on the liquid, and can be determined from X-ray or neutron diffraction experiments.

2.10 Units

Different sets of units are used in MD simulations. In general the use of *Standard International (SI)* units is recommended. In MD simulation of model systems, like Lennard-Jones liquids, people used to work with dimensionless quantities *(reduced units)* and apply the appropriate scaling afterward.

When choosing the SI system for molecular systems it is recommended to use the following basic units:

length:	r:	nm	$= 10^{-9}$ m
mass:	m:	u	= atomic mass unit
			= 1/12 of the mass of a ^{12}C atom
			$= 10^{-3}/N_\mathrm{A}$ kg
			$= 1.6605655 \cdot 10^{-27}$ kg
time:	t:	ps	$= 10^{-12}$ s
temperature:	T:	K	
charge:	q:	e	= elementary charge
			$= 1.6021892 \cdot 10^{-19}$ C

Consistent with these units are:

velocity:	v:	nm ps^{-1}	
energy:	E:	nm^2 u ps^{-2}	$\widehat{=}\ 10^3$ J mol$^-1$
			$\widehat{=}\ 1$ kJ mol^{-1}
force:	f:	nm u ps^{-2}	
pressure:	P:	nm^{-1} u ps^{-2}	$= 10^{30}$ mol$^{-1}/N_\mathrm{A}$ Pa
			$= 1.66057 \cdot 10^6$ Pa
			$= 16.6057$ bar

Here we have used:

$$\begin{array}{lll}
N_\mathrm{A} & = \text{Avogadro's number} & = 6.022045 \cdot 10^{23}\ \mathrm{mol}^{-1} \\
R & = \text{gas constant} & = 8.31441 \cdot 10^{-3}\ \mathrm{kJ\ mol}^{-1}\ \mathrm{K}^{-1} \\
k_\mathrm{B} & = \text{Boltzmann's constant} & = R/N_\mathrm{A} = 1.380662 \cdot 10^{-26}\ \mathrm{kJ\ K}^{-1}
\end{array}$$

3 About the MD Program

3.1 Compiling the program

Download the tar file `mdatom.tar` from the webpage for the exercises. Unpack it (e.g. on Linux with `tar xvf mdatom.tar`). This will create a new subdirectory `mdatom`.

This directory contains the program sources in `source_code`, inputs for some of the programming exercises described below in `input_files` and some useful scripts in `helper_scripts`.

The build environment is CMake. CMake allows you to generate the necessary files for the operating system and IDE of your choice. For instance, on MS Windows you can use Visual Studio or on Linux systems the Make environment. On Linux systems change into the `source_code` directory and create a directory `build`. In `build` issue the command `cmake ..` followed by `make` to compile the program. For more details on CMake see its documentation on www.cmake.org.

3.2 Program usage

`<executable> <parameter file>` or
`<executable> <parameter file> <initial coords file>`

The program expects one or two input files. The first (called `params.inp` in the example section) is always necessary as it contains the parameters controlling the MD simulation (for details see section 3.3). The second (called `coords.inp` in the example section) contains the initial coordinates of the atoms (if `XVInitialization` > 0) and initial velocities (if `XVInitialization` > 2).

The program writes output to the console which can be redirected into a file if necessary. In addition, the program generates a file called `coords.final` with the atomic coordinates and velocities at the final time point. This file can be used as the initial coordinates file for restarting a MD run. If `TrajectoryOutput` is 1, a file called `coords.traj` is created containing the atomic trajectories.

3.3 Input parameters

The input parameters for the program have to specified in a file passed to the program as the first command line option. The following example shows the format of the parameter file:

```
# Title
Sample input file
#                NumberAtoms                  AtomicMass                     MDType
                        1000                           1                          0
#                 BoxSize(x)                  BoxSize(y)                 BoxSize(z)
                      10.436                      10.436                     10.436
#              NumberMDSteps                 InitialTime                   TimeStep
                        1000                           0                      0.005
#         InitialTemperature           TargetTemperature    TemperatureCouplingTime
                       1.095                       1.095                        0.1
#          BoltzmannConstant                  RandomSeed
                           1                      121988
#           XVInitialization  CoordInitializationSpread              FinalXVOutput
                           0                           0                          1
#        NAtomsOnBoxEdge(x)          NAtomsOnBoxEdge(y)         NAtomsOnBoxEdge(z)
                          10                          10                         10
#                  EpsilonLJ                     SigmaLJ    InteractionCutoffRadius
                           1                           1                        2.5
#   PropertyPrintingInterval    NumberRadialDistrPoints     RadialDistrCutoffRadius
                           1                         100                        2.5
#           TrajectoryOutput      TrajectoryOutputFormat   TrajectoryOutputInterval
                           0                           2                          5
```

The coordinate input file (second optional command line argument) has the following format. Whether velocities (VX, VY, VZ) are required depends on the input parameters.

```
Line 1                     : Title
Line 2                     : NumberAtoms
Lines 3 to (NumberAtoms+2): X  Y  Z  VX  VY  VZ
```

The parameters and the formats are described in the source code. For completeness the parameters are also listed in Tab. 1. The values for the parameters are assumed to be consistent with the units detailed in Section 2.10.

3.4 Output

The console output starts with the input parameter successfully read in. Every `PropertyPrintingInterval` time step the following data is printed:

Table 1: Input parameter of the MD program.

`NumberAtoms`	number of atoms N in the system
`AtomicMass`	atomic mass m_i (> 0)
`MDType`	controls type of MD (constant E_{tot} or constant T)
	`0` classical MD, no coupling to temperature bath
	`1` MD with temperature coupling to bath with $T =$ `TargetTemperature`
`BoxSize(x/y/z)`	lengths of the edges of the periodic box (> 0), a, b and c
`NumberMDSteps`	number of MD time steps Δt to be performed
`InitialTemperature`	initial time t_0 at which the MD simulation is started
`TimeStep`	MD time step Δt
`InitialTemperature`	if $> 10^{-6}$, defines initial velocities from from a Maxwellian distrubution
`TargetTemperature`	reference temperature T_0 of heat bath (`MDType` $= 1$)
`TemperatureCouplingTime`	coupling time constant τ_{T} for coupling to a heat bath (> 0, `MDType` $= 1$); the factor $c_{\mathrm{v}}^{\mathrm{df}}(k_{\mathrm{B}}/2)^{-1}$ is taken equal to 1.
`RandomSeed`	random number generator seed to allow for reproducibility
`XVInitialization`	controls the reading of coordinates X and velocities V from the input file
	`0` X generated on a lattice, V = 0
	`1` X from ASCII file, V = 0
	`2` X from binary file, V = 0
	`3` X and V from ASCII file
	`4` X and V from binary file
`CoordInitializationSpread`	spread of atoms placed by class `CoordinatesAndVelocitiesInitializer` around lattice points (`XVInitialization` $= 0$)
`FinalXVOutput`	control writing final X and V to `coords.final`
	`0` X and V written to binary file
	`1` X and V written to ASCII file
`NAtomsOnBoxEdge(x/y/z)`	number of atoms to be placed along each box-edge by class `CoordinatesAndVelocitiesInitializer` (> 0, `XVInitialization` $= 0$)
`EpsilonLJ`	Lennard-Jones interaction parameter ε
`SigmaLJ`	Lennard-Jones interaction parameter σ
`InteractionCutoffRadius`	cut-off radius R_{c} for evaluating the interaction ($<$ smallest of `BoxSize(x/y/z)`$/2$)
`PropertyPrintingInterval`	controls printing per MD step, every `PropertyPrintingInterval` (> 0) steps energies, etc. are printed
`NumberRadialDistrPoints`	number of points r for which the radial distribution function $g(r)$ is computed (> 0, `RadialDistCutoffRadius` > 0)
`RadialDistrCutoffRadius`	cut-off radius for computing $g(r)$, ($<$ smallest of `BoxSize(x/y/z)`$/2$)
`TrajectoryOutput`	determines whether a trajectory file is created
	`0` no trajectory file is created
	`1` the coordinates are written every `TrajectoryOutputInterval` steps to a trajectory file
`TrajectoryOutputFormat`	controls format of writing of trajectory to file
	`0` time frames are written in binary form
	`2` time frames are written in ASCII form
`TrajectoryOutputInterval`	controls writing of trajectory to `coords.traj`, every `TrajectoryOutputInterval` (> 0) steps a time frame is written to file

