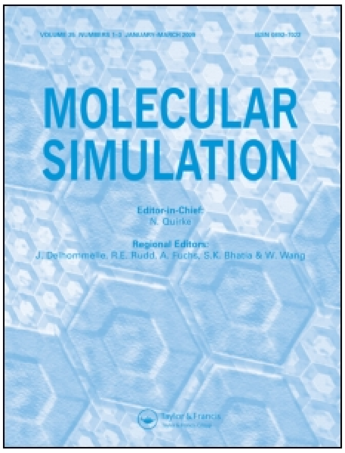

Molecular Simulation
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713644482

Leapfrog Rotational Algorithms

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

To cite this Article Fincham, David(1992) 'Leapfrog Rotational Algorithms', Molecular Simulation, 8: 3, 165 — 178
To link to this Article: DOI: 10.1080/08927029208022474
URL: http://dx.doi.org/10.1080/08927029208022474

Molecular Simulation, 1992, Vol. 8, pp. 165–178
Reprints available directly from the publisher
Photocopying permitted by license only

LEAPFROG ROTATIONAL ALGORITHMS

DAVID FINCHAM

SERC Daresbury Laboratory, Warrington WA4 4AD, U.K. and
Department of Physics, University of Keele, Keele, Staffordshire, ST5 5BG, U.K.;
Email: D. Fincham @ UK.AC.Keele Fax: +44 782 711093

(*Received May 1991; accepted September 1991*)

A new implicit rotational integrator for the orientation of rigid molecules is introduced, and compared with an existing explicit integrator. Both algorithms are categorised as leapfrogs since the quantities saved between time steps are the on-step orientation and the mid-step angular momentum. Orientations may be expressed in terms of principal axis vectors or, as in the implementations used here, quaternions. Thermostatted versions of the algorithms as well as conventional energy-conserving versions are described. The algorithms are extensively tested in simulations of liquid water, the aim being to study the effect of increased time steps on a range of measured properties. The implicit algorithm is superior to the explicit algorithm, and can be used with time steps up to 3 fs with energy-conserving dynamics. When thermostatted, it may be used with time steps up to at least 6 fs.

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. It is important to distinguish between the representation of the molecular orientation and the integration of its motion. Since there are three orientational degrees of freedom for non-linear molecules (linear molecules will be the subject of a separate paper) the orientation can be represented in terms of three parameters, the Euler angles. However, the equations of motion for the Euler angles contain singularities, and the most popular representation for the orientation is in terms of a four-component quaternion, with the single constraint of unit norm [1]. An alternative method of specifying molecular orientation is by means of three principal-axis unit vectors, giving nine components with the six constraints required by their orthonormality.

Many authors have integrated the quaternion motion with a high-order predictor-corrector algorithm. 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. By combining the translational leapfrog with a temperature constraint I was able to show [2] that reliable results could be obtained for liquid argon with the huge timestep of 60 fs.

In this spirit I published informally [3] an explicit second-order quaternion integrator a decade ago, which seemed to work well. Despite its inclusion in a review article [4] and the standard text [5] this integrator has never, to my knowledge, been extensively tested. The first objective of this paper is to remedy that omission. Since implicit integration schemes tend to be more stable than explicit schemes I also introduce here an implicit version of the integrator. Both algorithms are described in Section 2. Although I prefer to use a quarternion representation, the integration methods can equally well be applied to the principal axis vectors.

The alternative to direct integration of the rigid-body equations of motion is the use of a constraint method in which atoms of the molecule move independently under the applied forces, plus forces of constraint introduced to hold the inter-atomic distances constant. Such methods are generally believed to have good stability properties, but are complicated to implement when interaction sites are shifted away from mass centres, or there are more than the three atoms necessary to fix the orientation uniquely, or point electropole moments give torques directly, rather than site forces. Ahlrichs and Brode have introduced a hybrid method [6] in which the principal-axis vectors are treated as pseudo-atoms and constraint forces are introduced to maintain their orthinormality. From their results for energy conservation this is a good algorithm but it was not extensively tested and the runs were very short. Also, the algorithm is within a Verlet rather than leapfrog framework and does not contain the angular velocity explicitly. The present algorithms are more physically transparent since the torque, angular momentum and angular velocity all appear explicitly, and the inclusion of the velocity makes it easy to introduce a thermostat.They are also easy to substitute in an existing quaternion program.

How may an algorithm be tested? Conservation of energy is only a guide. 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. This was the technique used in [2] for translational motion, where it was shown that results are reliable until the time step is so large that a temperature drift is introduced. If the drift in temperature is controlled a further increase in time step is possible. In Section 3 I apply similar tests to a molecular system using the explicit and implicit rotational algorithms. The system chosen is liquid water which provides a very severe test of rotational algorithms because of the low moment of inertia of the molecule and the large torques due to the dipolar interactions.

2 LEAPFROG ALGORITHMS

2.1 *Basic Ideas*

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

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

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

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

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

but this is only first-order accurate. Various more accurate estimators are available [7].

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

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

and scaled by a factor β

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

and the integration completed by

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

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

Different choices for β produce different thermostats. If T_u is the unmodified instantaneous temperature derived from the unmodified velocities (3a), and T_{req} is the required temperature, then the constrained temperature dynamics of Brown and Clark [8] is produced by the choice

$$\beta = (T_{\mathrm{req}}/T_u)^{1/2} \tag{3d}$$

The thermostat of Berendsen *et al.* [9] is implemented by

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

where τ is a relaxation time. Toxvaerd's leapfrog formulation [10] of the Hoover thermostat [11] can be implemented by

$$\beta = (1 + \tfrac{1}{2}\eta^{n})^{-1} \tag{3f}$$

where η is integrated by

$$\eta^{n+1} = \eta^{n} + (T_p/T_{\mathrm{req}} - 1)(\Delta^2/\tau^2) \tag{3g}$$

with T_p being the temperature evaluated from the updated velocities $\mathbf{v}^{n+1/2}$.

