Dynamic simulation of molecular and ionic materials

David Fincham

SERC Daresbury Laboratory, Warrington, UK and Department of Physics, Keele University, Staffordshire, UK

Two new programs perform molecular dynamics simulations on molecular and ionic materials, respectively. The first simulates systems of small molecules, treated dynamically as rigid bodies, with interactions of the site–site Lennard–Jones form, plus point charges. The second simulates systems of point ions, which may be made polarizable by means of the shell model, treated by the method of adiabatic dynamics. Both programs handle Coulombic interactions in periodic boundaries by means of the Ewald sum. Force evaluations make use of efficient neighbor-search algorithms. The programs incorporate extensive analysis options in terms of structural and dynamic correlation functions. The programs are controlled by a command language, Dynamo, which provides a good environment for further development, as well as ease of use and flexibility in the choice of simulation protocol.

Keywords: molecular dynamics simulation, Ewald sum

INTRODUCTION

This article describes two new programs which perform molecular dynamics simulations of condensed phases of matter. Moliq-Dynamo simulates liquids and solids consisting of small, rigid molecules. The interaction between molecules is of the site–site Lennard–Jones form, with fractional charges on the sites to represent multipole moments. Shell-Dynamo simulates ionic materials using the Buckingham form of interionic potential. Polarizable ions can be represented by means of the shell model. The programs employ periodic boundaries, and currently are limited to cubic systems. In general, they use standard simulation techniques as described in the book by Allen and Tildesley,[1] but there are some novel features which are described below. The programs are controlled by a simple command language, Dynamo, which gives great flexibility and ease of use. They are available in source form (FORTRAN77 with common extensions) from the CCP5 Program Library,[2] and should be easily adaptable to other model systems.

Address reprint requests to Dr. Fincham at SERC Daresbury Laboratory, Warrington, WA4 4AD, UK; e-mail to D. Fincham @ KEELE.AC.UK.
Received 30 March 1993; revised 7 May 1993; accepted 12 May 1993

PROGRAM FEATURES

Model systems

In Moliq-Dynamo, the model system consists of small molecules which, dynamically, are treated as rigid bodies. The interaction between two molecules is expressed as a sum over interactions of the Lennard–Jones 12–6 form, between fixed sites on the molecules; sites may also carry charges. Thus the interaction potential between molecules i and j is of the form

$$U_{ij} = \sum_{\alpha \in i} \sum_{\beta \in j} \left\{ \frac{1}{4\pi\epsilon_0} \frac{q_{i\alpha} q_{j\beta}}{r_{i\alpha j\beta}} + 4\epsilon_{i\alpha j\beta} \left[\left(\frac{\sigma_{i\alpha j\beta}}{r_{i\alpha j\beta}} \right)^{12} - \left(\frac{\sigma_{i\alpha j\beta}}{r_{i\alpha j\beta}} \right)^{6} \right] \right\} \quad (1)$$

where $r_{i\alpha j\beta} = r_{i\alpha} - r_{j\beta}$ is the vector separating site α on molecule i from site β on molecule j, and $q_{i\alpha}$ is a site charge.

Molecule types are characterized by positions of the sites within the molecule, the site charges, and the parameters σ and ϵ. Sites may be positioned anywhere in the molecule, independent of the location of the mass centers: this is an advantage gained by the use of rigid-body equations of motion rather than the employment of bond-length constraints. The parameters σ and ϵ between sites on different molecule types are, by default, determined using standard mixing rules, but may be specified independently. Parameters such as the maximum number of molecular types, the maximum number of molecules in the system, and the maximum number of sites per molecule, are set at compile time.