Step number	n
Time at MD step	t
Total energy	E_{tot}
Kinetic energy	E_{kin}
Potential energy	E_{pot}
Virial	Ξ
Pressure	P
Temperature coupling scaling factor	λ

Averages and fluctuations of a quantity E are calculated employing

$$\langle E \rangle = \frac{1}{\texttt{NumberMDSteps}} \sum_{n=1}^{\texttt{NumberMDSteps}} E(t_n) \tag{33}$$

and

$$\left\langle \Delta E^2 \right\rangle = \left\langle (E - \langle E \rangle)^2 \right\rangle = \left\langle E^2 \right\rangle - \langle E \rangle^2 \tag{34}$$

The radial distribution function $g(r)$ is computed in two stages. For each MD step, an instance of `InstantaneousRadialDistribution` is created and contains in an array `radialCount[N]` the number of ordered atom pairs separated by a distance r, where N is determined by $\mathrm{N} = r/d_{gr}+1$ and $d_{gr} = \texttt{RadialDistrCutoffRadius/NumberRadialDistrPoints}$. Then, in the MD program, all `InstantaneousRadialDistribution`s are added together in an instance of `AveragedRadialDistribution` and $g(r)$ is computed using the formula

$$g(r) = \frac{\texttt{BoxSize(x)} \cdot \texttt{BoxSize(y)} \cdot \texttt{BoxSize(z)} \cdot \texttt{radialCount[N]}}{\texttt{NumberAtoms} \cdot (\texttt{NumberAtoms} - 1) \cdot 2\pi r^2 \cdot d_{gr} \cdot \texttt{NumberMDSteps}} \tag{35}$$

and subsequently printed to the console.

3.5 Structure and classes

The entry point of the program is the main function in `main.cpp`. It creates an instance of the class `MDSimulation`, which takes care of setting up the simulation variables and launching the actual simulation. Thereby, it relies on different classes, each of which has some specific use. Each class consists of a header file (`.h`) and of an implementation file (`.cpp`). The names for classes, functions and variables were chosen to be very explicit and the code should be self-explanatory.

The central classes are the following ones:

`MDSimulation`	launches a MD simulation starting from parameters and, optionally, coordinates.
`MDParameters`	contains the parameters loaded from the input file.
`MDRunOutput`	takes care of all output-related operations.
`MDRun`	executes the MD loop according to the leap-frog scheme.
`InteractionCalculator`	performs all interaction-related operations: calculation of the forces, virial, etc.

4 Testing the MD Program

First, the program needs to be compiled. Follow the instructions given in section 3.1. In the `examples` subdirectory the input files for the six tests are provided.

4.1 Tips

As the output of the MD program is not in the most convenient form for further processing and the setup of input files may by tedious, there are some scripts (for Linux systems) that might help you.

4.1.1 Extracting energy trajectories

The energies are written to the console as a function of time. If you simply want to get energies without all the additional information printed, there is a Python script called `energy.py` in the directory `scripts`. Redirect the output of the program into a file, e.g., `mdatom [input files] > outputfile`. The command `energy.py outputfile > energiesfile` and will print you all the energies at each step into the `energiesfile` in the form:
`STEP TIME E-TOTAL E-KINETIC E-POTENTIAL VIRIAL PRESSURE SCALE-T`

4.1.2 Extracting $g(r)$

The $g(r)$ values are printed at the end a MD run to the console. You can get them in a form that is useful for graphics programs like `gnuplot` or `xmgrace` by using another Python script called `gr.py`. The command `gr.py outputfile > radialdistrfile` and will print you all pairs of distance and corresponding radial distribution into the `radialdistrfile` in the form:
`R g(R)`

4.1.3 Background jobs

If you have jobs that take some time you can put them into background by starting them with an ampersand (`&`) appended. This allows you to work on while the computer is calculating. To make sure your job does not get killed when you log off, use `nohup <..> &`.

4.2 Two atom system

4.2.1 1st Exercise

The computation of the force, potential energy and virial can be checked by applying the MD program to a two atom system. The results can also be obtained by hand, and thus easily checked.

First, make a configuration file containing two atoms at a distance $\sigma = 1.0$ nm, see file `ex1/coords.inp`. Set `NumberAtoms` = 2, `XVInitialization` = 1 and `NumberMDSteps` = 1, see file `ex1/params.inp`. Run the program and check the value of E_{pot}. Has the virial Ξ the correct value? Argue whether the pressure P should be positive or negative.

4.2.2 2nd Exercise

Second, make a configuration file containing two atoms at a distance $2^{\frac{1}{6}}\sigma$, or 1.122462 nm if $\sigma = 1$ nm, see file `ex2/coords.inp`. Set again `NumberAtoms` = 2, `XVInitialization` = 1 and `NumberMDSteps` = 1, see file `ex2/params.inp`, and run the program. Check the values of E_{pot}, Ξ and P.

4.3 Reproduction of literature data with a 1000 atom system

The literature contains many results of simulation of Lennard-Jones systems, large ones ($N = 864$) and smaller ones. As a test case the data of Verlet are to be reproduced using the MD program. The data can be found in Verlet (1968); the radial distribution function $g(r)$ is displayed in Fig. 2. There, the author employed so-called reduced units, i.e. $T^* = k_{\mathrm{B}}T/\epsilon$ and $\rho^* = \sigma^3\rho$. In the following, we will not be using reduced units, but we will set `SigmaLJ` and `EpsilonLJ` equal to 1.

4.3.1 3rd Exercise

The conditions of the MD simulation are $\rho^* = 0.880$ and $T^* = 1.095$. Take a cubic N = 1000 atom system. Compute the value of `BoxSize(x)` = `BoxSize(y)` = `BoxSize(z)` from the relation $\rho = N/(\mathtt{BoxSize(x)} \cdot \mathtt{BoxSize(y)} \cdot \mathtt{BoxSize(z)})$. Use `XVInitialization` = 0, `InitialTemperature` = 120.27 and try the MD time step `TimeStep` = 0.005. A reasonable value for the cut-off radius is `InteractionCutoffRadius`

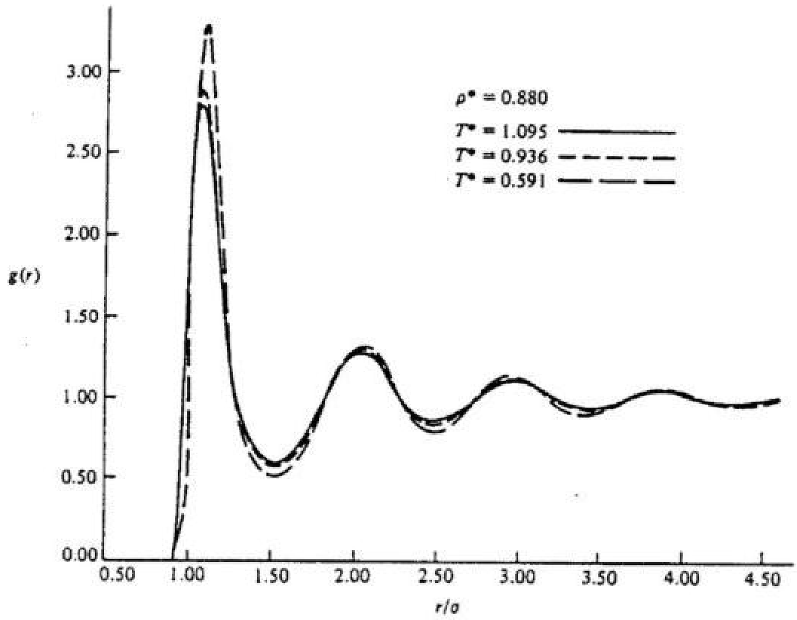


Figure 2: The radial distribution function of a fluid of molecules obeying a Lennard–Jones 6-12 potential from molecular dynamics calculations. $T^* = kT/\epsilon$ and $p^* = \sigma^3\rho$.

$= 2.5$, see file `ex3/params.inp`. Run the program for `NumberMDSteps` $= 1000$ steps and check the initial and average temperature. The conservation of the total energy can be checked by comparing the fluctuation of the total energy ΔE_{tot} with that of the kinetic energy ΔE_{kin}. The ratio should be smaller than 1:20, if `MDType` $=$ 0 was used. Check also whether the translational motion of the center of mass is conserved. Since the initial configuration is a regular lattice, the system needs time to equilibrate. Compare the $g(r)$ with that of Verlet. Discuss the difference.