We can write the equation of rigid body rotational motion as two first-order equations as follows. The rate of change of angular momentum is the torque

$$\frac{\mathrm{d}\mathbf{j}}{\mathrm{d}t} = \mathbf{t} \tag{4a}$$

and the angular velocity is related to the angular momentum by

$$\boldsymbol{\omega} = \mathbf{I}^{-1}\mathbf{j} \tag{4b}$$

where $\mathbf{I}$ is the moment of inertia tensor. If $\mathbf{e}$ is any vector fixed in the body then its velocity is

$$\frac{\mathrm{d}\mathbf{e}}{\mathrm{d}t} = \boldsymbol{\omega} \times \mathbf{e} \tag{4c}$$

The problem with deriving a leapfrog algorithm from Equation (4) is that the angular velocity depends on the orientation through the inertia tensor (4b). This cannot be handled with a simple leapfrog in which positions and velocities are known at different times. Below I derive explicit and implicit methods of solving this problem. (A similar problem arises with translation motion where the accelerations are velocity dependent, as with a charged particle moving in a magnetic field. It was exercises on this problem by my undergraduate students which first alerted me to the superiority of implicit over explicit algorithms.)

It is helpful to be able to transform between coordinate axes fixed in the rotating body and axes fixed in space. Let $\mathbf{e}_a$, $\mathbf{e}_b$, $\mathbf{e}_c$ be the orthogonal unit vectors of the principal-axis body-fixed system, in which the inertia tensor is diagonal. Take the origin of this system to be at the centre of mass of the molecule. Then, if a vector $\mathbf{d}$ is fixed in the body it can be expressed in terms of its components in the principal axis system, $(\hat{d}_a, \hat{d}_b, \hat{d}_c)$ and the instantaneous principal-axis vectors by

$$\mathbf{d} = \hat{d}_a\mathbf{e}_a + \hat{d}_b\mathbf{e}_b + \hat{d}_c\mathbf{e}_c \tag{5a}$$

or in matrix notation

$$\mathbf{d} = \mathbf{R}\hat{\mathbf{d}} \tag{5b}$$

where $\mathbf{R} = [\mathbf{e}_a, \mathbf{e}_b, \mathbf{e}_c]$ is called the rotation matrix. The transformation is orthogonal, $\mathbf{R}^T = \mathbf{R}^{-1}$ and so the inverse transformation, from space- to body-fixed systems, is

$$\hat{\mathbf{d}} = \mathbf{R}^T\mathbf{d} \tag{5c}$$

Instead of principal-axis unit vectors the orientation of the molecule can be represented by a four-component quaternion $\mathbf{q} = [q_0, q_1, q_2, q_3]^T$ with unit norm. The rotation matrix is given in terms of the quarternion components by [5]

$$\mathbf{R} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{6a}$$

where $\mathbf{R}$ is the transpose of the matrix $\mathbf{A}$ of reference [5]. If we write $\hat{\mathbf{w}} = [0, \hat{\omega}]^T$ Equation (4c) is replaced by

$$\frac{d\mathbf{q}}{dt} = \mathbf{Q}\hat{\mathbf{w}} \tag{6b}$$

where

$$\mathbf{Q} = \tfrac{1}{2}\begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \tag{6c}$$

The algorithms given below may be called leapfrog algorithms in the sense that two quantities are stored from time step $n - 1$ to time step n: an on-step orientation $\mathbf{\Omega}^n$ (where $\mathbf{\Omega}$ can be the set of principal-axis vectors $\mathbf{e}_a$, $\mathbf{e}_b$, $\mathbf{e}_c$ or the quaternion $\mathbf{q}$) and the angular momentum from the previous mid-step, $\mathbf{j}^{n-1/2}$. If $\hat{\mathbf{d}}_\alpha$ is the position of an interaction site in the molecule in the body-fixed system, its space-fixed position is calculated as

$$\mathbf{d}_\alpha^n = \mathbf{R}(\boldsymbol{\Omega}^n)\hat{\mathbf{d}}_\alpha \tag{7}$$

From the site positions the site forces $\mathbf{f}_\alpha^n$ are evaluated in the usual way, so that the torque

$$\mathbf{t}^n = \sum_\alpha \mathbf{d}_\alpha^n \times \mathbf{f}_\alpha^n \tag{8}$$

can be found. Of course, the total force on the molecule is also found so that its centre-of-mass motion can be integrated using the simple translational leapfrog, Equation (2).

