UNIVERSITY OF CAMBRIDGE

Numerical Analysis Reports

A Symplectic Method for Rigid-Body Molecular Simulation

A. Kol, B. Laird, & B. Leimkuhler

DAMTP 1997/NA5
March 1997

Department of Applied Mathematics and Theoretical Physics
Silver Street
Cambridge
England CB3 9EW

A Symplectic Method for Rigid-Body Molecular Simulation

Ayla Kol[1] and Brian B. Laird[2]
Department of Chemistry, University of Kansas, Lawrence, KS 66045

Benedict J. Leimkuhler[3]
Department of Mathematics, University of Kansas, Lawrence, KS 66045
(April 1, 1997)

Abstract

Rigid body MD simulations typically are performed in a quaternion representation. The nonseparable form of the Hamiltonian in quaternions prevents the use of a standard leap-frog (Verlet) integrator, so nonsymplectic Runge-Kutta, multistep or extrapolation methods are generally used. This is unfortunate since symplectic methods like Verlet exhibit superior energy conservation in long time integrations. In this article, we describe an alternative method, RSHAKE, in which the entire rotation matrix is evolved (using the scheme of McLachlan and Scovel [J. Nonlin. Sci., **16** 233 (1995)]) in tandem with the particle positions. We employ a fast approximate Newton solver to preserve the orthogonality of the rotation matrix. We test our method on a system of soft-sphere dipoles and compare with quaternion evolution using a 4th-order predictor-corrector integrator. Although the short-time error of the quaternion algorithm is smaller for fixed time step than that for RSHAKE, the quaternion scheme exhibits an energy drift which is not observed in simulations with RSHAKE, hence a fixed energy tolerance can be achieved by using a larger time step. The superiority of RSHAKE increases with system size.

I. INTRODUCTION

An important problem in molecular simulation is the development of stable and efficient algorithms for integrating the equations of motion of orientational degrees of freedom. The straightforward parameterization for such degrees of freedom, Euler angles [4], is at a severe disadvantage for numerical work because of the singularities inherent in the description. To overcome this difficulty, a parameterization based on quaternions was proposed by Evans [5] and has become standard [6]. A significant drawback in this approach arises from the fact that the rigid-body Hamiltonian is nonseparable in the quaternion representation, therefore preventing the use of the popular leap-frog (Verlet) algorithm for integrating the rotational equations of motion. Usually, then, higher order methods such as Gear predictor-corrector methods are utilized. The superiority of the Verlet scheme arises from the fact that it is, like the true Hamiltonian dynamics, *symplectic*; that is, it preserves the wedge product $dr \wedge dp$ (p is the momentum congugate to r) [7,8]. A more familiar, but weaker property of symplectic maps is that the flow preserves volumes in phase space. Such symplectic integration algorithms have been shown to posses excellent long-term energy stability, often far superior to non-symplectic methods (even those with higher-order local error). For a recent example showing the improved long-term stability of integration methods incorporating geometric structure, see [9].

For systems of rigid molecules in which the potential is easily written in terms of intermolecular interations between atomic sites on the molecules, it is possible to circumvent the parameterization of orientational degrees of freedom by considering the fundamental variables to be the individual Cartesian coordinates of the atomic sites. The dynamics are determined by integrating the equations of motion for these sites, subject to the contraint that the molecule remains rigid (intramolecular bond distances and angles are fixed). The constraints can be implemented using appropriate Lagrange multipliers. A generalization of the Verlet integration scheme (SHAKE algorithm [12]) is then used to integrate the constrained equations of motion. SHAKE is algebraically equivalent to the RATTLE discretization of [10], and the latter scheme is formally symplectic along the manifold of constraints [11]. This fact helps to explain SHAKE's excellent long time stability. The use of SHAKE on a constrained particle description becomes inconvenient for general rigid body integration for two reasons. First, as the number of atoms in each rigid molecule increases the number of constraints increases dramatically, which decreases the efficiency of the computation. Second, additional computational complexity is introduced in the force computations for intermolecular potentials that are not easily decomposable into direct site-site interactions such as potentials expressed as multi-pole expansions.

In this paper, we develop an algorithm in which the rotation matrix is not parameterized, but is evolved directly. The structure of the equations of motion is such that a SHAKE/RATTLE scheme is possible, except that, instead of constraining the bond lengths (or angles), we constrain the rotation matrix to be orthogonal (cf. [14,15]). We also describe efficient iteration procedures for the nonlinear equations that must be solved at each timestep. We will refer to this approach as RSHAKE for Rotation SHAKE. The advantages of this model over SHAKE with bond contraints are that the number of constraints does not increase with molecular size and that it is well suited for non-site-to-site interactions such as those generated from multipole expansions.