4.3.2 4th Exercise

Continue the MD run from the previous exercise (start from the coordinates obtained there) for another `NumberMDSteps` $= 1000$ steps, using `XVInitialization` $= 3$ and `InitialTemperature` $= 0.0$, and set `PropertyPrintingInterval` $= 10$ in order to reduce the size of the output file, see input file `ex4/params.inp`. Compare $g(r)$ again.

4.3.3 5th Exercise

Continue the MD run for another `NumberMDSteps` = 1500 with `PropertyPrintingInterval` = 100 (see `ex5/params.inp`). The output should be compared with the previous two jobs: are the properties very different from the previous exercise? How long is the oscillation period of E_{pot} and E_{kin}? Discuss the size of ΔE_{tot}, ΔE_{pot} and ΔE_{kin} as a function of the length of the simulation.

4.3.4 6th Exercise

This example is the equivalent of job 3, but with `MDType` = 1, that is, with the system coupled to a temperature bath with `TemperatureCouplingTime` = $\tau_T^* = 0.1$ (see `ex6/params.inp`). Compare the results especially ΔE_{tot} and the center of mass motion.

4.4 Atom-atom interaction parameters

Different atoms will have different Lennard-Jones parameters describing the atom-atom interaction. Value for the noble gas atoms are given by Maitland et al. (1981):

Atom	m (a.m.u.)	ε/k_B (K)	σ (nm)
He	4.0026	10.2	0.228
Ne	20.183	47.0	0.272
Ar	39.948	119.8	0.341
Kr	83.80	164.0	0.383

4.5 Phase diagram of Lennard-Jonesium

Hansen and Weis [10] have generated the following phase diagram (Fig. 3) for Lennard-Jonesium. Under the bell-shaped dashed curve two phases (gas and liquid) coexist. The top of the bell represents the critical point. At the left of the bell the system is in the gas phase, at the right in the liquid phase. The point with discontinuous derivative represents the triple point. Two phases (liquid and solid) coexist between the two straight lines. The right-hand side of the diagram corresponds to the solid phase. Which phase is simulated in section 4.3?

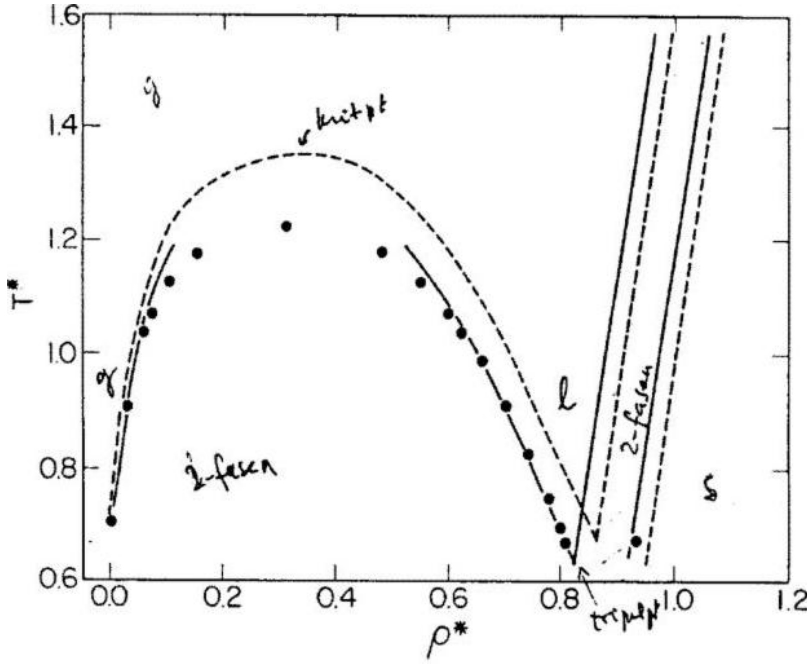


Figure 3: Phase diagram for neon. The dashed line is for the classical Lennard-Jones system and the full line includes quantum corrections. The dots are experimental data. After Hansen and Weis [10].

5 Exercises with the MD program

5.1 Effect of double versus single precision on accuracy

It is interesting to evaluate the effect of single versus double precision on the fluctuations of E_{tot} and on the conservation of the center of mass motion. To do so, change the type of the variables and function return values from `double` to `float`. Continue the constant total energy (`MDType` $= 0$) MD run of Exercise 5 in section 4.3 using a variable amount of steps (`NumberMDSteps` $= 100$, 200, 500, 1000, 2000, 5000, 10000, 20000, ...) in order to analyze the dependence of E_{tot}, E_{kin} and E_{pot} on the length of a run, and using single and double precision in order to evaluate the effect of machine precision. Plot the results as a function of simulation length. Discuss the curves. One would expect ΔE_{tot}(single) to be larger than ΔE_{tot}(double). Why? List other sources of error that will enhance ΔE_{tot}. If time permits, also test the use of the `long double` type.

5.2 Accuracy as a function of the size of the MD time step

Determine the value of $\Delta E_{\text{tot}}/\Delta E_{\text{kin}}$ as a function of the MD time step Δt. Perform MD runs for different values of `TimeStep` and `NumberMDSteps` with `MDType` = 0, `InteractionCutoffRadius` = 2.5 and `NumberAtoms` = 1000. In order to compare the fluctuations the lengths of the runs should be equal, so `NumberMDSteps` changes with changing `TimeStep`, e.g., `TimeStep` = $0.02; 0.01; 0.005; 0.002; \ldots$, `NumberMDSteps` = $1000; 2000; 4000; 10000; \ldots$. Start from an equilibrated configuration and make a graph of $\Delta E_{\text{tot}}/\Delta E_{\text{kin}}$ as a function of Δt and discuss and explain the result.

5.3 CPU-time and physical properties as a function of the number of atoms N

The dependence of the required CPU-time for a MD run on the number of atoms N can be easily determined. Run short `NumberMDSteps` = 100 jobs using $N = 125$, $N = 216$, $N = 512$, and $N = 1000$. Make a graph of CPU-time as a function of N. Which N-dependence do you expect? Discuss and explain the results.

In order to determine the dependence of physical properties, like E_{pot} and E_{kin} per atom, the pressure and radial distribution function, on the size of the simulated system, longer runs have to be performed. Start from equilibrated configurations for $N = 125$, $N = 216$, $N = 512$ and $N = 1000$ and compare the mentioned quantities for longer (`NumberMDSteps` ≥ 10000) runs. Compare the size dependency for constant volumes and constant particle densities.

5.4 CPU-time and accuracy as a function of the cut-off radius

The dependence of the required CPU-time for a MD run on the cut-off radius R_{c} can be easily determined. Run short `NumberMDSteps` = 100 jobs using $N = 1000$ (why?) and change `InteractionCutoffRadius` from 1.0 to 5.0. Which R_{c}-dependence do you expect? Discuss and explain the results.

In order to determine the accuracy, that is, the size of ΔE_{tot} as a function of R_{c}, longer MD runs (`NumberMDSteps` ≥ 1000) have to be performed. Take the size of the MD time step Δt sufficiently small, so that the value of ΔE_{tot} / ΔE_{kin} is not determined by Δt, (see section 5.2), but by R_{c}. Make a graph of the R_{c} dependence of ΔE_{tot} / ΔE_{kin}. Make also a graph of the R_{c}-dependence of ΔE_{tot} / ΔE_{pot}. Discuss and explain the results.

5.5 Effect of coupling to a temperature bath

The dependence of the fluctuations ΔE_{tot}, ΔE_{kin} and ΔE_{pot} on the value of the coupling time constant τ_{T} can be determined from relatively short (`NumberMDSteps` = 10000, `TimeStep` = 0.005) MD runs using `MDType` = 1 and various values for `TemperatureCouplingTime` > `TimeStep`. For corresponding results on simulations of liquid water see figure 2 in Ref. [4]. Make a similar graph for liquid Lennard-Jonesium.

5.6 Effect of periodic boundary versus vacuum boundary

A vacuum boundary condition can be realized by changing from one job to the next the size of the periodic box to infinite, while keeping the number of atoms constant. A multiplication of the values `BoxSize(x/y/z)` by a factor three will be sufficient. Why? Use `MDType` = 1. Why? Use `TemperatureCouplingTime` = 10 * `TimeStep` and `TimeStep` = 0.005. Equilibrate the vacuum system for 500 steps and continue the run for another 10000 steps for analysis. Compare the average values of E_{pot}, $g(r)$, etc. to those of the periodic system.