2.2 *Explicit Algorithm*

This algorithm begins with an auxiliary part in which a first-order expansion is used over half a step. The angular momentum is propagated to time level n

$$\mathbf{j}^n = \mathbf{j}^{n-1/2} + \tfrac{1}{2}\mathbf{t}^n \tag{9a}$$

This is then rotated to the body-fixed system

$$\hat{\mathbf{j}}^n = \mathbf{R}^T(\boldsymbol{\Omega}^n)\mathbf{j}^n \tag{9b}$$

so that the principal components of angular velocity can easily be found. In this system Equation (4b) can be written

$$\hat{\boldsymbol{\omega}}^n = \hat{\mathbf{I}}^{-1}\hat{\mathbf{j}}^n \tag{9c}$$

and $\hat{\mathbf{I}}$ is diagonal and constant, with its diagonal elements being the principal moments of inertia, so that (9c) reduces to

$$\hat{\omega}_\alpha^n = \hat{I}_\alpha^{-1}\hat{j}_\alpha^n \tag{9d}$$

with $\alpha = a, b, c$ and with $\hat{I}_a$, $\hat{I}_b$, $\hat{I}_c$ being the principal moments of inertia. At this point the kinetic energy may be evaluated.

$$K = \tfrac{1}{2}\hat{I}_a\hat{\omega}_a^2 + \tfrac{1}{2}\hat{I}_b\hat{\omega}_b^2 + \tfrac{1}{2}\hat{I}_c\hat{\omega}_c^2 \tag{9e}$$

If representing orientation with axis vectors Equation (4c) can be used to propagate the orientation to time level $(n + \frac{1}{2})$ after rotating the angular velocity back to the space-fixed frame

$$\boldsymbol{\omega}^n = \mathbf{R}(\mathbf{e}^n)\hat{\boldsymbol{\omega}}^n \tag{10a}$$

$$\mathbf{e}^{n+1/2} = \mathbf{e}^n + \tfrac{1}{2}\boldsymbol{\omega}^n \times \mathbf{e}^n \tag{10b}$$

where $\mathbf{e}$ is any of $\mathbf{e}_a$, $\mathbf{e}_b$, $\mathbf{e}_c$. Alternatively, using quaternions, there is no need to rotate $\hat{\omega}$ since Equation (6b) can be used to propagate the quaternion

$$\mathbf{q}^{n+1/2} = \mathbf{q}^n + \tfrac{1}{2}\mathbf{Q}(\mathbf{q}^n)\hat{\mathbf{w}}^n \tag{11}$$

The main part of the algorithm uses the auxiliary value $\boldsymbol{\Omega}^{n+1/2}$ to integrate over a full time step in a manner very similar to the simple leapfrog

$$\mathbf{j}^{n+1/2} = \mathbf{j}^{n-1/2} + \mathbf{t}^n \tag{12a}$$

$$\hat{\mathbf{j}}^{n+1/2} = \mathbf{R}^T(\boldsymbol{\Omega}^{n+1/2})\mathbf{j}^{n+1/2} \tag{12b}$$

$$\hat{\omega}_x^{n+1/2} = \hat{I}_x^{-1}\hat{j}_x^{n+1/2} \tag{12c}$$

and then either

$$\boldsymbol{\omega}^{n+1/2} = \mathbf{R}(\mathbf{e}^{n+1/2})\hat{\boldsymbol{\omega}}^{n+1/2} \tag{13a}$$

$$\mathbf{e}^{n+1} = \mathbf{e}^{n} + \boldsymbol{\omega}^{n+1/2} \times \mathbf{e}^{n+1/2} \tag{13b}$$

or

$$\mathbf{q}^{n+1} = \mathbf{q}^{n} + \mathbf{Q}(\mathbf{q}^{n+1/2})\hat{\mathbf{w}}^{n+1/2} \tag{14}$$

This completes the algorithm. If using axis vectors it is strictly necessary to propagate only two of them to fix the molecular orientation..The angular momentum $\mathbf{j}^{n+1/2}$ and orientation $\boldsymbol{\Omega}^{n+1}$ are the only quantities that need to be saved for the next step. However, since the torque and angular velocity also appear explicitly they can easily be saved if required to calculate correlation functions.

The thermostatted version of the explicit algorithm follows Equation (9) so that the unmodified temperature and hence the scaling factor β may be found as in the translational case, Equation (3). After multiplying the angular momentum $\mathbf{j}^n$ by β, the algorithm proceeds as before, except that Equation (12a) is replaced by

$$\mathbf{j}^{n+1/2} = (2 - \beta^{-1})\mathbf{j}^{n} + \tfrac{1}{2}\mathbf{t}^{n} \tag{15}$$

2.3 *Implicit Algorithm*

The idea of this algorithm is to derive an implicit equation for $\boldsymbol{\Omega}^{n+1}$ which can be solved, without the need for an auxiliary estimate of $\boldsymbol{\Omega}^{n+1/2}$. As in the explicit algorithm, $\mathbf{j}$ is integrated directly from $(n - \frac{1}{2})$ to $(n + \frac{1}{2})$ by means of Equation (12a). In the translation case it is easy to show that the algorithm

$$\mathbf{v}^{n} = \mathbf{v}^{n-1/2} + \tfrac{1}{2}\mathbf{a}^{n} \tag{16a}$$

$$\mathbf{v}^{n+1/2} = \mathbf{v}^{n-1/2} + \mathbf{a}^{n} \tag{16b}$$

$$\mathbf{v}^{n+1} = \mathbf{v}^{n} + \mathbf{a}^{n} \tag{16c}$$

$$\mathbf{r}^{n+1} = \mathbf{r}^{n} + \tfrac{1}{2}(\mathbf{v}^{n} + \mathbf{v}^{n+1}) \tag{16d}$$

is equivalent to the simple leapfrog, Equation (2). In the above $\mathbf{v}^n$ and $\mathbf{v}^{n+1}$ are auxiliary quantities and do not need to be saved between steps. The rotational analogue of (16a–c) is

$$\mathbf{j}^{n} = \mathbf{j}^{n-1/2} + \tfrac{1}{2}\mathbf{t}^{n} \tag{17a}$$

$$\mathbf{j}^{n+1/2} = \mathbf{j}^{n-1/2} + \mathbf{t}^{n} \tag{17b}$$

$$\mathbf{j}^{n+1} = \mathbf{j}^{n} + \mathbf{t}^{n} \tag{17c}$$

In the case where orientation is represented by axis vectors the analogue of (16d) is

$$\mathbf{e}^{n+1} = \mathbf{e}^{n} + \tfrac{1}{2}(\boldsymbol{\omega}^{n} \times \mathbf{e}^{n} + \boldsymbol{\omega}^{n+1} \times \mathbf{e}^{n+1}) \tag{18a}$$