II. SIMULATION OF RIGID BODY MOTION

We consider a collection of N interacting rigid bodies. The time evolution of a rigid body can be studied by considering seperately the translational motion of the center of mass and the rotations about the principal axes (which pass through the center of mass). The total energy, E, for such a system is

$$E = \sum_i \frac{M_i \mathbf{v}_i^T \mathbf{v}_i}{2} + \sum_i \frac{\mathbf{\Omega}_i^T I_i \mathbf{\Omega}_i}{2} + \Phi(\{\mathbf{r}_i\}, \{\theta_i\}) , \tag{1}$$

where $\mathbf{r}_i$, $\mathbf{\Omega}_i$ and I_i are the Cartesian position vector of the center of mass, the (body-frame) angular velocity and the moment-of-inertia tensor (also in the body frame) of particle i, respectively, Φ is the potential energy function and θ_i is a representation of the orientation of particle i. The resulting equations of motion for the translational coordinates are

$$\ddot{\mathbf{r}}_i = -\nabla_{\mathbf{r}_i} \Phi \tag{2}$$

Numerically, these can be integrated easily using the standard leap-frog (Verlet) algorithm

$$\mathbf{v}_{i,n+1/2} = \mathbf{v}_{i,n-1/2} - \frac{h}{M_i} \nabla_{\mathbf{r}_i} \Phi_n \tag{3}$$

$$\mathbf{r}_{i,n+1} = \mathbf{r}_{i,n} + h\, \mathbf{v}_{i,n+1/2} , \tag{4}$$

where we use the common convention that $\mathbf{r}_n$ denotes the numerical approximation at time $t_n = hn$ to the value $\mathbf{r}(t_n)$ with h being the fixed step size. The translational velocity defined by $\mathbf{v}_i = \dot{\mathbf{r}}_i$ is approximated at mesh points as

$$\mathbf{v}_{i,n} = \frac{\mathbf{v}_{i,n+1/2} - \mathbf{v}_{i,n-1/2}}{h} \tag{5}$$

This algorithm is second order and, more importantly, it is also symplectic when formulated in canonical variables [16].

To determine the rotational motion one needs to make use of two different reference frames: a space-fixed frame (also called the laboratory-fixed frame) and a body-fixed frame. Forces and torques are more conveniently calculated in the laboratory frame whereas the rotational equations of motion are simpler in the body frame. We will use the convention that small letters stand for the representations of variables in the fixed laboratory frame and their counterparts in the body frame will be denoted by capital letters. With this convention we have

$$\mathbf{r} = \mathcal{Q}\mathbf{R} \tag{6}$$

where $\mathbf{r}$ and $\mathbf{R}$ are vectors in the laboratory and body frames, respectively, and $\mathcal{Q}$ is the 3×3 time-dependent rotation matrix subject to the orthogonality condition

$$\mathcal{Q}\mathcal{Q}^T = 1 = \mathcal{Q}^T \mathcal{Q} . \tag{7}$$

A consequence of this condition is that only three independent parameters are necessery to describe the nine elements of the rotation matrix.

One standard parametrization of $\mathcal{Q}$ is in terms of the well-known Euler angles [4], but the singularities inherent in this description make it unsuitable for numerical work. Another is the quaternion parametrization of Hamilton, where the orthogonality condition is again explicitly implemented utilizing four quaternions instead of the three Euler angles. The quaternion method and its implementation is outlined in the next section.

In Section IV we present a rotational evolution algorithm for molecular simulations (RSHAKE) that is both explicit and algebraically equivalent to a symplectic method. The method is compared in Section VI for a system of dipolar soft spheres (described in Section V) with a quaternions method using a fourth-order (Gear) predictor corrector integrator.

III. QUATERNION METHOD

In the quaternion representation [5], the orientation of a rigid body is parameterized in terms of a set of four scalar quantities, $(q_0, q_1, q_2, q_3) \equiv \mathbf{q}$, with the condition

$$q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1 \; . \tag{8}$$

In terms of the standard Euler angles, θ, ϕ and ψ, [4,6], the quaternion parameters are

$$\begin{aligned} q_0 &= \cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\phi+\psi}{2}\right) \\ q_1 &= \sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\phi-\psi}{2}\right) \\ q_2 &= \sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\phi-\psi}{2}\right) \\ q_3 &= \cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\phi+\psi}{2}\right) \; . \end{aligned} \tag{9}$$

and the rotation matrix is given by

$$\mathcal{Q} = \begin{pmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & q_0^2-q_1^2+q_2^2-q_3^2 & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & q_0^2-q_1^2-q_2^2+q_3^2 \end{pmatrix} \tag{10}$$

where the normalization condition (8) ensures the orthogonality of $\mathcal{Q}$. The introduction of these four quaternion parameters (plus a constraint) in place of the 3 Euler angles removes the singularities of the Euler angle representation [4].

