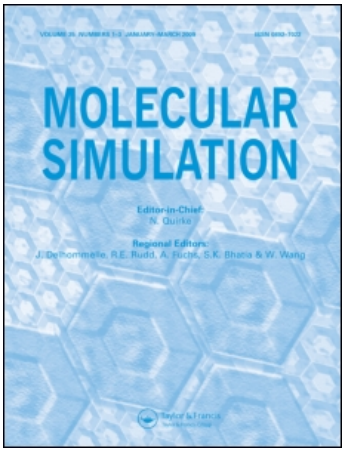

Molecular Simulation
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713644482

Leapfrog Rotational Algorithms for Linear Molecules

David Fincham[ab]
[a] Department of Physics, University of Keele, Keele, Staffordshire, U.K. [b] SERC Daresbury Laboratory, Warrington, U.K.

To cite this Article Fincham, David(1993) 'Leapfrog Rotational Algorithms for Linear Molecules', Molecular Simulation, 11: 1, 79 — 89
To link to this Article: DOI: 10.1080/08927029308022178
URL: http://dx.doi.org/10.1080/08927029308022178

Molecular Simulation, 1993, Vol. 11, pp. 79–89
Reprints available directly from the publisher
Photocopying permitted by license only

LEAPFROG ROTATIONAL ALGORITHMS FOR LINEAR MOLECULES

DAVID FINCHAM

Department of Physics, University of Keele, Keele, Staffordshire, ST5 5BG, U.K. and SERC Daresbury Laboratory, Warrington, WA4 4AD, U.K.
Email: D. Fincham @ UK.AC. KEELE

(Received August 1992, accepted September 1992)

A study is made of four algorithms which integrate the rotational equations of motion for rigid linear molecules. They are leapfrog algorithms in the sense that the quantities saved between time steps are the on-step orientation and the mid-step angular velocity. Thermostatted versions of the algorithms as well as conventional energy-conserving versions are described. The algorithms are extensively tested in simulations of liquid nitrogen, the aim being to study the effect of increased time steps on a range of measured properties. The most successful algorithm, based on applying a length constraint to the axis vector, shows remarkable stability and can be used with very large time steps.

KEY WORDS: molecular dynamics, rotation, leapfrog algorithm

1 INTRODUCTION

Molecular dynamics simulation of small rigid molecules is now a routine affair but, surprisingly, there is still something new to say about algorithms to integrate the rigid body equations of motion. Often these equations are integrated by predictor-corrector methods. While such algorithms are very accurate at small time steps, they tend to quickly become unstable as the time step is increased. But the primary requirement of molecular dynamics algorithms is stability, since the larger the time step the more rapidly can phase space be sampled, saving on expensive computer time. Because of the chaotic nature of particle trajectories accuracy is not of great importance. It is because of its stability that the simple second-order leapfrog algorithm is widely used to integrate translational motion. In a previous paper [1] I introduced and studied a new leapfrog algorithm for the rotational motion of rigid non-linear molecules, finding it to be useable with surprisingly large time steps, especially when thermostatted. In this paper I consider leapfrog algorithms for the rotational motion of linear molecules. Four such algorithms are described in Section 2.

Conservation of energy is a primary requirement for molecular dynamics algorithms. On this basis I am able in Section 3 to choose one of the four algorithms for further study. In the choice of timestep, the most appropriate technique is to steadily increase the time step and look for the effects of algorithm errors on a range of measured properties: thermodynamic, structural and dynamic. I test the chosen algorithm in this way, using liquid nitrogen as the test system.

2 LEAPFROG ALGORITHMS

2.1 The translational leapfrog

By writing Newton's law of motion as two first order equations

$$\frac{d\mathbf{v}}{dt} = \frac{\mathbf{f}}{m} = \mathbf{a} \tag{1a}$$

$$\frac{d\mathbf{r}}{dt} = \mathbf{v} \tag{1b}$$

and Taylor expanding to second order in the timestep Δ, for velocity $\mathbf{v}$ around time $n\Delta$ and for position $\mathbf{r}$ around time $(n+\frac{1}{2})\Delta$, it is easy to derive the leapfrog algorithm

$$\mathbf{v}^{n+\frac{1}{2}} = \mathbf{v}^{n-\frac{1}{2}} + \mathbf{a}^n \tag{2a}$$