It is possible to find $\boldsymbol{\omega}^n$ as before by rotating the angular momentum to the body-fixed frame, dividing by the principal moments of inertia and rotating back

$$\boldsymbol{\omega}^{n} = \mathbf{R}(\mathbf{e}^{n})\hat{\mathbf{I}}^{-1}\mathbf{R}^{T}(\mathbf{e}^{n})\mathbf{j}^{n} \tag{18b}$$

A similar equation applies to $\boldsymbol{\omega}^{n+1}$

$$\omega^{n+1} = \mathbf{R}(\mathbf{e}^{n+1})\hat{\mathbf{I}}^{-1}\mathbf{R}^{T}(\mathbf{e}^{n+1})\mathbf{j}^{n+1} \tag{18c}$$

but of course $\mathbf{e}^{n+1}$ is not yet known. Thus (18) is an implicit equation for $\mathbf{e}^{n+1}$ which appears on the right hand side both directly, and through the rotation matrix. It could be solved iteratively, but I have not tried this, preferring to concentrate on the quaternion representation.

In the quaternion case the analalogue of (16d) is

$$\mathbf{q}^{n+1} = \mathbf{q}^{n} + \tfrac{1}{2}(\mathbf{Q}(\mathbf{q}^{n})\hat{\mathbf{w}}^{n} + \mathbf{Q}(\mathbf{q}^{n+1})\hat{\mathbf{w}}^{n+1}) \tag{19a}$$

It is possible to find $\hat{\mathbf{w}}^{n}$

$$\hat{\mathbf{w}}^{n} = [0, \hat{\omega}^{n}]^{T} = [0, \hat{\mathbf{I}}^{-1}\mathbf{R}^{T}(\mathbf{q}^{n})\mathbf{j}^{n}]^{T} \tag{19b}$$

A similar expression applied to $\hat{\mathbf{w}}^{n+1}$

$$\hat{\mathbf{w}}^{n+1} = [0, \hat{\mathbf{I}}^{-1}\mathbf{R}^{T}(\mathbf{q}^{n+1})\mathbf{j}^{n+1}]^{T} \tag{19c}$$

Equation (19) is an implicit equation for $\mathbf{q}^{n+1}$ which can be solved by iteration, taking

$$\mathbf{q}^{n+1} = \mathbf{q}^{n} + \mathbf{Q}(\mathbf{q}^{n})\hat{\mathbf{w}}^{n} \tag{20}$$

as the initial guess.

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

$$\mathbf{v}^{n} = \beta(\mathbf{v}^{n-1/2} + \tfrac{1}{2}\mathbf{a}^{n}) \tag{21a}$$

$$\mathbf{v}^{n+1/2} = (2 - \beta^{-1})\mathbf{v}^{n} + \tfrac{1}{2}\mathbf{a}^{n} \tag{21b}$$

$$\mathbf{v}^{n+1} = (3 - 2\beta^{-1})\mathbf{v}^{n} + \mathbf{a}^{n} \tag{21c}$$

$$\mathbf{r}^{n+1} = \mathbf{r}^{n} + \tfrac{1}{2}(\mathbf{v}^{n} + \mathbf{v}^{n+1}) \tag{21d}$$

is equivalent to the algorithm of Equation (3). In the rotational version Equation (17a) is used to calculate the unmodified angular momentum and then the angular velocity and temperature, and hence the scaling factor β. Equation (17) is then replaced by

$$\mathbf{j}^{n} = \beta(\mathbf{j}^{n-1/2} + \tfrac{1}{2}\mathbf{t}^{n}) \tag{22a}$$

$$\mathbf{j}^{n+1/2} = (2 - \beta^{-1})\mathbf{j}^{n} + \tfrac{1}{2}\mathbf{t}^{n} \tag{22b}$$

$$\mathbf{j}^{n+1} = (3 - 2\beta^{-1})\mathbf{j}^{n} + \mathbf{t}^{n} \tag{22c}$$

and the algorithm proceeds as before.

3 TESTS OF THE ALGORITHMS

I have implemented the quaternion versions of the explicit and implicit algorithms within DYNAMO, a general purpose molecular liquids program. A simple leapfrog is used for the translational motion. The quaternions are renormalised at the end of each time step (though I have no evidence that this is necessary). Thermostatted versions of the algorithms have also been implemented, the instantaneous translational and rotational temperatures being separately evaluated and controlled. When evaluating intermolecular interactions the program applies the nearest-image transformation and spherical cut off to the centre-of-mass separations, not to the individual site–site

Table 1 Thermodynamic results for SPC water over 24 ps runs, using implicit and explicit quaternion leapfrog rotational algorithms without thermostatting.