Then in the quaternion representation, the equations of motion for the angular velocities augmented with the time derivative of the constraint (8) on the quaternions results in

$$\frac{d\mathbf{q}}{dt} = \mathcal{A}\boldsymbol{\Omega} \; , \tag{11}$$

where

$$\mathcal{A} = \frac{1}{2}\begin{pmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{pmatrix} \tag{12}$$

and

$$\mathbf{\Omega} = \begin{pmatrix} 0 \\ \Omega_x \\ \Omega_y \\ \Omega_z \end{pmatrix} \tag{13}$$

is the angular velocity in body-frame. The time derivative of the angular velocity is related to the torque $\mathcal{T}$ and the principal moments of inertia in the body-frame by

$$\begin{aligned} \dot{\Omega}_x &= \frac{\mathcal{T}_x}{I_{xx}} + \left(\frac{I_{yy} - I_{zz}}{I_{xx}}\right)\Omega_y\Omega_z \\ \dot{\Omega}_y &= \frac{\mathcal{T}_y}{I_{yy}} + \left(\frac{I_{zz} - I_{xx}}{I_{yy}}\right)\Omega_z\Omega_x \\ \dot{\Omega}_z &= \frac{\mathcal{T}_z}{I_{zz}} + \left(\frac{I_{xx} - I_{yy}}{I_{zz}}\right)\Omega_x\Omega_y \end{aligned} \tag{14}$$

The above seven coupled equations can be integrated numerically using a variety of methods. The standard leap-frog (Verlet) algorithm cannot be applied since the time-derivatives of quaternions depend not only on the angular velocity Ω, but on the quaternions themselves through $\mathcal{A}$; the Hamiltonian is nonseparable in the quaternion representation. Generally, higher-order nonsymplectic algorithms such as the Gear predictor-corrector method (see Appendix E of reference [6]) are used.

Recently Fincham [13] has devised an implicit leap-frog-"like" integration algorithm for quaternions. His method is based on (11) and the equation of motion for the angular momentum

$$\frac{d\,\boldsymbol{\ell}}{dt} = \mathbf{t}, \tag{15}$$

where $\boldsymbol{\ell}$ is the three dimensional angular momentum vector in the laboratory frame and $\mathbf{t}$ is the torque in the same coordinate system. The implicit algorithm of Fincham then proceeds as follows

$$\boldsymbol{\ell}_{n+1/2} = \boldsymbol{\ell}_{n-1/2} + h\mathbf{t}_n \tag{16}$$

$$\mathbf{q}_{n+1} = \mathbf{q}_n + \frac{h}{2}(\mathcal{A}_n\mathbf{\Omega}_n + \mathcal{A}_{n+1}\mathbf{\Omega}_{n+1})\ . \tag{17}$$

The last equation is an implicit one defined through the auxilary parameters $\boldsymbol{\ell}_n = \boldsymbol{\ell}_{n-1/2} + h\mathbf{t}_n$ and $\boldsymbol{\ell}_{n+1} = \boldsymbol{\ell}_n + h\mathbf{t}_n$ which are not stored, but used to calculate the angular velocities in the body frame

$$\boldsymbol{\Omega}_n = \begin{pmatrix} 0 \\ I^{-1}\mathcal{Q}_n^T\boldsymbol{\ell}_n \end{pmatrix} . \tag{18}$$

In Section VI we compare Fincham's implicit method and RSHAKE on a model system.