$$\mathbf{r}^{n+1} = \mathbf{r}^n + \mathbf{v}^{n-\frac{1}{2}} \tag{2b}$$

in which velocity and position are defined on alternate half time steps. For convenience I choose units in which $\Delta = 1$. If an estimator for $\mathbf{v}^n$ is required, for example to check energy conservation, the usual choice is

$$\mathbf{v}^n = \mathbf{v}^{n-\frac{1}{2}} + \tfrac{1}{2}\mathbf{a}^n = \tfrac{1}{2}(\mathbf{v}^{n-\frac{1}{2}} + \mathbf{v}^{n+\frac{1}{2}}) \tag{2c}$$

but this is only first-order accurate. Various more accurate estimators are available [2].

It is possible to introduce various thermostats within the leapfrog framework. The unmodified on-step velocity is first calculated

$$\mathbf{v}_u^n = \mathbf{v}^{n-\frac{1}{2}} + \tfrac{1}{2}\mathbf{a}^n \tag{3a}$$

and scaled by a factor β

$$\mathbf{v}^n = \beta\,\mathbf{v}_u^n \tag{3b}$$

and the integration completed by

$$\mathbf{v}^{n+\frac{1}{2}} = (2-\beta^{-1})\,\mathbf{v}^n + \tfrac{1}{2}\mathbf{a}^n \tag{3c}$$

which satisfies $\mathbf{v}^n = \frac{1}{2}(\mathbf{v}^{n+\frac{1}{2}} + \mathbf{v}^{n-\frac{1}{2}})$. The position is then updated, equation (2b).

Different choices for β produce different thermostats. In this paper I use the thermostat of Berendsen *et al.* [3], which is implemented by

$$\beta = (1 + [T_{req}/T_u - 1][\Delta/\tau])^{\frac{1}{2}} \tag{3d}$$

where τ is a relaxation time, T_u is the unmodified instantaneous temperature

derived from the unmodified velocities (3a), and T_{req} is the required temperature. Other choices for β are discussed in [1].

2.2 *Rotational equations of motion*

For the rotational motion of a linear molecule, the orientation is conveniently specified by a unit vector **e** along the molecule axis. The position of an interaction site, situated at a distance d_α from the molecular centre of mass, is $\mathbf{r}_\alpha = d_\alpha \mathbf{e}$, and if $\mathbf{f}_\alpha$ is the force on this site due to interactions with other molecules, then the torque on the molecule is

$$\mathbf{T} = \sum_\alpha \mathbf{r}_\alpha \times \mathbf{f}_\alpha = \mathbf{e} \times \sum_\alpha d_\alpha \mathbf{f}_\alpha \tag{4a}$$

$$= \mathbf{e} \times \mathbf{g}$$

which defines the quantity **g** which we may call the "turning force". It is also useful to introduce the "perpendicular turning force" by subtracting out the component of **g** parallel to **e**, which has no effect on the rotational motion:

$$\mathbf{g}_p = \mathbf{g} - \mathbf{e}(\mathbf{e}.\mathbf{g}) \tag{4b}$$

Then

$$\mathbf{T} = \mathbf{e} \times \mathbf{g}_g \tag{4c}$$

and note that $\mathbf{T}^2 = \mathbf{g}_p^2$.

To derive leapfrog rotational algorithms it is necessary to write the equations of motion as two first order equations. For the motion of a general rigid body

$$\frac{\mathrm{d}\mathbf{J}}{\mathrm{d}t} = \mathbf{T} \tag{5a}$$

$$\mathbf{J} = \mathbf{I}\boldsymbol{\omega} \tag{5b}$$

$$\frac{\mathrm{d}\mathbf{e}}{\mathrm{d}t} = \boldsymbol{\omega} \times \mathbf{e} \tag{5c}$$

where **J** is angular momentum, $\boldsymbol{\omega}$ is angular velocity, **I** is the momentum of inertia tensor, and **e** is a vector fixed in the body. In the case of a linear molecule, there is only a scalar moment of inertia I, and $\boldsymbol{\omega}.\mathbf{e} = 0$. Equation (5) reduces to

$$\frac{\mathrm{d}\boldsymbol{\omega}}{\mathrm{d}t} = I^{-1}\,\mathbf{T} \tag{6a}$$

$$\frac{\mathrm{d}\mathbf{e}}{\mathrm{d}t} = \boldsymbol{\omega} \times \mathbf{e} \tag{6b}$$