		Implicit			*Explicit*	
Δ/fs	*Value*	*S.D.*	*Drift*/ps	*Value*	*S.D.*	*Drift*/ps
Total energy/kJ mol^{-1}						
2	−34.43	0.03	0.003	−33.51	0.53	0.077
3	−33.94	0.02	0.021	−30.40	2.20	0.318
4	−33.35	0.47	0.07			
6	−28.17	3.47	0.50			
Temperature/K						
2	297.9	8.7	0.002	310.7	11.4	1.0
3	302.5	8.6	−0.01	353.4	34.3	4.7
4	308.5	10.7	0.8			
6	380.2	56.4	7.8			
Potential energy/kJ mol^{-1}						
2	−41.84	0.22	0.003	−41.24	0.43	0.05
3	−41.47	0.26	0.02	−39.19	1.42	0.20
Pressure/MPa						
2	26.3	72.3	0.2	69.2	75.4	1.3
3	45.2	73.4	−0.7	147.2	99.4	8.8
Mean square force/10^{-18}N^2						
2	0.447	0.045		0.465	0.048	
3	0.457	0.046		0.511	0.066	
Mean square torque/10^{-38}(N m)^2						
2	0.256	0.015		0.264	0.016	
3	0.257	0.014		0.280	0.020	

separations. The test system chosen was the SPC model of water [12], at a density of 0.997 Mg/m^3 and temperature of 298 K. This system is most commonly simulated using a constraint algorithm with a time step of 2 fs. To reduce cut-off effects I used a system of 256 molecules in cubic boundaries and an accurate Ewald sum. The real-space cut off was at half the box length (about 0.98 nm), the convergence parameter was $\alpha L = 7.0$, and the reciprocal space cut off was $k_{\max} = 8$. The program used single precision throughout. It was implemented in sequential mode on a Meiko Computing Surface using i860 processors and executed at around 10 s per time step. A number of runs with different time steps were carried out for the explicit and implicit leapfrog rotational algorithms. The implicit rotational algorithm converged to a tolerance of 10^{-5} in an average of 3.0 iterations with a 2 fs time step; 3.3 iterations with a 3 fs time step; 3.8 iterations with a 4 fs time step; and 4.6 iterations with a 6 fs time step. Simulations were performed with the Brown and Clarke constrained temperature thermostat (NVT dynamics) as well as with conventional energy-conserving (NVE) dynamics. All runs started from the same liquid configuration, re-equilibrated for 2.4 ps and then took measurements over 24 ps.

In the simulations the following thermodynamic quantities were evaluated every step and averaged: total energy, potential energy, temperature, pressure, and mean-square force and torque. Standard deviations were also measured, and drifts with time found by a linear fit through the values. Results are shown in Tables 1 and 2. The structure of the liquid was investigated by determining the oxygen–oxygen and oxygen–hydrogen radial distribution functions (RDFs). Oxygen–oxygen RDFs are shown in Figure 1. Diffusion was investigated in two ways. Centre-of-mass velocity autocorrelation functions (VACFs) were found (not shown) and integrated out to

Table 2 Thermodynamic results using implicit rotational algorithms incorporating a temperature constraint.

		Implicit			*Explicit*	
Δ/fs	*Value*	*S.D.*	*Drift*/ps	*Value*	*S.D.*	*Drift*/ps
Total energy/kJ mol^{-1}						
2	−34.40	0.37	−0.002	−34.40	0.40	−0.001
3	−34.47	0.36	−0.006	−36.42	2.28	−0.20
4	−34.16	0.41	−0.019	−36.72	1.71	−0.20
6	−33.81	0.41	−0.017	−34.51	0.52	−0.03
Temperature Constrained to 298 K						
Potential energy/kJ mol^{-1}						
2	−41.81	0.37	−0.002	−41.81	0.40	−0.001
3	−41.88	0.36	−0.006	−43.83	2.28	−0.30
4	−41.58	0.41	−0.009	−44.14	1.71	−0.20
6	−41.22	0.41	−0.017	−41.93	0.52	−0.03
Presure/MPa						
2	36.0	68.4	0.37	37.3	75.7	0.9
3	49.3	75.0	−0.04	24.7	79.2	−0.3
4	48.2	73.2	−0.51	34.0	75.4	−1.8
6	49.6	71.9	−0.02	38.9	76.8	1.3
Mean square force/10^{-18}N^2						
2	0.445	0.042		0.449	0.045	
3	0.454	0.045		0.448	0.048	
4	0.453	0.045		0.454	0.048	
6	0.460	0.047		0.457	0.048	
Means square torque/10^{-38}(Nm)^2						
2	0.255	0.015		0.257	0.015	
3	0.261	0.015		0.194	0.068	
4	0.264	0.015		0.198	0.042	
6	0.277	0.016		0.267	0.017	

0.6 ps to get estimates of the diffusion constant (Table 3). It should be noted that at 0.6 ps the VACFs still have a negative tail and that these values will therefore be overestimates. The angular momentum autocorrelation functions were also found (not shown). The mean-square displacements (MSDs) of the molecular centres of mass were also found as a function of time, with averaging over time origins (Figure 2). It was noticed that these graphs tended to have a slight change in slope at around 6 ps. Diffusion constants were therefore evaluated from the slope of a linear fit over the region 6–12 ps (Table 4). Orientational relaxation was studied by evaluating the correlation function $\langle \mathbf{m}(t) \cdot \mathbf{m}(0) \rangle$ where **m** is the molecular dipole axis. Results are shown in Figure 3.