6 Extensions of the MD program

When discussing possible extensions of the MD program no complete referencing to original papers is attempted in order to keep the list of references limited. However, original papers can be traced via the references given here. Related extensions are grouped together in paragraphs. The effort required for implementation of the various extensions of the MD program will differ from one extension to the other.

6.1 Algorithm for MD

Numerical methods to solve sets of differential equations can be found in almost any general textbook on applied mathematics. All methods are based on finite differences and solve the equations step by step in time. A discussion of algorithms suitable to application in MD simulations has been given by Berendsen and van Gunsteren [6].

6.1.1 Verlet algorithm with improved velocity formula

Modify the MD program such that the integration of the equations of motion is performed by the algorithm of Verlet [26]. Compare the conservation of E_{tot} with that obtained using the leap-frog scheme. Apply the more accurate formula (3.45) of Ref. [6] and discuss the result.

6.1.2 Beeman algorithm

Modify the MD program such that the integration of the equations of motion is performed by the algorithm proposed by Beeman [3]. Compare the conservation of E_{tot} with that obtained using the leap-frog or Verlet scheme and discuss the result.

6.1.3 Gear algorithm

Modify the MD program such that the integration of the equations of motion is performed by a fourth-order Gear algorithm [9]. Compare the performance with that obtained with the leap-frog or Verlet scheme and discuss the result.

6.2 The interatomic interaction function

Below a few changes of the applied interaction function are evaluated.

6.2.1 Exponential repulsion

Modify the repulsive term in the interaction function to an exponential $Ae^{-Br_{ij}}$ one. First find appropriate parameters A and B such that the position and the depth of the exponential potential well are similar to those obtained of the Lennard-Jones interaction function. Perform a MD simulation and compare the results with those obtained for Lennard-Jonesium.

6.2.2 Switching functions

When applying a cut-off radius R_c a discontinuity is introduced in the potential $V(r)$ at a distance R_c. The discontinuity can be removed by the introduction of a so-called *switching function* $S(r)$ by which the potential function $V(r)$ is to be multiplied (see Ref. [6]). Show that the lowest degree polynomial function $S(r)(R_\mathrm{S} \leq r \leq R_\mathrm{c})$ that obeys the conditions:

$$S(r) = 1 \text{ and } \frac{\mathrm{d}S}{\mathrm{d}r} = 0 \text{ at } r = R_S \tag{36}$$

and

$$S(r) = 0 \text{ and } \frac{\mathrm{d}S}{\mathrm{d}r} = 0 \text{ at } r = R_c \tag{37}$$

is given by

$$S(r) = \frac{(R_\mathrm{c} - r)^2 (R_\mathrm{c} + 2r - 3R_\mathrm{S})}{(R_\mathrm{c} - R_\mathrm{S})^3} \tag{38}$$

Determine $\mathrm{d}S(r)/\mathrm{d}r$ and modify the MD program according to this switching function. Perform a MD simulation and compare the results to those obtained without a switching function.

6.2.3 Shifted potential

An alternative to let both the potential and the force vanish at the cut-off radius R_c is to apply a shifted potential (see Ref. [6]):

$$V_\mathrm{shifted}(r) = V(r) - V(R_\mathrm{c}) - \left.\frac{\mathrm{d}V(r)}{\mathrm{d}r}\right|_{r=R_\mathrm{c}} (r - R_\mathrm{c}) \tag{39}$$

Modify the MD program accordingly, perform a MD simulation and compare the results to those obtained without a shifted potential.

6.3 Simulation of molecular systems

6.3.1 Polyatomic molecules using harmonic bonds

Change the MD program into a program for polymer dynamics by adding a harmonic interaction

$$V^{\mathrm{ho}}(\mathbf{r}) = \sum_{i=1}^{N-1} \frac{1}{2} K_0 \left[r_{i,i+1} - r_0\right]^2 \tag{40}$$

to the Hamiltonian, where K_0 is a force constant and r_0 an ideal bond length. Simulate polymer dynamics and the end-to-end distance distribution as a function of N and T. Compare the results with those of a freely joined polymer chain. See Ref. [8] for more details.

6.3.2 Polyatomic molecules using bond constraints

Change the MD program into a program for polymer dynamics by adding a constraint with length r_0 between sequential atoms. Use the shake algorithm to constrain the bond lengths. Answer the same questions as in the previous question. See Ref. [5] for details.

6.4 Searching neighbor atoms

In MD simulations the bulk of the computer time is used for calculating of the non-bonded forces, that is, for finding the nearest neighbor atoms and subsequently evaluating the interaction terms for the obtained atom pairs. Therefore, various schemes for performing this task as efficiently as possible have been proposed (see Ref. [24]).

6.4.1 Atom pair-list technique

The simplest way to find the neighbors of an atom, that is, the atoms that lie within a distance R_c, is to scan all possible atom pairs in the system. For a system consisting of N atoms, the number of pairs amounts to $N(N-1)/2$, which makes the computer time required for finding the neighbors in this way proportional to N^2. Once the neighbors have been found, the time required for calculating the non-bonded interaction is proportional to N. Therefore Verlet [26] has proposed to update the neighbor list or *pair-list* only every n-th MD step, where n is typically of the order of 10. Implement the pair-list technique and evaluate its performance as a function of the number of atoms N, the cut-off radius R_c and the number of MD steps n between updates of the pair-list.

6.4.2 Linked-list technique

Hockney, Goel and Eastwood [13] have proposed a linked-list technique for finding neighbors which requires computer time proportional to N. In that method a grid (or mesh) consisting of (cubic) cells covers the space occupied by the system. The length h of the side of a cell is chosen such that $h > R_c$ and, if periodic boundary conditions are applied, such that

$$nh = B \qquad\qquad n = \text{integer}, \geq 2 \tag{41}$$

where B denotes the length of the sides of the (cubic) periodic box. First, it is determined to which cell each atom belongs. The computer time for this operation is proportional to N. Then, all neighbors of an atom are to be found in either its own cell or in the 26 neighbor cells. The time for this operation is proportional to the number of grid cells in the system. The name *linked-list* technique originates from the way the bookkeeping of which atoms belong to which cells is done. Implement the linked-list technique and evaluate its performance as a function of the number of atoms N and the cut-off radius R_c.

6.4.3 Grid-cell technique

The neighbor search technique proposed by Quentrec and Brot [18] also uses a grid, but here the grid cells are chosen so small that not more than one atom occur in one cell. In this way the bookkeeping using a linked list is avoided. In order to ensure that no two atoms occur in one cell one takes $h < R_{\mathrm{vdW}}$, where R_{vdW} denotes the van der Waals radius of the smallest atom in the system. For periodic systems condition (41) is applied in addition. When calculating the non-bonded interaction all the grid cells are scanned. If a cell is occupied by an atom, all neighbor cells within a volume approximating the cut-off sphere are scanned for neighbor atoms. Implement the grid-cell technique and evaluate its performance as a function of the number of atoms N and the cut-off radius R_c.

6.5 Analysis of MD trajectories

6.5.1 Formulae for averaging

When analyzing a MD trajectory averages $\langle x \rangle$ and fluctuations $\left\langle \Delta x^2 \right\rangle^{\frac{1}{2}} = \left\langle (x - \langle x \rangle)^2 \right\rangle^{\frac{1}{2}}$ of a quantity x are to be computed, see, e.g., formulae (33 – 34). The variance σ_x^2 of a series of N_x values, $\{x_i\}$, can be computed from

$$\sigma_x^2 = \frac{1}{N_x - 1} \left(\sum_{i=1}^{N_x} x_i^2 - \frac{1}{N_x} \left(\sum_{i=1}^{N_x} x_i \right)^2 \right) \tag{42}$$

Unfortunately this formula is numerically not very accurate, especially when σ_x is small compared to the values of x_i. The following (equivalent) expression is numerically more accurate

$$\sigma_x^2 = \frac{1}{N_x - 1} \sum_{i=1}^{N_x} (x_i - \langle x \rangle)^2 = \frac{Q_{N_x}}{N_x - 1} \tag{43}$$

with

$$\langle x \rangle = N_x^{-1} \sum_{i=1}^{N_x} x_i = \frac{S_{N_x}}{N_x} \tag{44}$$

Using formulae (43 – 44) one has to go twice through the series of x_i values, once to determine $\langle x \rangle$ and again to compute σ_x^2, whereas formula (42) requires only one sequential scan of the series $\{x_i\}$. However, one may cast formula (43) in another form, containing partial sums, which allows for a sequential update algorithm ($i = 2, 3, \ldots, N_x$)

$$Q_i = Q_{i-1} + \frac{(S_{i-1} - (i-1)x_i)^2}{i(i-1)} \tag{45}$$

and

$$S_i = S_{i-1} + x_i \tag{46}$$

Using formulae (45 – 46) the average $\langle x \rangle$ and the fluctuation $\left\langle \Delta x^2 \right\rangle^{\frac{1}{2}}$ can be obtained by one sweep through the data. Implement formulae (45 – 46) in the MD program and compare the performance to that of formula 42.