There is an alternative to the use of the angular velocity $\boldsymbol{\omega}$, and that is to work

directly in terms of the velocity **u** of the axis vector:

$$\mathbf{u} = \frac{\mathrm{d}\mathbf{e}}{\mathrm{d}t} \tag{7a}$$

Differentiating, we find

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \frac{\mathrm{d}\boldsymbol{\omega}}{\mathrm{d}t} \times \mathbf{e} + \boldsymbol{\omega} \times \frac{\mathrm{d}\mathbf{e}}{\mathrm{d}t}$$

$$= \boldsymbol{I}^{-1}(\mathbf{e} \times \mathbf{g}_p) \times \mathbf{e} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{e})$$

We use $\mathbf{g}_p$ rather than **g** since it simplifies the next step. Using the rules for vector triple products, and using $\mathbf{e}.\mathbf{e} = 1$, $\mathrm{g}_p.\mathbf{e} = 0$, $\boldsymbol{\omega}.\mathbf{e} = 0$, leading to $\mathbf{u}^2 = \boldsymbol{\omega}^2$, this becomes

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = I^{-1}\,\mathbf{g}_p - u^2\mathbf{e} \tag{7b}$$

The second term on the right-hand side is just the centripetal acceleration necessary to preserve the normalisation of **e**. Leapfrog rotational algorithms for linear molecules can be based on either Equation (6) or Equation (7).

2.3 Rotational algorithms using $\boldsymbol{\omega}$

The previous paper [1] studied explicit and implicit leapfrog integrators for non-linear molecules based on a quaternion representation of the orientation. Similar, but simpler, versions of these algorithms can be obtained for linear molecules, and are based on the equations of motion in the form of Equation (6).

The explicit algorithm (denoted EXP in subsequent discussion) begins with an auxiliary part in which first-order expansion is used over half a step. The angular velocity is propagated from time $(n - \frac{1}{2})$ to time n (again taking units where $\Delta = 1$)

$$\boldsymbol{\omega}^n = \boldsymbol{\omega}^{n-\frac{1}{2}} + \tfrac{1}{2} I^{-1}\,\mathbf{T}^n \tag{8a}$$

and the orientation from n to $(n + \frac{1}{2})$

$$\mathbf{e}^{n+\frac{1}{2}} = \mathbf{e}^n + \tfrac{1}{2}(\boldsymbol{\omega}^n \times \mathbf{e}^n) \tag{8b}$$

These quantities are then used to propagate the mid-step angular velocity and on-step orientation in a manner analogous to the simple translational leapfrog

$$\boldsymbol{\omega}^{n+\frac{1}{2}} = \boldsymbol{\omega}^{n-\frac{1}{2}} + I^{-1}\,\mathbf{T}^n \tag{8c}$$

$$\mathbf{e}^{n+1} = \mathbf{e}^n + (\boldsymbol{\omega}^{n+\frac{1}{2}} \times \mathbf{e}^{n+\frac{1}{2}}) \tag{8d}$$

The thermostatted version of this algorithm proceeds by analogy with Equation (3).

The unmodified on-step angular velocity is found

$$\boldsymbol{\omega}_u^n = \boldsymbol{\omega}^{n-1/2} + \tfrac{1}{2} I^{-1} \mathbf{T}^n \tag{9a}$$

enabling the kinetic energy, unmodified temperature T_u, and hence the scaling factor β to be found. Then

$$\boldsymbol{\omega}^n = \beta \boldsymbol{\omega}_u^n \tag{9b}$$

$$\mathbf{e}^{n+1/2} = \mathbf{e}^n + \tfrac{1}{2}(\boldsymbol{\omega}^n \times \mathbf{e}^n) \tag{9c}$$

$$\boldsymbol{\omega}^{n+1/2} = (2 - \beta^{-1})\, \boldsymbol{\omega}^n + \tfrac{1}{2} I^{-1}\, \mathbf{T}^n \tag{9d}$$

$$\mathbf{e}^{n+1} = \mathbf{e}^n + (\boldsymbol{\omega}^{n+1/2} \times \mathbf{e}^{n+1/2}) \tag{9e}$$