Shell-Dynamo is closely related to Moliq-Dynamo but differs in several important respects. It does not deal with multicenter molecules, but only with single-center atoms or ions. As is conventional in the modeling of ionic materials, the interaction potential between the ions has the form

$$U_{ij} = \frac{1}{4\pi\epsilon_0} \frac{q_i q_j}{r_{ij}} + A_{ij} \exp\left[-\frac{r_{ij}}{\rho_{ij}} \right] - \frac{C_{ij}}{r_{ij}^6} - \frac{D_{ij}}{r_{ij}^8} \quad (2)$$

In addition, ions may be made polarizable by means of the shell model, which is widely used in static lattice calculations and harmonic lattice dynamics, but not to any great extent in dynamic simulations. In this model, an ion consists of a core and a shell, both carrying charges, and connected by a harmonic spring, thus allowing the ion to polarize. Non-Coulombic interactions between ions involve only the shells. The introduction of polarizability is known to have a

profound effect on the dynamics of ionic systems. Reliable and efficient techniques for shell-model dynamics have recently been described.[3,4]

Boundary conditions

The programs make the simulated systems pseudo-infinite by the use of periodic boundaries. These may be cubic, or in the case of liquid systems, truncated octahedral boundaries can also be used. The latter have the advantage that, for a given number of molecules or ions, they enable spatial correlations to be obtained to a greater distance. Periodic boundaries are implemented by means of the usual minimum-image transformation. In simulating ionic systems, or systems of dipolar molecules, the long-range nature of the interactions must be taken into account. This is achieved by the usual Ewald sum technique which separates the interaction into real-space and reciprocal-space parts, both of short range. The technique used to implement the Ewald sum in periodic truncated octahedral boundaries has recently been described.[5]

Neighbor searching

Interactions between molecules can be assumed to be negligible beyond a certain distance, the cut-off radius. This is true even in the case of Coulombic interactions if the Ewald sum is used. Corrections may be applied to the energy and pressure to compensate for omitted interactions beyond the cut-off: these corrections assume a uniform distribution of interaction centers. In the case of multicentered molecules Moliq-Dynamo applies the cut-off to the molecular center-of-mass separations.

It is important to locate interacting pairs efficiently, and the programs offer several neighbor searching options. The simplest technique is to loop over all the pairs in the system, apply the minimum-image transformation to the pair separation vector, then only evaluate the interaction if the magnitude of the separation is less than the cut-off radius. Rather than perform the loop over pairs every time step, it is possible to do it less frequently by setting up a list of pairs separated by less than an outer cut-off, somewhat larger than the interaction cut-off. This list is revised whenever it is necessary to do so to avoid missing any pairs whose separation may have come within the interaction cut-off. Typically, in a liquid this would be every ten time steps.

The all-pairs method clearly has an execution time which scales with the number N of molecules as N^2. Even if used in conjunction with a neighbor list, an N^2 component remains. For larger systems, say $N > 1000$, it is preferable to use a cell-based method for which the execution time is order N with short-ranged forces, and order $N^{1.5}$ when the Ewald sum is used and its parameters optimized. The methods rely on dividing the computational space into subcells, when the neighbors of a particular molecule can be found only in a restricted set of subcells surrounding its own subcell. In Moliq-Dynamo and Shell-Dynamo the subcells are taken to be cubic, with a side equal to half the cut-off radius: this gives near-optimum speed and is much better than the more common choice of subcell sides that are equal to the cut-off radius. The programs also have an option to use the cell decomposition to set up a neighbor list, and this is the fastest method of all for large systems.

Motion integration

The fundamental algorithm used to integrate the classical equations of motion is the leapfrog. This is a second-order time-reversible algorithm which gives good energy conservation and is stable at surprisingly large values for the time step.[6] In Moliq-Dynamo the ordinary leapfrog is applied to the motion of the centers-of-mass of the molecules. The orientation of a molecule is specified by quaternion parameters, and the rigid-body equations of motion are integrated by a recently developed implicit leapfrog-type algorithm;[7] an older explicit leapfrog is available as an option but is less accurate. In Shell-Dynamo, when treating polarizable ions, the mass is divided between the core and shell, and their motions are both integrated by the leapfrog algorithm. Appropriate choice of the shell mass ensures that the frequency of the core-shell spring lies well above the frequencies of the lattice vibrations. The motion of the fictitious internal degree of freedom of the ion is then adiabatic: it does not thermalize to the system temperature, but instead gains only sufficient energy to enable the ion to polarize. Under these circumstances the results obtained are in agreement with the massless shell model.

In addition to standard energy-conserving dynamics, options in the program allow for thermostatting the system, having the effect of allowing it to exchange energy with a heat bath at a specified temperature. There is also a controlled pressure option which allows the simulation volume to change in order to equilibrate the system to a specified pressure.