Examining first the results with a 2 fs time step, it can be seen that the implicit NVE algorithm shows the best energy conservation, with a drift of only 1 part in 10^4 per ps. The ratio of the standard deviation in total energy to the standard deviation in potential energy, often used as a measure of the quality of energy conservation, is 12%. It should be noted that short runs can give a very misleading idea of energy conservation: over the first 2.4 ps of the run both the energy fluctuations and the rate of energy drift are an order of magnitude smaller than over the complete 24 ps. This algorithm and timestep also give no temperature drift. The measured potential energy (−41.8 MPa) are in very close agreement with published rsults [13] for the same

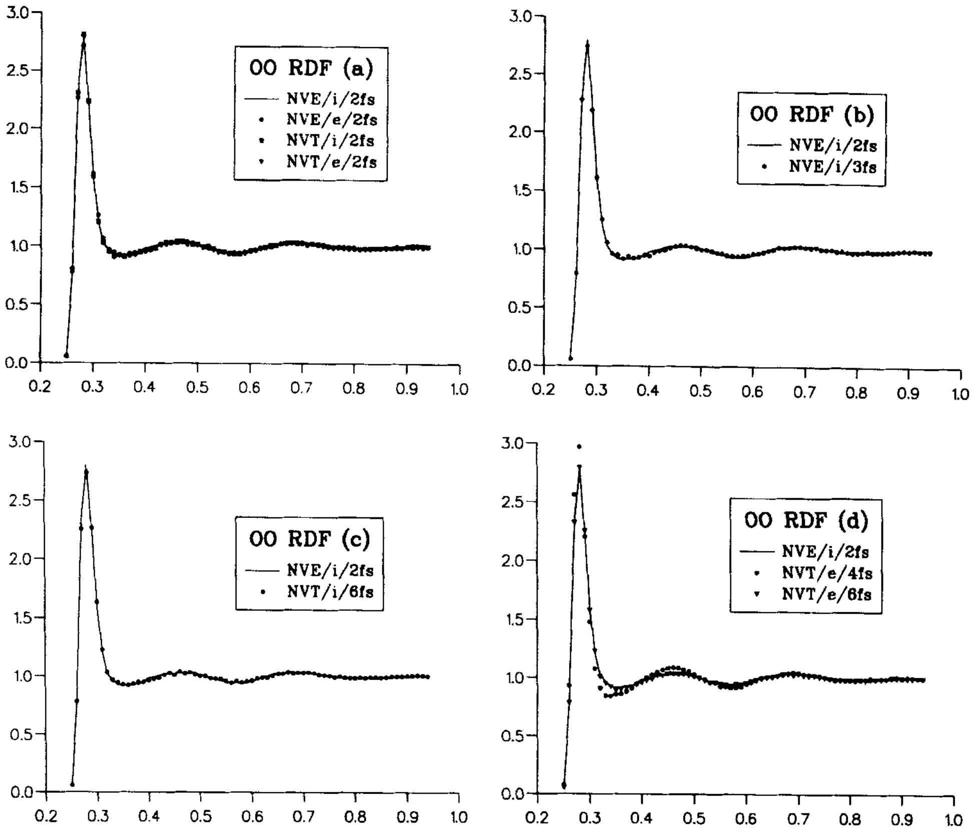

Figure 1 Oxygen-oxygen radial distribution function, as a function of distance in nm. (a) Implicit, "i", and explicit, "e", NVE and NVT algorithms with a 2 fs time step. (b) Implicit NVE algorithm with a 3 fs time step. (c) Implicit NVT algorithm with a 6 fs time step. (d) Explicit NVT algorithm with 4 fs and 6 fs time steps. In this and subsequent figures the implicit NVE benchmark with a 2 fs time step always appears as a solid line.

Table 3 Diffusion constants estimated from the integral of the VACF to 0.6 ps Units are $10^{-9} m^2/s$.

Δ/fs	*Implicit*	*Explicit*
	NVE dynamics	
2	4.02	4.98
3	4.67	7.60
	NVT dynamics	
2	3.80	4.60
3	4.37	3.32
4	4.27	3.18
6	4.45	4.82