The idea of the implicit algorithm (IMP) is to derive an implicit equation for $\mathbf{e}^{n+1}$ which can be solved, without the need for an auxiliary estimate of $\mathbf{e}^{n+1/2}$. In the translational case it is easy to show that the algorithm

$$\mathbf{v}^n = \mathbf{v}^{n-1/2} + \tfrac{1}{2}\mathbf{a}^n \tag{10a}$$

$$\mathbf{v}^{n+1/2} = \mathbf{v}^{n-1/2} + \mathbf{a}^n \tag{10b}$$

$$\mathbf{v}^{n+1} = \mathbf{v}^n + \mathbf{a}^n \tag{10c}$$

$$\mathbf{r}^{n+1} = \mathbf{r}^n + \tfrac{1}{2}(\mathbf{v}^n + \mathbf{v}^{n+1}) \tag{10d}$$

is equivalent to the simple leapfrog, Equation (2). The rotational analogue is

$$\boldsymbol{\omega}^n = \boldsymbol{\omega}^{n-1/2} + \tfrac{1}{2} I^{-1} \mathbf{T}^n \tag{11a}$$

$$\boldsymbol{\omega}^{n+1/2} = \boldsymbol{\omega}^{n-1/2} + I^{-1} \mathbf{T}^n \tag{11b}$$

$$\boldsymbol{\omega}^{n+1} = \boldsymbol{\omega}^n + I^{-1} \mathbf{T}^n \tag{11c}$$

$$\mathbf{e}^{n+1} = \mathbf{e}^n + \tfrac{1}{2}(\boldsymbol{\omega}^n \times \mathbf{e}^n + \boldsymbol{\omega}^{n+1} \times \mathbf{e}^{n+1}) \tag{11d}$$

This is an implicit equation for $\mathbf{e}^{n+1}$, which may be solved by iteration, taking as an initial guess

$$\mathbf{e}^{n+1} = \mathbf{e}^n + (\boldsymbol{\omega}^n \times \mathbf{e}^n) \tag{11e}$$

There is also a thermostatted version of this algorithm. In the translational case it is easy to show that the algorithm

$$\mathbf{v}_u^n = \mathbf{v}^{n-1/2} + \tfrac{1}{2}\mathbf{a}^n \tag{12a}$$

$$\mathbf{v}^n = \beta \mathbf{v}_u^n \tag{12b}$$

$$\mathbf{v}^{n+1/2} = (2 - \beta^{-1})\, \mathbf{v}^n + \tfrac{1}{2}\mathbf{a}^n \tag{12c}$$

$$\mathbf{v}^{n+\frac{1}{2}} = (3-2\beta^{-1})\ \mathbf{v}^n + \mathbf{a}^n \tag{12d}$$

$$\mathbf{r}^{n+1} = \mathbf{r}^n + \tfrac{1}{2}(\mathbf{v}^n + \mathbf{v}^{n+1}) \tag{12e}$$

is equivalent to the algorithm of Equation (3). The rotational version of this, in the present case, is

$$\boldsymbol{\omega}_u^n = \boldsymbol{\omega}^{n-\frac{1}{2}} + \tfrac{1}{2} I^{-1}\ \mathbf{T}^n \tag{13a}$$

$$\boldsymbol{\omega}^n = \beta \boldsymbol{\omega}_u^n \tag{13b}$$

$$\boldsymbol{\omega}^{n+\frac{1}{2}} = (2-\beta^{-1})\ \boldsymbol{\omega}^n + \tfrac{1}{2} I^{-1}\ \mathbf{T}^n \tag{13c}$$

$$\boldsymbol{\omega}^{n+1} = (3-2\beta^{-1})\ \boldsymbol{\omega}^n + I^{-1}\ \mathbf{T}^n \tag{13d}$$

and then $\mathbf{e}^{n+1}$ is found as above, using (11d) with (11e) as initial guess. It should be emphasised that, in all the above algorithms, the only quantitites saved to the next step are $\boldsymbol{\omega}^{n+\frac{1}{2}}$ and $\mathbf{e}^{n+1}$. All other quantities have an auxiliary role only.