6.5.2 Computation of correlation functions

When analyzing a MD trajectory often the auto correlation function

$$C(t) = (T_{\mathrm{MD}} - t)^{-1} \int_0^{T_{\mathrm{MD}} - t} x(t') x(t' + t) \mathrm{d}t' \tag{47}$$

for a quantity $x(t)$ is to be calculated. In case the quantity $x(t)$ is an atomic quantity $x_i(t)$ the averaging can also be performed over equivalent atoms $i = 1, 2, \ldots, N$

$$C(t) = \langle x_i(t')x_i(t'+t)\rangle_{t',i} \tag{48}$$

In general the quantities $x(t)$ are only available at N_{MD} discrete equally spaced time points $n\Delta t$ with $n = 0, 1, \ldots, N_{\mathrm{MD}} - 1$. The discrete equivalent of (47) is then

$$C(n\Delta t) = (N_{\mathrm{MD}} - n)^{-1} \sum_{k=0}^{N_{\mathrm{MD}}-n-1} x(k\Delta t)x((k+n)\Delta t) \tag{49}$$

This *direct multiplication* formula for calculation of the correlation function $C(t)$ requires computation time proportional to N_{MD}^2. A much faster method based on the convolution theorem combined with a fast Fourier transform (FFT) algorithm may be used to improve the efficiency of calculating correlation functions [14]. It proceeds as follows. The discrete Fourier transform of the quantity $x(t)$ with respect to time t is

$$\hat{x}(m\Delta\omega) = \sum_{k=0}^{N_{\mathrm{MD}}-1} x(k\Delta t)e^{im\Delta\omega k\Delta t} \tag{50}$$

where $m = 0, 1, \ldots, N_{\mathrm{MD}} - 1$ and

$$\Delta\omega = \frac{2\pi}{N_{\mathrm{MD}}\Delta t} \tag{51}$$

By taking the Fourier transform of (49) and using the convolution theorem the summation (integral) reduces to a product of the Fourier transformed function $\hat{x}(\omega)$ and its complex conjugate $[\hat{x}(\omega)]^*$, which may be subsequently inversely transformed to obtain the correlation function

$$C(n\Delta t) = \frac{1}{N_{\mathrm{MD}}(N_{\mathrm{MD}} - n)} \sum_{m=0}^{N_{\mathrm{MD}}-1} \hat{x}(m\Delta\omega)^*\hat{x}(m\Delta\omega)e^{-im\Delta\omega n\Delta t} \tag{52}$$

where $C(t)$ is assumed to be periodic with period $N_{\mathrm{MD}}\Delta t$. This assumption introduces spurious correlations in $C(t)$, which can be avoided by simply adding a series of N_{MD} zeros to the $n = 0, 1, \ldots, N_{\mathrm{MD}} - 1$ known values $x(n\Delta t)$. The summation in (52) now contains $2N_{\mathrm{MD}}$ terms and the FFT *expression* for the correlation function becomes

$$C(n\Delta t) = \frac{1}{2N_{\mathrm{MD}}(N_{\mathrm{MD}} - n)} \sum_{m=0}^{2N_{\mathrm{MD}}-1} \hat{x}(m\Delta\omega)^*\hat{x}(m\Delta\omega)e^{-im\Delta\omega n\Delta t} \tag{53}$$

with

$$\Delta\omega = \frac{2\pi}{2N_{\mathrm{MD}}\Delta t} \tag{54}$$

Using the FFT expression the required computer time becomes proportional to $N_{\mathrm{MD}} \log_2(N_{\mathrm{MD}})$, which is much better than the N_{MD}^2 dependence of the direct multiplication expression for $C(t)$. Write a program to compute the velocity auto correlation function of the Lennard-Jones atoms using both methods sketched above, the direct method (49) and the FFT method (53). Modify the MD program such that the atomic velocities are written to disk in a similar way as the atomic coordinates (trajectories) can be written to disk. Compare the performance of both methods as a function of the number of time frames N_{MD}.

6.6 MD with coupling to a temperature or pressure bath

A great variety of methods to perform MD at constant temperature or pressure is available, see Refs. [4, 11, 7].

6.6.1 Maxwellian thermalization

Andersen [2] has proposed a method for MD simulation at constant temperature T_0 in which the atoms in the system suffer from stochastic collisions. The procedure is the following. Classical MD is performed until the first stochastic collision occurs.

Suppose atom i is suffering the collision. The value of the momentum $\mathbf{p}_i$ of atom i after the collision is chosen at random from a Maxwellian distribution at temperature T_0. The change in momentum takes place instantaneously. All other atoms are unaffected by the collision. Then the classical equations of motion are integrated until the time of the next collision and the procedure is repeated. Atoms that are colliding should be picked randomly. This *Maxwellian thermalization* scheme depends on one parameter, the mean time Δt_{coll} between collisions. Implement this scheme in the MD program and evaluate its performance as a function of the value of Δt_{coll}. Compare the results to those obtained by the weak coupling scheme of section 2.5.

6.6.2 Stochastic coupling to a heat bath

An alternative stochastic coupling to a heat bath is the use of the *Langevin equation*

$$\frac{m_i d\mathbf{v}_i(t)}{dt} = \mathbf{f}_i(\{\mathbf{r}_i\}) - m_i \gamma_i \mathbf{v}_i(t) + \mathbf{f}_i^{\mathrm{st}}(t) \tag{55}$$

where the atomic friction coefficient is denoted by γ_i. The random force is denoted by $\mathbf{f}_i^{\mathrm{st}}(t)$. It is assumed to be a zero-mean Gaussian distributed random variable and to have no correlation with prior velocities, nor with the systematic force $\mathbf{f}_i(t)$. The width of the Gaussian distribution of $\mathbf{f}_i^{\mathrm{st}}(t)$ is related to the friction coefficient γ_i by the second fluctuation-dissipation theorem

$$\left\langle (\mathbf{f}_i^{\mathrm{st}})^2 \right\rangle = 6 m_i \gamma_i k_{\mathrm{B}} T_0 \tag{56}$$

see, e.g., Refs. [25, 23]. Schneider and Stoll [21] used (55) to couple the system to a heat bath. This procedure mimics frequent collisions with much lighter particles that have a Maxwellian velocity distribution at the given reference temperature T_0. Implement this *stochastic coupling* scheme in the MD program and evaluate its performance as a function of the value of the friction coefficient γ. Compare the results to those obtained by the other temperature coupling schemes discussed above.

6.6.3 Weak coupling to an external pressure bath

The weak coupling scheme for performing MD simulations at constant temperature T_0 discussed in section 2.5 can also be applied to coupling to a pressure bath of reference pressure P_0, see Ref. [4].The pressure scaling factor μ by which the atomic coordinates and the (periodic) box volume V is to be scaled in every MD time step, is

$$\mu = \left(1 + \frac{\beta_{\mathrm{T}} \Delta t}{\tau_{\mathrm{P}}} (P(t_n) - P_0) \right)^{\frac{1}{3}} \tag{57}$$

where the isothermal compressibility is denoted by β_{T} and the pressure coupling time constant by τ_{P}, in complete analogy with the treatment of the temperature scaling in section 2.5. Implement this *weak pressure coupling* scheme in the MD program and evaluate the effect of pressure coupling as was done in section 5.5 for the temperature coupling.

6.7 Other simulation methods

Other methods that may be used to simulate the behavior of liquid systems are the Monte Carlo (MC) method, which only yields static properties, and the method of Stochastic Dynamics (SD), which is an extension of the MD method.