IV. THE RSHAKE METHOD

In RSHAKE, the rotation matrix $\mathcal{Q}$ is evolved directly, as opposed to evolving a set of parameters (Euler angles or quaternions) used to represent $\mathcal{Q}$. In doing so, we use a scheme similar to that of McLachlan and Scovel [14] (see also Reich [15]) in which one defines a "momentum" that is canonically conjugate to $\mathcal{Q}$ - the resulting Hamilton's equations of motion subject to the orthogonality constraint $\mathcal{Q}^T\mathcal{Q} = 1$ are then of the proper form for the implementation of proveably symplectic integrators, such as SHAKE/RATTLE. To determine the equations of motion, the rotational kinetic energy must be expressed in terms of $\mathcal{Q}$. This is done by considering the molecule to be made up of discrete atoms of mass m_j centered at position $\mathbf{R}_j$ in the body-fixed frame. (The total mass of the molecule is then $M = \sum_j m_j$.) The rotational kinetic energy of the molecule, K_{rot}, is then given by

$$K_{rot} = \frac{1}{2}\sum_i m_i \mathbf{v}_i \cdot \mathbf{v}_i = \frac{1}{2}\sum_i m_i Tr[\dot{\mathcal{Q}}\mathbf{R}_i\mathbf{R}_i\dot{\mathcal{Q}}^T] , \tag{19}$$

since $\mathbf{v}_i = \dot{\mathcal{Q}}\mathbf{R}_i$. Defining a body-frame tensor

$$J \equiv \sum_i m_i \mathbf{R}_i\mathbf{R}_i , \tag{20}$$

gives

$$K_{rot} = \frac{1}{2}Tr[\dot{\mathcal{Q}}J\dot{\mathcal{Q}}^T] . \tag{21}$$

For a continuous mass distribution described by a mass density $\rho(\mathbf{R})$, we have

$$J \equiv \int d\mathbf{R}\rho(\mathbf{R})\mathbf{R}\mathbf{R} . \tag{22}$$

Note that, J differs slightly from the moment-of-inertia tensor I, so that if, as usual, the body frame is chosen to be such that J (and therefore I) is diagonal we have

$$I_{\alpha\alpha} = J_{\beta\beta} + J_{\gamma\gamma} , \tag{23}$$

where (α, β, γ) is a cyclic permutation of $(1, 2, 3)$.

In terms of $\mathcal{Q}$, the Lagrangian for a *single* rigid body with center of mass position vector $\mathbf{r}_{cm}$ is then

$$\mathcal{L} = \frac{M}{2}\dot{\mathbf{r}}_{cm} \cdot \dot{\mathbf{r}}_{cm} + \frac{1}{2}Tr[\dot{\mathcal{Q}}J\dot{\mathcal{Q}}^T] - \Phi(\mathcal{Q}, \mathbf{r}_{cm}) + Tr[\lambda(\mathcal{Q}^T\mathcal{Q} - 1)] \tag{24}$$

The first and the third terms are the usual translational kinetic energy and the total potential energy. The last term, involving the Lagrange multiplier matrix λ, has been added to ensure that the equations of motion are such that the rotation matrix remains orthogonal. λ is a symmetric matrix since it enforces a constraint on a matrix which is symmetric.

To construct the rotational equations of motion, we define the momentum Π congugate to $\mathcal{Q}$ in the usual way

$$\Pi \equiv \frac{\partial \mathcal{L}}{\partial \dot{\mathcal{Q}}} = \dot{\mathcal{Q}}J \; . \tag{25}$$

(Like $\mathcal{Q}$, Π is a 3×3 matrix.) The rotational kinetic energy becomes

$$K_{rot} = \frac{1}{2}Tr[\Pi J^{-1}\Pi^T] \; . \tag{26}$$

Hamilton's equations for the rotational motion are then

$$\dot{\mathcal{Q}} = \Pi J^{-1} \tag{27}$$
$$\dot{\Pi} = -\partial_{\mathcal{Q}}\Phi + 2\mathcal{Q}\lambda \tag{28}$$
$$g(\mathcal{Q}) = \mathcal{Q}^T\mathcal{Q} - 1 = 0 \; , \tag{29}$$

The structure of these equations is such that a SHAKE/RATTLE integration algorithm is applicable:

$$\Pi_{n+1/2} = \Pi_{n-1/2} + h[2\mathcal{Q}_n\lambda_n - \partial_{\mathcal{Q}_n}\Phi] \tag{30}$$
$$\mathcal{Q}_{n+1} = \mathcal{Q}_n + h\Pi_{n+1/2}J^{-1} \tag{31}$$
$$g(\mathcal{Q}_{n+1}) = 0 \; . \tag{32}$$

Equations 30- 32 define RSHAKE (Rotational SHAKE). We now have $2 \times 9 = 18$ parameters to evolve instead of the seven for quaternions, plus the added work to solve for the six independent elements of λ. A discussion of a variety of methods for solving the nonlinear constraint equations is included in Appendix A. If the constraint is iterated to convergence the local error occuring after one step with the RSHAKE integrator is $O(h^3)$. (Note that, for planar and linear molecules the matrix J is not invertible. For these cases, minor modifications in the equations of motion and the RSHAKE algorithm are required. These are discussed in Appendix B.)