2.4 Rotational algorithms using $\mathbf{u}$

Algorithms can also be derived from the alternative form of the equations of motion, (7a, b) by replacing the magnitude of the centripetal term in (7b) by an undetermined multiplier λ. Equations (7) become

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = I^{-1}\ \mathbf{g}_p + \mu \mathbf{e} \tag{14a}$$

$$\frac{\mathrm{d}\mathbf{e}}{\mathrm{d}t} = \mathbf{u} \tag{14b}$$

(It is possible to use $\mathbf{g}$ instead of $\mathbf{g}_p$.) Leapfrog integration of these equations could be accomplished by

$$\mathbf{u}^{n+\frac{1}{2}} = \mathbf{u}^{n+\frac{1}{2}} + I^{-1}\ \mathbf{g}_p^n + \lambda \mathbf{e}^n \tag{15a}$$

$$\mathbf{e}^{n+1} = \mathbf{e}^n + \mathbf{u}^{n+\frac{1}{2}} \tag{15b}$$

if the multiplier λ were known.

One method of determining λ is to apply the constraint that the length of the $\mathbf{e}$ vector remain unity. This is achieved as follows. Find the $\mathbf{e}$ vector which would result by ignoring the λ term

$$\hat{\mathbf{e}} = \mathbf{e}^n + \mathbf{u}^{n-\frac{1}{2}} + I^{-1}\ \mathbf{g}_p^n \tag{16a}$$

Then (15b) becomes

$$\mathbf{e}^{n+1} = \hat{\mathbf{e}} + \lambda \mathbf{e}^n \tag{16b}$$

and we require $(\mathbf{e}^{n+1})^2$ to be unity, i.e

$$1 = \hat{\mathbf{e}}^2 + 2\lambda \hat{\mathbf{e}}.\mathbf{e}^n + \lambda^2 (\mathbf{e}^n)^2$$

or, since $(\mathbf{e}^n)^2 = 1$

$$\lambda^2 + 2\lambda \hat{\mathbf{e}}.\mathbf{e}^n + \hat{\mathbf{e}}^2 - 1 = 0$$

This is a quadratic for λ which gives

$$\lambda = -\hat{\mathbf{e}}.\mathbf{e}^n + [(\hat{\mathbf{e}}.\mathbf{e}^n)^2 - \hat{\mathbf{e}}^2 + 1]^{1/2} \tag{16c}$$

(The positive square root is taken since λ must be small.) The algorithm is then: find $\hat{\mathbf{e}}$, using (16a); find λ, using (16c); find $\mathbf{e}^{n+1}$ using (16b). It is also necessary to find the new mid−step velocity, using

$$\mathbf{u}^{n+1/2} = \mathbf{e}^{n+1} - \mathbf{e}^n \tag{16d}$$

If $\mathbf{u}^n$ is required, to check energy conservation, it is found by

$$\mathbf{u}^n = \tfrac{1}{2}(\mathbf{u}^{n+1/2} + \mathbf{u}^{n-1/2}) \tag{16e}$$

I published this method informally a decade ago (4), deriving it by applying the method of constraints to an equivalent homonuclear diatomic. I refer to this as the LEN method (indicating that it involves a length constraint).

This algorithm cannot be thermostatted as directly as the other algorithms since the on−step velocity does not appear explicity. In this paper the mid-step velocity $\mathbf{u}^{n+1/2}$ is used to calculate the temperature and is then scaled by β before the next step.

It is possible to derive an alternative constraint algorithm by using the condition that $\mathbf{u}$ should remain perpendicular to $\mathbf{e}$ to determine the multiplier λ. Using the first-order expansion over half a step we obtain

$$\mathbf{u}^n = \mathbf{u}^{n-1/2} + \tfrac{1}{2} I^{-1} \mathbf{g}_p^n + \tfrac{1}{2}\lambda \mathbf{e}^n \tag{17a}$$

Applying the condition that $\mathbf{e}^n.\mathbf{u}^n = 0$ then gives

$$0 = \mathbf{u}^{n-1/2}.\mathbf{e}^n + \tfrac{1}{2} I^{-1} \mathbf{g}_p^n.\mathbf{e}^n + \tfrac{1}{2}\lambda \mathbf{e}^n.\mathbf{e}^n$$

By construction, $\mathbf{g}_p^n.\mathbf{e}^n = 0$, and also $\mathbf{e}^n.\mathbf{e}^n = 1$. Thus

$$\lambda = -2\mathbf{u}^{n-1/2}.\mathbf{e}^n \tag{17b}$$

The algorithm, which I call ORT since it involves an orthogonality constraint, is then: evaluate λ using (17b); find $\mathbf{u}^{n+1/2}$ using (15a) and $\mathbf{e}^{n+1}$ from (15b). If required, $\mathbf{u}^n$ can be found from (17a) or equivalently from (16e). This algorithm was also published informally by myself [5]