6.7.1 Monte Carlo of simple liquids: 1-particle moves

The usual MC method to simulate simple liquids is the Metropolis method [16]. Given a starting configuration of an N atom system, a new configuration is generated by random displacement of one or more atoms. The displacements should be such that in the limit of a large number of successive displacements the available Cartesian space of all atoms is homogeneously sampled. The newly generated configuration is either accepted or rejected on the basis of an energy criterion involving the change ΔE of the potential energy (2) with respect to the previous configuration. It is accepted if $e^{-\Delta E/kT} > r$, where r is a random number homogeneously distributed over the interval (0,1). Otherwise it is rejected and the previous configuration is counted again and used as a starting point for another random displacement. Thus each configuration occurs in the ensemble with a probability proportional to its Boltzmann factor [1, 4]. Implement this 1-particle *Monte Carlo* scheme in the MD program and compare the MC simulated properties with those generated by the MD method: E_{pot}, ΔE_{pot}, $g(r)$, etc. as a function of the number of steps (averages, convergence). Choose the MC step $\Delta\mathbf{r} = (\Delta x, \Delta y, \Delta z)$ such that the acceptance ratio is about 0.5 for a given temperature.

6.7.2 Monte Carlo of simple liquids: N-particle moves

The Monte Carlo algorithm can also be implemented by performing N-particle random moves, i.e., execute the random moves of all N particles in the system simultaneously. Implement this N-particle Monte Carlo scheme in the MD program and compare the MC simulated properties with those generated by the generated by the MD method: E_{pot}, ΔE_{pot}, $g(r)$, etc. as a function of the number of steps (averages, convergence). Choose the MC step $\Delta\mathbf{r}^N = (\Delta x^N, \Delta y^N, \Delta z^N)$ such that the acceptance ratio is about 0.5 for a given temperature.

6.8 Simulation of atomic liquids

6.8.1 Determination of the diffusion constant from the mean square atomic displacement

The diffusion constant D can be determined from the mean square atomic displacements using the formula (see, e.g., Ref. [1]):

$$\frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{r}_i(t)-\mathbf{r}_i(0)\right)^2 = 6Dt \tag{58}$$

or the formula

$$\frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{r}_i(t)-\langle\mathbf{r}\rangle\right)^2 = 3Dt \tag{59}$$

In order to obtain the correct D, $\mathbf{r}_i(t)$ should be a continuous trajectory, that is, the periodicity translation (7) should not be applied to $\mathbf{r}_i(t)$ that are used in (58) or (59). Correct the D for the contribution of the (constant) motion of the center of mass of the system to the mean square displacement.

Monitor (plot) the quantity (58) or (59) as a function of time for different temperatures T and densities ρ. Choose at least one combination of T and ρ for which a comparison with literature values can be made (see, e.g., Ref. [19]). Determine the dependence of D on T and on ρ separately. Plot and explain the results. Compare the results with those obtained in section 6.8.2

6.8.2 Determination of the diffusion constant from the atomic velocity auto correlation function

The diffusion constant D can be determined from the atomic velocity auto correlation function using the formula [1]:

$$3D = \int_0^{T_{\mathrm{MD}}}\left(\frac{1}{N}\sum_{i=1}^{N}C_i(t)\right)dt \tag{60}$$

where the atomic velocity auto correlation function reads (see section 6.5.2):

$$C_i(t) = (T_{\mathrm{MD}}-t)^{-1}\int_0^{T_{\mathrm{MD}}-t}\mathbf{v}_i(t')\mathbf{v}_i(t'+t)dt' \tag{61}$$

and the time period covered by the MD simulation is denoted by T_{MD}.

Plot the integrand of (60) as a function of time for different temperatures T and densities ρ. Choose at least one combination of T and ρ for which a comparison with literature values can be made (see, e.g., Ref. [19]). Determine the dependence of D on T and on ρ separately. Plot and explain the results. Compare the results with those obtained in section 6.8.1

6.8.3 Surface structure of a liquid

A study of the structure of a liquid film can be found in Ref. [20]. An initial configuration of a periodic liquid film (in the x-, y-direction) can be obtained from a homogeneous liquid configuration in a box with lengths a, b, and c by enlarging c such that the atoms do not interact with periodic images in the z-direction.

Monitor the atom density in the z-direction as a function of time in order to determine the stability of the film. Is the stability dependent on the temperature T or the Lennard-Jones parameters ε and σ?

6.8.4 Simulation of liquid mixtures

A study of the local composition of a binary mixture can be found in Ref. [17]. Three model mixtures of components 1 and 2 are defined:

A: The association model: $\varepsilon_{12} = \varepsilon_{11}$

B: The Lorentz-Berthelot mixture: $\varepsilon_{12} = (\varepsilon_{11}\varepsilon_{22})^{\frac{1}{2}}$

C: The solvation model: $\varepsilon_{12} = \varepsilon_{22}$

The mass m and interaction parameters are those of argon, $\varepsilon_{\mathrm{Ar}} = (\varepsilon_{11}\varepsilon_{22})^{\frac{1}{2}}$ and $\varepsilon_{22} = 2\varepsilon_{11}$.

The local fraction of the two components may be defined by:

$$x_{11}(r) = \frac{n_{11}(r)}{n_{21}(r) + n_{11}(r)} \tag{62}$$

$$x_{22}(r) = \frac{n_{22}(r)}{n_{12}(r) + n_{22}(r)} \tag{63}$$

where $n_{ij}(r)$ is the number of atoms of type i around an atom of type j within a distance r

$$n_{ij}(r') = \int_0^r \frac{N_i}{V} g_{ij}(r') 4\pi r'^2 \mathrm{d}r' \tag{64}$$

and N_i is the number of atoms of type i in the computational box of volume V.

Determine the local fractions for the three model mixtures for two different temperatures. Check whether the systems are sufficiently equilibrated and discuss the results.

6.8.5 Deviation from ideal gas behavior

The Van der Waals equation of state

$$\left(P + a\frac{n^2}{V^2}\right)(V - nb) = nRT \tag{65}$$

provides a reasonably good representation of the PVT data of gases in the range of moderate deviations of ideal behavior. The Van der Waals constant for the compressibility of the atoms is denoted by a and the Van der Waals constant for the atomic volume is b. The parameters a and b describe the deviation from ideal behavior. They can be obtained for Lennard-Jonesium by rewriting (65) as

$$P = \frac{nRT}{V - nb} - \frac{n^2 a}{V^2} \tag{66}$$

and determining the pressure P as a function of the temperature T at a given volume V. The slope of $P(T)$ yields b and the value $P(T = 0)$ yields a.

Choose a number of VT values for which Lennard-Jonesium is in the gas phase, and plot P as a function of T with constant V. Determine, e.g., by least squares fitting, a and b and check for which density deviation from equation (65) occurs.

Use the obtained value of a and b to check the correctness of (65) by comparison with simulated results for state points in the neighborhood of the critical line.

6.8.6 Surface tension of a liquid film

Determine the surface tension of liquid Lennard-Jonesium as a function of temperature by direct simulation of a vapor-liquid coexistence region. Use the formula

$$\gamma = \frac{L_\mathrm{z}}{2}\left\langle\left[p_\mathrm{zz} - \frac{1}{2}\left(p_\mathrm{xx} + p_\mathrm{yy}\right)\right]\right\rangle \tag{67}$$

where L_z is the length of periodic box in the z-direction, perpendicular to the vapor-liquid surface, and p_xx, p_yy and p_zz are the diagonal elements of the pressure tensor. Literature: Ref. [22].

6.9 Extend classical particle interaction calculation

The intermolecular as well as intramolecular interactions for the potential energy of the simulated particle collection can be modelled by various approximations.

6.9.1 Diatomic molecules with harmonic and Morse potential bond term

The simplest approximation for the interatomic interaction of diatomic molecules consists of the so-called harmonic approximation given by

$$E_{\text{pot,harmonic}}(r) = \frac{1}{2}k_r(r - r_0)^2 \tag{68}$$

where r_0 is the equilibrium bond length corresponding to the bond length for the minimum energy configuration of the molecule. The harmonic force constant k_r is given by

$$k_r = \frac{d^2 E_{\text{pot}}}{dr^2}|_{r=r_0}. \tag{69}$$

While this approximation is valid for interatomic distances close to the equilibrium bond length, it is not able to model bond-breaking effects. This can be circumvented by employing the so-called Morse potential (Morse, 1929) given by

$$E_{\text{pot}}(r) = D_{\text{e}}(1 - e^{-a(r-r_{\text{e}})})^2 \tag{70}$$

where r is the distance between the atoms and r_e again the equilibrium distance. The constants D_e and a represent the well depth relative to the dissociation limit and the width of the potential, respectively.