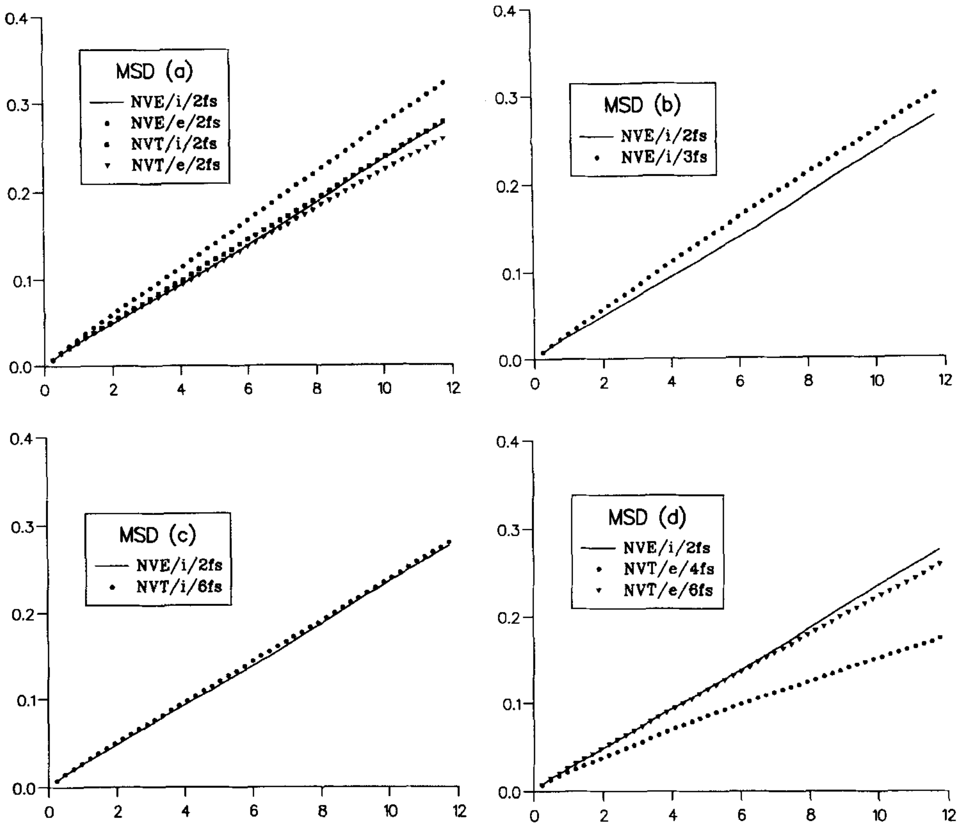

Figure 2 Mean-square displacements of the molecules in nm^2 as a function of time in ps. Other details as Figure 1.

model and state point. I take this simulation to be a benchmark against which other algorithms and time steps are to be compared.

With the explicit NVE algorithm at 2 fs time step the energy conservation is much worse, with a total energy drift of 2 parts in 10^3 per ps, or 5% over the whole run, and

Table 4 Diffusion constants estimated from linear fits to the MSD from 6 to 12 ps. Units are 10^{-9}m^2/s.

Δ/fs	*Implicit*	*Explicit*
	NVE dynamics	
2	4.03	4.47
3	4.23	6.30
	NVT dynamics	
2	3.88	3.52
3	3.62	3.15
4	3.80	2.18
6	3.95	3.55

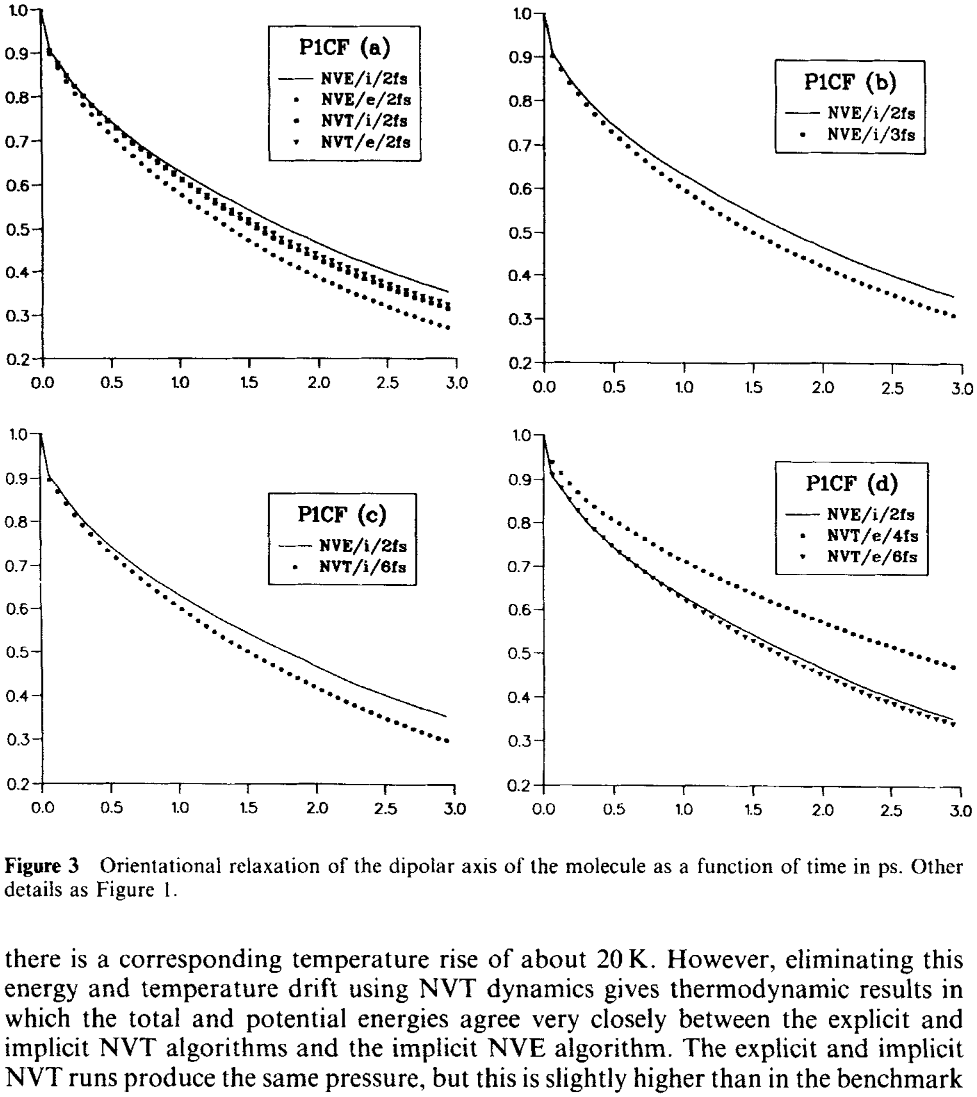

Figure 3 Orientational relaxation of the dipolar axis of the molecule as a function of time in ps. Other details as Figure 1.

there is a corresponding temperature rise of about 20 K. However, eliminating this energy and temperature drift using NVT dynamics gives thermodynamic results in which the total and potential energies agree very closely between the explicit and implicit NVT algorithms and the implicit NVE algorithm. The explicit and implicit NVT runs produce the same pressure, but this is slightly higher than in the benchmark NVE run.