A thermostatted version of this algorithm can be obtained in a manner very

similar to the EXP algorithm above. After evaluating λ using (17b) it proceeds as follows:

$$\mathbf{u}_u^n = \mathbf{u}^{n-\frac{1}{2}} + \tfrac{1}{2}I^{-1}\,\mathbf{g}_p^n + \tfrac{1}{2}\lambda\mathbf{e}^n \tag{18a}$$

$$\mathbf{u}^n = \beta\mathbf{u}_u^n \tag{18b}$$

$$\mathbf{u}^{n+\frac{1}{2}} = (2-\beta^{-1})\,\mathbf{u}^n + \tfrac{1}{2}I^{-1}\,\mathbf{g}_p^n + \tfrac{1}{2}\lambda\mathbf{e}^n \tag{18c}$$

and $\mathbf{e}^{n+1}$ is found using (15b). This thermostatted version of the algorithm was introduced by Fincham, Quirke and Tildesley [6].

3 TESTS OF THE ALGORITHMS

I have implemented the algorithms described above within DYNAMO, a general purpose molecular liquids program. A simple leapfrog is used for the translational motion. When evaluating inter-molecular interactions the program applies the nearest-image transformation and spherical cut off to the centre-of-mass separations, not to the individual site-site separations. In the thermostatted versions of the algorithms the instantaneous translational and rotational temperatures are separately evaluated and controlled. The program uses single precision throughout. To avoid the build up of errors due to rounding the **e** vectors are renormalised on every step.

The test system chosen was a two-centre Lennard-Jones model of nitrogen (7), at a density of 0.7029 Mg/m^3 and temperature of 97 K. A number of runs with different time steps were carried out for the various algorithms, using a system of 216 molecules in periodic truncated octahedral boundaries with a cut off at 1.2 nm. Runs were of 24 ps after equilibration.

Examining first the results for total energy conservation, Table 1, it is clear that the LEN algorithm is superior to the others, showing remarkable stability with no

Table 1 Total energy in kJ/mol.

Algorithm	Δ/*fs*	*Mean*	*S.D.*	*Drift/ps*
EXP	10	−2.351	0.0061	0.0008
	15	−2.329	0.0216	0.0026
	20	−2.300	0.0432	0.0060
IMP	10	−2.368	0.0006	0.0001
	15	−2.341	0.0018	0.0002
	20	−2.348	0.0043	0.0006
LEN	10	−2.340	0.0004	0.0000
	15	−2.316	0.0006	0.0000
	20	−2.311	0.0010	0.0000
	30	−2.330	0.0028	0.0001
	36	−2.273	0.0186	0.0026
PRT	10	−2.346	0.0013	0.0002
	15	−2.324	0.0041	0.0006
	20	−2.323	0.0104	0.0015

Table 2. Thermodynamic results.

Δ/fs	*10*	*20*	*30*	*40*	*S.E.*
	Constant energy simulations				
Total energy/kJ mol^{-1}	−2.340	−2.311	−2.330		
Potential energy/kJ mol^{-1}	−4.362	−4.350	−4.351		0.004
Pressure/MPa	−0.4	−0.1	−0.4		0.8
Temperature/K	97.6	98.4	97.5		0.2
	Thermostatted simulations				
Total energy/kJ mol^{-1}	−2.350	−2.348	−2.359	−2.357	0.006
Potential energy/kJ mol^{-1}	−4.358	−4.352	−4.356	−4.348	0.004
Pressure/MPa	−0.1	−0.9	−0.4	1.2	0.8
Temperature/K	97.9	96.6	96.3	96.1	0.2

appreciable energy drift even with a time step as large as 30 fs. This is more than four times greater than the value of 7 fs used in reference [7] for this system. The IMP algorithm is the next best, followed by the ORT algorithm with the EXP algorithm being the worst. Of the two algorithms based on the equations of motion in **u** vector form, the ORT algorithm is clearly less satisfactory than the LEN algorithm. Unfortunately it is the former which is recommended in the standard text [8].