Calculate the potential energy for the harmonic and Morse potential bond terms for different values of r. Determine a value for the force constant k_r in the harmonic approximation and the parameter a in the Morse potential, respectively. Plot, compare and explain the results. Can you think of any other suitable potential functions that suitably approximate the behaviour of diatomic molecules?

6.9.2 Polyatomic chain molecules without bond angle terms

The potential energy of a simulated system requires the specification of the system topology since it should account for all interactions which are physically important, e.g. covalent interactions or long range non-bonded interactions, respectively. In general, the potentential energy can be written as

$$E_{\text{pot,tot}} = E_{\text{vdW}} + E_{\text{Coul}} + E_{\text{cov}}. \tag{71}$$

In the equation above, the term E_{vdW} represents the van der Waals interactions which are normally modelled by a Lennard-Jones potential. The term E_{Coul} describes the Coulomb interactions and E_{cov} parametrizes the covalent interactions within your system. We can further split up E_{cov} into three contributions, namely

$$E_{\text{cov}} = E_{\text{bond}} + E_{\text{angle}} + E_{\text{dihedral}}. \tag{72}$$

The parametrization of each individual term in the equation above depends on the respective force field parametrization. For instance, for the AMBER force field (Cornell, 1995), we have

$$E_{\text{cov}}(r^N) = \sum_{\text{bonds}} k_b(r - r_0)^2 + \sum_{\text{angles}} k_a(\theta - \theta_0)^2 + \sum_{\text{torsions}} \sum_{n} \frac{1}{2} V_n[1 + \cos(n\omega - \gamma)]. \tag{73}$$

The parametrizations of the force constants can be found in the original publication (Cornell, 1995).

Run different simulations where you switch on/off the bond angle terms responsible for $E_{\text{cov}}(r^N)$. Plot and explain the potential energies for the different cases. When including the bond angle terms, try different parametrizations for the force constants. What kind of changes in the simulation can you observe? Which observable quantities are changed when you switch off the bond angle terms?

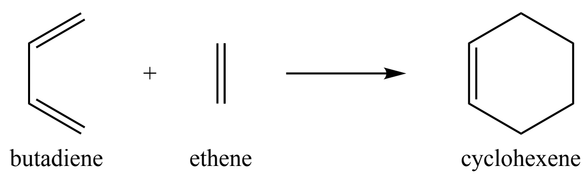
6.10 Calculate forces from an electronic structure method (Born–Oppenheimer MD)

The forces needed in a MD simulation can also be obtained directly from quantum mechanics; then, one speaks of *ab initio* molecular dynamics (AIMD), of which Born–Oppenheimer molecular dynamics (BOMD) is a sub-field. Because of the high computational cost of quantum chemical methods, such simulations can be very time-consuming. Here, we will limit the computational cost by employing semi-empirical quantum chemical method which are relatively fast due to a high degree of parametrization and precomputation of parameters.

We will provide you with a library that performs the calculation of semi-empirical energy and forces. You will have to integrate it into the MD program in order to perform BOMD simulations.

6.10.1 Reproduction of a Diels–Alder reaction

With the density-functional tight-binding (DFTB) method, we are able to actually reproduce chemical reactions in MD simulations. Here we consider the Diels–Alder reaction:



Set up initial coordinates for butadiene and ethene. Lauch several simulations where butadiene and ethene have different initial relative velocities and record what minimal velocity is needed in order to overcome the reaction barrier. Estimate the height of the reaction barrier and the pressure needed for the reaction to happen.

Optional: Study the influence of different substituents on the reaction (ask your assistant for more details).

6.10.2 Intra- and intermolecular reactions of polyenes

Unsaturated hydrocarbon chains are much more reactive than their saturated counterparts. Perform DFTB simulations of systems containing one or two unsaturated hydrocarbon chains $H_2C{=}CH{-}(CH{=}CH)_n{-}CH{=}CH_2$ at different temperatures and record the frequencies and types of chemical reactions. Repeat the simulations with more flexible chains by inserting saturated units ($-(CH_2)_m-$) in between the double bonds.

6.10.3 *Ab initio* MD simulations of mono- and diatomic gases

With the Parametrized Method 6 (PM6), another semi-empirical method, we are able to perform *ab initio* calculations of different gases. Reproduce MD simulations of noble gases (He, Ne, Ar, Kr) under hard-wall boundary conditions and compare them with simulations involving a Lennard–Jones potential only.

Then, study how different the properties of diatomic gases (H_2, N_2, O_2) are. Simulate mixtures of H_2 and O_2 as well as of H_2 and Cl_2 and study their chemical reactivity at different temperatures.

References

[1] M. P. Allen and D. J. Tildesley. *Computer Simulation of Liquids*. Clarendon Press, Oxford, 1987.

[2] H. C. Andersen. Molecular dynamics simulations at constant pressure and/or temperature. *J. Chem. Phys.*, 72:2384–2393, 1980.

[3] D. Beeman. Some multistep methods for use in molecular dynamics calculations. *J. Comput. Phys.*, 20:130–139, 1976.

[4] H. J. C. Berendsen, J. P. M. Postma, W. F. van Gunsteren, A. DiNola, and J. R. Haak. Molecular dynamics with coupling to an external bath. *J. Chem. Phys.*, 81:3684–3690, 1984.

[5] H. J. C. Berendsen and W. F. van Gunsteren. Stochastic dynamics of polymers. In J. W. Perram, editor, *Physics of Superionic Conductors and Electrode Material*, volume B92 of *NATO ASI Series*, pages 241–256, New York, 1983. Plenum.

[6] H. J. C. Berendsen and W. F. van Gunsteren. Practical algorithms for dynamic simulations. In G. Ciccotti and W. G. Hoover, editors, *Molecular-Dynamics Simulation of Statistical-Mechanical Systems, Proceedings of the International School of Physics "Enrico Fermi"*, volume course 97, pages 43–65, Amsterdam, North-Holland, 1986.

[7] D. Brown and J. H. R. Clarke. A comparison of constant energy, constant temperature and constant pressure ensembles in molecular dynamics simulations of atomic liquids. *Mol. Phys.*, 51:1243–1252, 1984.

[8] M. Doi and S. F. Edwards. *The Theory of Polymer Dynamics*. Oxford University Press, 1990.

[9] C. W. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice-Hall, Englewood Cliffs, N. J., 1971.

[10] J. P. Hansen and J.-J. Weis. Quantum corrections to the coexistence curve of neon near the triple point. *Phys. Rev.*, 188:314–318, 1969.

[11] D. M. Heynes. Molecular dynamics at constant pressure and temperature. *Chem. Phys.*, 82:285–301, 1983.

[12] J. O. Hirschfelder, C. F. Curtiss, and R. B. Bird. *Molecular theory of gases and liquids*. Wiley, New York, 1954.

[13] R. W. Hockney, S. P. Goel, and J. W. Eastwood. Quiet high-resolution computer models of a plasma. *J. Comput. Phys.*, 14:148–158, 1974.

[14] E. Kestemont and J. van Craen. On the computation of correlation functions in molecular dynamics experiments. *J. Comput. Phys.*, 22:451–458, 1976.

[15] D. A. McQuarrie. *Statistical Mechanics*. Harper & Row, New York, 1976.

[16] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087–1092, 1953.

[17] K. Nakanishi and K. Toukubo. Molecular dynamics studies of lennard-jones liquid mixtures. v. local composition in several kinds of equimolar mixtures with different combining rule. *J. Chem. Phys.*, 70:5848–5850, 1979.

[18] B. Quentrec and C. Brot. New method for searching for neighbors in molecular dynamics computations. *J. Comput. Phys.*, 13:430–432, 1973.

[19] A. Rahman. Correlations in the motion of atoms in liquid argon. *Phys. Rev.*, 136:405–411, 1964.

[20] M. Rao and D. Levesque. Surface structure of a liquid film. *J. Chem. Phys.*, 65:3233–3236, 1976.