Figures 1(a)–3(a) show the correlation functions for the four 2 fs runs. There is no detectable difference in the RDFs. The VACFs are very close, but the explicit NVE is less negative in the tail and this leads to a higher diffusion constant (Tables 3 and 4). This effect is also seen in the MSD and of course is expected since this run shows an upward temperature drift. Diffusion constants for the other three algorithms agree within about 10%. The angular momentum correlation functions for the four runs are identical. The orientational relaxation functions are close but not identical, with the graph of the benchmark run lying above the other three.

With a 3 fs timestep the explicit NVE algorithm is hopeless with a large temperature drift, but the implicit NVE is satisfactory, with negligible temperature drift and energies agreeing with the benchmark run to within a percent or two. The pressure is higher, but this quantity is very sensitive to small errors. The correlation functions for the implicit NVE 3 fs run are compared with the benchmark run in Figures 1(b)–3(b). The agreement in the structure is good. The VACF is less negative in the tail giving a diffusion integral about 15% higher (Table 3) but this effect is less evident in the MSD (Figure 2(b) and Table 4).

There is little point in pursuing the NVE algorithms to time steps larger than 3 fs as temperature drifts become evident. What about the NVT algorithms? Turning first to the implicit algorithm, it is clear that even with the very large time step of 6 fs, thermodynamic results agree with the benchmark within a percent or two. Correlation functions are shown in Figures 1(c)–3(c). The RDFs are practically identical with the benchmark, as are the VACF, MSD and diffusion constant. The orientational correlation function is the only one to show a systematic discrepancy, though this is not large.

From the tables, and Figures 1(d)–3(d), it is clear that the explicit NVT algorithm with 3 fs and 4 fs time steps has produced an anomalously ordered state of water. Compared with the normal state it has a lower energy, lower torques, more structured RDFs, more rapidly decaying angular momentum auto-correlation function, slower diffusion and slower orientational relaxation. Heating this system to 350 K for 2.4 ps, followed by re-equilibration to 298 K for 2.4 ps, produced a normal state which lasted for 10 ps before the anomalous ordering re-appeared. As far as I am aware this is the first time that algorithm errors have been observed to produce such an effect. Work is underway to investigate further the nature of this state and the mechanism producing it. It is very curious that this same algorithm with a 6 fs timestep does not produce such an anomalous state, and nor does the implicit algorithm with any time step.

4 CONCLUSIONS

The new implicit quaternion leapfrog algorithm introduced in this paper is superior to the explicit algorithm for the integration of the rotational motion of the rigid water molecule, which provides a severe test of any algorithm. Being implicit, the algorithm involves an iterative solution, but converges rapidly. For SPC water it can be used with a 3 fs time step, but at larger time steps energy and temperature drifts become apparent. If a thermostat is introduced much larger time steps can be used. The largest time step studied was 6 fs, and the only noticeable, if small, discrepancies from the 2 fs run were in the pressure and the relaxation of the dipole axis. The explicit algorithm, when combined with a temperature constraint, produced an anomalously ordered state of water with a 3 fs or 4 fs time sep, but not with a 2 fs or 6 fs step.

Acknowledgement

I would like to thank G. Kroes for a helpful correspondence, W. Smith for useful discussions, and S. Toxvaerd for sending me reference [10] prior to publication.

References

[1] D.J. Evans, "On the representation of orientation space", *Mol. Phys.*, **34**, 317–25 (1977).

[2] D. Fincham, "Choice of timestep in molecular dynamics simulation", *Comput. Phys. Commun.*, **40**, 263–9 (1986).

[3] D. Fincham, "An algorithm for the rotational motion of rigid molecules", *CCP5 Information*

Quarterly, **2**, 6–10 (1981). (This informal publication can be obtained on request from SERC Daresbury Laboratory, Warrington WA4 4AD, U.K.)

[4] D. Fincham and D.M. Heyes, "Recent advances in molecular dynamics computer simulation", in *Dynamical processes in condensed matter*, M.W. Evans, ed., John Wiley, New York (1985).

[5] M.P. Allen and D.J. Tildesley, *Computer simulation of liquids*, Oxford University Press (1986).

[6] R. Ahlrichs and S. Brode, "A new rigid motion algorithm for MD simulations", *Comput. Phys. Commun.*, **42**, 59–64 (1986).

[7] M. Amini and D. Fincham, "Evaluation of temperature in molecular dynamics simulation", *Comput. Phys. Commun.*, **56**, 313–24 (1990).

[8] 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**, 1243–52 (1984).

[9] 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**, 3684–90 (1984).

[10] S. Toxvaerd, "Algorithms for canonical molecular dynamics simulations", *Molec. Phys.* **72**, 159–68 (1991).

[11] W.G. Hoover, "Canonical dynamics: equilibrium phase space distributions", *Phys. Rev.*, **A31**, 1695–7 (1985).

[12] H.J.C. Berendsen, J.P.M. Postma, W.F. van Gunsteren and J. Hermans, "Interaction models for water in relation to protein hydration", in *Intermolecular forces*, B. Pullman, ed., D. Reidel, Dordrecht (1981).

[13] K. Watanabe and M.L. Klein, "Effective pair potentials and the properties of water", *Chem. Phys.*, **131**, 157–67 (1989).