In view of its clear superiority I selected the LEN algorithm for further study, carrying out runs with constant energy dynamics for time steps up to 30 fs, and thermostatted runs for time steps up to 40 fs. None of the algorithms, even when thermostatted, was stable at a time step of 50 fs. Results for thermodynamic quantities are shown in Table 2. The last column of the table gives an estimate of the true standard errors in the measured means as determined by the method of Friedberg and Cameron [9, see also 10]. Although results differ in some cases by several standard errors, this may be attributed to the difficulty in achieving exactly the specified temperature. Taking this into account the overall agreement between the different simulations is excellent, with, for example, potential energies agreeing within 0.25%, and pressure agreement within 2 Mpa. Radial distribution functions were also found and showed excellent agreement.

To investigate diffusion centre-of-mass velocity autocorrelation functions were found and integrated out to 1.2 ps to get estimates of the diffusion constant. Results are shown in Table 3, and again are remarkably consistent between simulations.

The mean-square displacements (MSDs) of the molecular centres of mass were also found as a function of time, with averaging over time origins. Some

Table 3 Diffusion constant from velocity autocorrelation function. Units are $10^{-9}\,m^2\,s^{-1}$.

Δfs	*Constant energy*	*Thermostatted*
10	6.3	6.1
20	6.1	6.1
30	6.2	5.9
40		6.2

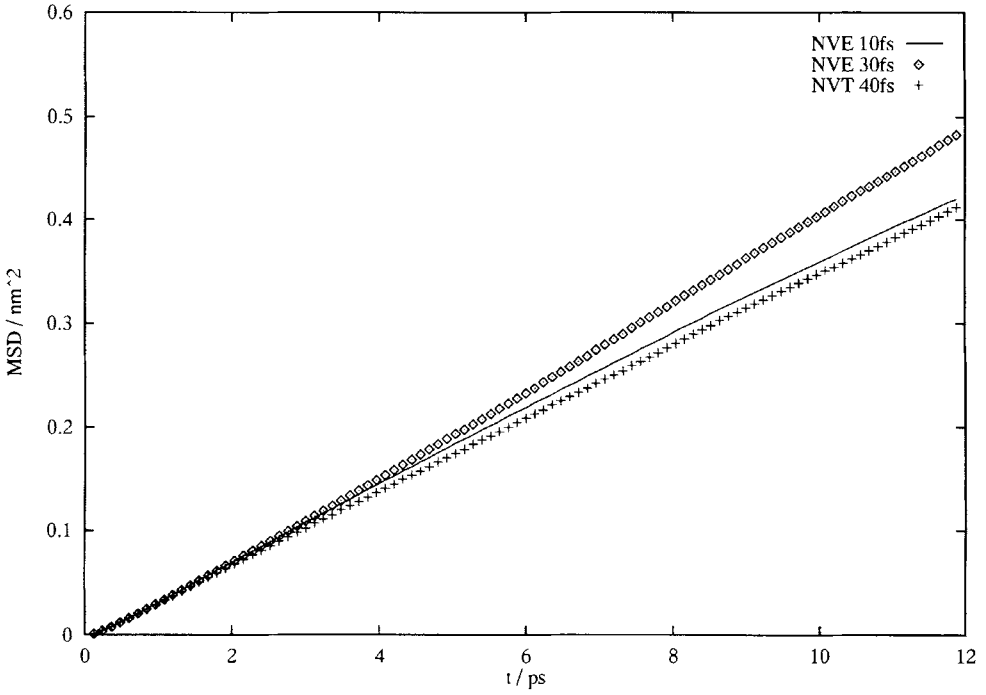

Figure 1 The mean-square displacements as functions of time. The continuous line is a constant energy simulation with a time step of 10 fs; the diamonds are a constant energy simulation with a time step of 30 fs; the crosses a thermostatted simulation with a time step of 40 fs.