[21] T. Schneider and E. Stoll. Molecular-dynamics study of a three-dimensional one-component model for distortive phase transitions. *Phys. Rev. B*, 17:1302–1322, 1978.

[22] I. G. Tironi, P. Fontana, and W. F. van Gunsteren. A molecular dynamics simulation study of liquid carbon tetrachloride. *Mol. Simulation*, 18:1–11, 1996.

[23] W. F. van Gunsteren and H. J. C. Berendsen. Algorithms for brownian dynamics. *Mol. Phys.*, 45:637–647, 1982.

[24] W. F. van Gunsteren, H. J. C. Berendsen, F. Colonna, D. Perahia, J. P. Hollenberg, and D. Lellouch. On searching neighbours in computer simulations of macromolecular systems. *J. Comput. Chem.*, 5:272–279, 1984.

[25] W. F. van Gunsteren, H. J. C. Berendsen, and J. A. C. Rullmann. Stochastic dynamics for molecules with constraints brownian dynamics of n-alkanes. *Mol. Phys.*, 44:69–95, 1981.

[26] L. Verlet. Computer "experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Phys. Rev.*, 159:98–103, 1967.

A Input Files

Test 1

```
# Title
Test 1: energy, virial, etc. of 2 atoms
#                  NumberAtoms                   AtomicMass                       MDType
                             2                            1                            0
#                    BoxSize(x)                   BoxSize(y)                   BoxSize(z)
                        10.436                       10.436                       10.436
#                NumberMDSteps                  InitialTime                     TimeStep
                             1                            0                        0.001
#           InitialTemperature            TargetTemperature     TemperatureCouplingTime
                             0                        1.095                          0.1
#                   RandomSeed
                      10021988
#   CoordInitializationSpread                FinalXVOutput
                             1                            0                            1
#            NAtomsOnBoxEdge(x)           NAtomsOnBoxEdge(y)           NAtomsOnBoxEdge(z)
                            10                           10                           10
#                    EpsilonLJ                      SigmaLJ    InteractionCutoffRadius
                             1                            1                          2.5
#     PropertyPrintingInterval    NumberRadialDistrPoints    RadialDistrCutoffRadius
                             1                          100                          2.5
#             TrajectoryOutput     TrajectoryOutputFormat   TrajectoryOutputInterval
                             1                            2                            1
```

Test 2

```
# Title
Test 2: energy, virial, etc. of 2 atoms
#                  NumberAtoms                   AtomicMass                       MDType
                             2                            1                            0
#                    BoxSize(x)                   BoxSize(y)                   BoxSize(z)
                        10.436                       10.436                       10.436
#                NumberMDSteps                  InitialTime                     TimeStep
                             1                            0                        0.001
#           InitialTemperature            TargetTemperature     TemperatureCouplingTime
                             0                        1.095                          0.1
#                   RandomSeed
                      10021988
#             XVInitialization   CoordInitializationSpread                FinalXVOutput
                             1                            0                            1
```

```
#          NAtomsOnBoxEdge(x)         NAtomsOnBoxEdge(y)         NAtomsOnBoxEdge(z)
                           10                         10                         10
#                   EpsilonLJ                    SigmaLJ    InteractionCutoffRadius
                            1                          1                        2.5
#    PropertyPrintingInterval   NumberRadialDistrPoints    RadialDistrCutoffRadius
                            1                        100                        2.5
#            TrajectoryOutput    TrajectoryOutputFormat   TrajectoryOutputInterval
                            0                          2                          5
```

Test 3

```
# Title
Test 3, 1000 atoms
#                 NumberAtoms                 AtomicMass                     MDType
                         1000                          1                          0
#                  BoxSize(x)                 BoxSize(y)                 BoxSize(z)
                       10.436                     10.436                     10.436
#               NumberMDSteps                InitialTime                   TimeStep
                         1000                          0                      0.005
#          InitialTemperature          TargetTemperature    TemperatureCouplingTime
                       120.27                     120.27                        0.1
#                  RandomSeed
                     10021988
#            XVInitialization  CoordInitializationSpread              FinalXVOutput
                            0                          0                          1
#          NAtomsOnBoxEdge(x)         NAtomsOnBoxEdge(y)         NAtomsOnBoxEdge(z)
                           10                         10                         10
#                   EpsilonLJ                    SigmaLJ    InteractionCutoffRadius
                            1                          1                        2.5
#    PropertyPrintingInterval   NumberRadialDistrPoints    RadialDistrCutoffRadius
                            1                        100                        2.5
#            TrajectoryOutput    TrajectoryOutputFormat   TrajectoryOutputInterval
                            0                          2                          5
```

Test 4

```
# Title
Test 4, 1000 atoms
#                 NumberAtoms                 AtomicMass                     MDType
                         1000                          1                          0
#                  BoxSize(x)                 BoxSize(y)                 BoxSize(z)
                       10.436                     10.436                     10.436
#               NumberMDSteps                InitialTime                   TimeStep
```

```
                         1000                          5                      0.005
#          InitialTemperature          TargetTemperature   TemperatureCouplingTime
                            0                     120.27                        0.1
#                  RandomSeed
                     10021988
#             XVInitialization  CoordInitializationSpread             FinalXVOutput
                            3                          0                          1
#           NAtomsOnBoxEdge(x)         NAtomsOnBoxEdge(y)         NAtomsOnBoxEdge(z)
                           10                         10                         10
#                    EpsilonLJ                    SigmaLJ    InteractionCutoffRadius
                            1                          1                        2.5
#    PropertyPrintingInterval   NumberRadialDistrPoints    RadialDistrCutoffRadius
                           10                        100                        2.5
#             TrajectoryOutput    TrajectoryOutputFormat   TrajectoryOutputInterval
                            0                          2                          5
```

Test 5

```
# Title
Test 5, 1000 atoms
#                  NumberAtoms                AtomicMass                     MDType
                         1000                          1                          0
#                   BoxSize(x)                 BoxSize(y)                 BoxSize(z)
                       10.436                     10.436                     10.436
#                NumberMDSteps               InitialTime                   TimeStep
                         1500                         10                      0.005
#          InitialTemperature          TargetTemperature   TemperatureCouplingTime
                            0                     120.27                        0.1
#                  RandomSeed
                     10021988
#             XVInitialization  CoordInitializationSpread             FinalXVOutput
                            3                          0                          1
#           NAtomsOnBoxEdge(x)         NAtomsOnBoxEdge(y)         NAtomsOnBoxEdge(z)
                           10                         10                         10
#                    EpsilonLJ                    SigmaLJ    InteractionCutoffRadius
                            1                          1                        2.5
#    PropertyPrintingInterval   NumberRadialDistrPoints    RadialDistrCutoffRadius
                          100                        100                        2.5
#             TrajectoryOutput    TrajectoryOutputFormat   TrajectoryOutputInterval
                            0                          2                          5
```

Test 6

```
# Title
Test 6, 1000 atoms
#               NumberAtoms                AtomicMass                    MDType
                       1000                         1                         1
#                BoxSize(x)                BoxSize(y)                BoxSize(z)
                     10.436                    10.436                    10.436
#             NumberMDSteps               InitialTime                  TimeStep
                       1000                         0                     0.005
#        InitialTemperature         TargetTemperature   TemperatureCouplingTime
                     120.27                    120.27                       0.1
#                RandomSeed
                   10021988
#          XVInitialization CoordInitializationSpread             FinalXVOutput
                          0                         0                         1
#        NAtomsOnBoxEdge(x)        NAtomsOnBoxEdge(y)        NAtomsOnBoxEdge(z)
                         10                        10                        10
#                 EpsilonLJ                   SigmaLJ   InteractionCutoffRadius
                          1                         1                       2.5
#  PropertyPrintingInterval  NumberRadialDistrPoints   RadialDistrCutoffRadius
                          1                       100                       2.5
#          TrajectoryOutput    TrajectoryOutputFormat  TrajectoryOutputInterval
                          0                         2                         5
```

B Initial Coordinates

Test 1

```
 2 Lennard-Jones atoms at rij=1 nm, coords x
   2
      0.0     0.0     0.0
      1.0     0.0     0.0
```

Test 2

```
 2 Lennard-Jones atoms at rij=2^1/6 nm, coords x
   2
      0.0     0.0     0.0
    1.122462  0.0     0.0
```