Analysis

Various types of analysis of the simulated system are incorporated into the programs and may be carried out during the course of the simulation. Thermodynamic quantities such as energy, temperature, and pressure are found on every time step, and incrementally averaged. Standard deviations are also found, and linear fits made, to look for drifts as a function of time. Other analyses may be switched on as required. Structural information is found by means of radial distribution functions. In the case of molecular systems both center–center and site–site distributions may be found. It is also possible to find various time autocorrelation functions. These include velocity and force, and in the case of Moliq-Dynamo, angular momentum, torque, and orientation. Diffusion coefficients may be found from mean-square displacements, as well as by integration of the velocity autocorrelation functions.

Other features

Initial configurations may be created by a lattice generation routine. Restart configurations containing coordinates and velocities, and optionally, all analysis accumulators may be written and read. Trajectory files of thermodynamic quantities, coordinates, and velocities may be written at intervals throughout a simulation for later analysis.

DYNAMO

The language Dynamo has been developed to provide an interface to simulation programs which can easily be adapted by academic researchers to suit their own requirements. There are two major advantages to the use of such a command language, compared to the more usual approach, in which programs are controlled by means of a file of parameters which is read at the beginning of program execution.

The first advantage is ease of development. Programs are normally developed in stages: typically one gets the basic algorithm going for a simple system; then it is extended to more complex systems; and then various options and analysis features are added. If a parameter file is being used, more and more parameters need to be added as the program evolves, so the format of the file is continually changing. Old files then cease to work, and to recreate the original, simple, test runs may involve extensive editing of what is, by now, quite a complex file. Using a command language one develops the program by adding new commands; but the old command files still work, and can be rerun to test that no errors have been introduced. In this way, the use of a command language encourages a modular and incremental approach to program development which minimizes errors and speeds progress.

The second advantage is ease and flexibility of use. Editing a parameter file to set up a particular simulation can be a tedious and error-prone task, particularly if it involves remembering the values and purposes of a large number of control switches. Use of a command language is much easier since commands can have meaningful names and only the commands and parameters actually required need to be specified. More importantly, the use of a command language can give great flexibility in the simulation protocol adopted. Most simulation programs controlled by parameter files have only one option: perform N steps of equilibration followed by M steps of measurement. Using the Dynamo language the user has complete freedom to change algorithms or parameters at any stage of the simulation, switch on or off any analysis, restart averaging, or print selected outputs. In a single execution of the program, it is quite possible to simulate several state points, change a mixture composition, or even simulate completely different systems.

The general form of a command in Dynamo is very simple: it consists of a command name, followed by none or more subcommands, followed by none or more string parameters, followed by none or more numerical parameters. Commands and subcommands may be abbreviated to four characters, and are case-insensitive. Commands are read from standard input so that Dynamo programs can be run interactively, which is useful during testing. A command OBEY enables a file of commands to be read and obeyed. It is useful to establish a library of such files describing various model systems. There is also a looping feature in the language. An ECHO command enables the output file to be annotated freely. Numerical parameters are read using FORTRAN list-directed reads. This enables built-in facilities for such reads to be used: for example, values may be omitted and will then take their default values.

To illustrate the use of Dynamo, Figure 1 gives a command file which carries out a simulation of liquid water using Moliq-Dynamo.

```
echo
echo  * TIPS2 model of water
echo  * see Jorgensen and others
echo  * J. Chem. Phys. 79 (1983) 926
echo

! This is a four-site model. One site has no mass
! Only the O site has a LJ interaction. Coords are
! given relative to the O
molecule WATER
!          x        y   z        sig     eps     q       m
ljsite Oxy  0.0      0.0 0.0      0.3241 65.17  0.0     16.0
ljsite Hyd -0.07570 0.0 0.05859 0.0      0.0   0.535   1.0
ljsite Hyd  0.07570 0.0 0.05859 0.0      0.0   0.535   1.0
ljsite M    0.0      0.0 0.015   0.0      0.0  -1.07    0.0
endmolecule

boundary psc
number water 256

state temper 293
state dengcc 1.0

value dt 0.002

! Start from f.c.c lattice
fillbox fcc
setvels

! Use ewald sum: default parameters with default cutoff
ewald

! Equilibration phase
output thermo 500
value twindow 25
runmd equil 2000

! Measurement phase
zero
rdf limits 0.2 1.0 0.02
rdf cc 20
acf velocs 4 50
msd 29 50
runmd leap 2000
output correl
```

Figure 1. Example of a Dynamo control file.