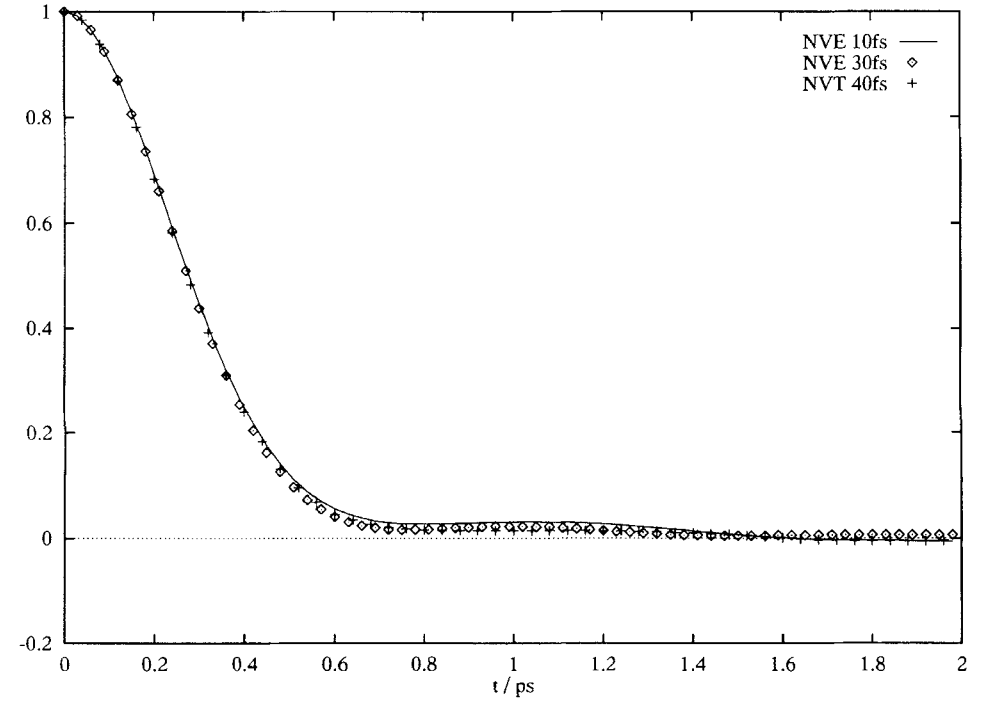

Figure 2 The PI orientational correlation function. Other details as in Figure 1.

discrepancies between the different simulations are evident at longer times (Figure 1), but in view of the results from the VACFs these may indicate no more than the difficulty of studying long-time correlations in simulations of small systems.

One of the most sensitive tests of a rotational algorithm is likely to be the molecular orientational relaxation. This was studied by evaluating the correlation function $P_1(t) = \langle \mathbf{e}(t).\mathbf{e}(0) \rangle$. Results are shown in Figure 2. Although some discrepancies occur, they are small.

4. CONCLUSIONS

The LEN algorithm, based on applying a length constraint to the axis vector, is clearly superior to the other algorithms and shows remarkable stability. In liquid nitrogen it can be used with the huge time step of 30 fs in constant energy simulations, and 40 fs when thermostatted.

References

[1] D. Fincham, "Leapfrog rotational algorithms", *Molecular Simulations,* **8**, 165 (1992)
[2] M. Amini and D.Fincham, "Evaluation of temperature in molecular dynamics simulation", *Comput. Phys. Commun.,* **56**, 313 (1990).
[3] H.J.C. Berendsen, J.P.M. Postma, W.F. van Gunsteren, A. di Nicola and J.R. Haak, "Molecular dynamics with coupling to an external bath", *J. Chem. Phys.,* **81**, 3684 (1984).
[4] D. Fincham, "Rotational motion of linear molecules", *Information Quarterly for MD and MC Simulations,* **10**, 43 (1983). This informal publication, the newsletter of the SERC project CCP5, is obtainable on request from SERC Daresbury Laboratory, Warrington WA4 4AD, U.K.
[5] D.Fincham, "More on rotational motion of linear molecules", *Information Quarterly for MD and MC Simulations,* **12**, 47 (1984).
[6] D. Fincham, N. Quirke and D.J. Tildesley, "Computer simulation of molecular liquid mixtures", *J. Chem. Phys.,* **84**, 4535 (1985).
[7] P.S.Y. Cheung and J.G. Powles, "The properties of liquid nitrogen", *Molec. Phys.,* **32**, 721 (1975).
[8] M.P. Allen and D.J.Tildesley, *Computer simulations of liquids*, Oxford University Press, 1986, pp 90–91.
[9] R. Friedberg and J.E. Cameron, "Test of the Monte Carlo method", *J. Chem. Phys.,* **52**, 6049 (1970).
[10] D. Fincham, "Choice of time step in molecular dynamics simulation", *Comput. Phys. Commun.,* **40**, 263 (1986).