The command MOLECULE introduces the definition of a molecule type and gives it a name. The molecule definition is terminated by the command ENDMOLECULE. The command LJSITE specifies a Lennard–Jones (plus point charge) interaction site. In the potential model used here there are four sites, one of them massless. The command BOUNDARY PSC specifies the boundary as periodic simple cubic; this is actually the default and this command could have been omitted, but other options are available. The command NUMBER is used to specify the number of each type of molecule included in the simulation. The command STATE is used to specify the desired state point, here with subcommands TEMPER (required temperature in K) and DENGCC (density in gm per cc). Other subcommands enable a volume rather than density to be specified, and in addition a pressure for controlled pressure simulations.

The command VALUE is used to specify various numerical parameters: here the timestep is set (in ps). Many other quantities can be set by VALUE, for example the interaction cut-off, but all have defaults: here the default value for the interaction cut-off, half the box length, is used. FILLBOX sets up the initial lattice, and SETVELS sets the initial velocities, according to the required temperature. The command EWALD switches on the Ewald sum, which is advisable, as the water molecule has a large dipole moment.

The simulation illustrated has two phases, an equilibration phase and a measurement phase. The command OUTPUT THERMO gives an interval in steps between output of thermodynamic quantities, and like many commands it controls what happens during subsequent RUNMD commands. RUNMD is the only command which actually performs a simulation, and has a subcommand which specifies the type of dynamics to be used. EQUIL (equilibration dynamics) involves temperature windowing in combination with a heat-bath algorithm. After 2000 steps of equilibration the command FIXCM subtracts out net momentum of the system which may have built up as a result of numerical errors.

The command ZERO is particularly important in Dynamo. Any type of measurement may be used in conjunction with any type of dynamics, and thermodynamic quantities are always calculated and incrementally averaged. ZERO resets to zero all the accumulators, of correlation functions as well as of thermodynamic quantities, and so restarts all averaging and analysis. Here it is used to start the measurement phase of the simulation. The commands RDF, ACF, and MSD switch on radial distribution functions, time autocorrelation functions and mean-square displacement analyses for the next RUNMD call. They have a few subcommands and numerical parameters which it is not necessary to describe here. The command RUNMD LEAP performs 2000 steps of simulation using standard leapfrog (energy-conserving dynamics). Finally, OUTPUT CORREL will print out all nonzero correlation functions which have been accumulated during this phase of the simulation.

CONCLUSIONS

Moliq-Dynamo and Shell-Dynamo provide many features which make them useful in the simulation of molecular and ionic materials, respectively. The incorporation of the shell model of ionic polarizability, in particular, is a novel feature which enables dynamical processes in ionic materials to be studied much more realistically than has hitherto been possible. The language Dynamo makes the programs easy to use and allows very great flexibility in the simulation protocol. In addition, it provides a very suitable framework for the development of new simulation programs. A major restriction of the present programs, which will be lifted in future versions, is their limitation to cubic systems.

REFERENCES

1 Allen, M.P. and Tildesley, D.J. *Computer Simulation of Liquids*. Oxford University Press, Oxford, 1986

2 Programs obtainable from The CCP5 Librarian, SERC Daresbury Laboratory, Warrington WA4 4AD, UK

3 Lindan, P.J.D. and Gillan, M.J. Shell-model molecular dynamics simulation of superionic conduction in CaF_2. *J. Phys. Condens. Matter*. 1993, **5**, 1019–1031

4 Mitchell, P.J. and Fincham, D. Shell model simulations by adiabatic dynamics. *J. Phys. Condens. Matter*. 1993, **5**, 1031–1038

5 Smith, W. and Fincham, D. The Ewald sum in truncated octahedral and rhombic dodecahedral boundary conditions. *Molecular Simulation*. 1993 (in press)

6 Fincham, D. Choice of time step in molecular dynamics simulation. *Comput. Phys. Commun*. 1986, **40**, 263–269

7 Fincham, D. Leapfrog rotational algorithms. *Molecular Simulation*. 1992, **8**, 165–178