It should be noted that a method for propagating the rotation matrix directly was presented by Ahlrichs and Brode [17] - the relation to the present work is not immediately apparent, since the notations differ substantially. The two methods differ, primarily, in the way the orthogonality constraint is enforced. The method of Ahlrich and Brode does not, as in RSHAKE, solve the constraint equation to convergence, but instead writes the rotation matrix as an exponential of some anti-symmetric matrix A and, using information about A from the last step determines the new value of A to third-order in h by expanding the equations of motion. The new A is then exponentiated to give the new rotation matrix - which is guaranteed to be orthogonal due to the anti-symmetry property of A. From the point of view of the present paper, this procedure has two principal drawbacks: First, a matrix exponentiation is necessary at each step (and for each molecule); even though the exponential of a 3x3 matrix can be computed fairly efficiently using Rodrigues' formula, this step can be expensive relative to a matrix factorization. Second, and more important, since the Lagrange multiplier matrix is only being solved to third-order in h, the method will not be symplectic.

V. MODEL AND IMPLEMENTATION

We have applied the algorithms outlined in the previous sections to a system of N dipolar soft spheres (DSS). This system has been the subject of earlier simulations using the quaternion method [18,19]. The DSS interact with two-body potential of the form

$$U = U_S + U_D \ , \tag{33}$$

where

$$U_S = 4\epsilon \left(\frac{\sigma}{r_{ij}}\right)^{12} \ , \tag{34}$$

and

$$U_D = \left(\frac{1}{r_{ij}^3}\right) \left(\boldsymbol{\mu}_i \cdot \boldsymbol{\mu}_j - 3(\boldsymbol{\mu}_i \cdot \hat{\mathbf{r}}_{ij})(\boldsymbol{\mu}_j \cdot \hat{\mathbf{r}}_{ij})\right) \ , \tag{35}$$

where ϵ and σ are energy and length scales, $\boldsymbol{\mu}$ are the dipole moments of particles, and $\mathbf{r}_{ij}$ is the seperation of particles i and j. For a given particle i the dipole moment can be written as

$$\boldsymbol{\mu}_i(t) = Q_i(t)\boldsymbol{\mu}_{ref} \ , \tag{36}$$

where we have taken all the reference dipoles parallel to $\hat{\mathbf{z}}$. Both potentials generate a force on the centers of mass, while only the dipolar interaction results in a torque. For the purposes of the present simulations we have truncated the above potentials at a distance $r_c = 2.5\sigma$. A shifting function was then added to ensure that both the potentials and the forces approach zero continuously as r_c is approached. For the short ranged part of the potential (U_S) we have added a linear term of the form

$$U_S \Longrightarrow U_S + Ar_{ij} + B \tag{37}$$

For the dipole-dipole part of the potential we shift only the radial part of the potential, as follows

$$\frac{1}{r_{ij}^3} \Longrightarrow \frac{1}{r_{ij}^3} + ar_{ij}^4 + b \, . \tag{38}$$

The constants A, B, a and b are determined from the continuity condition.

All simulations were carried at a density of $\rho\sigma^3$=0.5 with a dipolar strength $\mu^{*2} = \mu^2/(\epsilon\sigma^3) = 2$. Initially, all the particles were placed on an fcc lattice. The initial dipole orientations randomly distributed on the unit sphere. To implement the quaternion method, the rotational velocities and their time derivatives as well as the time derivatives of the quaternions were taken to be zero initially. In RSHAKE, we took $\Pi_{-1/2} = 0$ for all the particles. Cubic periodic boundary conditions were used. All particles were assumed to be spherical using a diagonaly body-frame moment-of-inertia tensor with $I_{xx} = I_{yy} = I_{zz} = 0.025M\sigma^2$, where M is the mass of each particle. For spherical particles, the J matrix (22) reduces to $\frac{1}{2}I$.

VI. RESULTS AND DISCUSSION

To compare RSHAKE with the standard quaternion method (using a 4th-order Gear predictor/corrector integrator), we have performed simulations on the DSS system using both methods for a variety of system sizes (N = 108,256,500, 864), and time steps ranging from 0.001 to $0.010t_0$ where $t_0 = \sqrt{m\sigma^2/\epsilon}$. (For SPC water, the time scale t_0 will be around 2ps) All reported results are for a fixed total time of $t^* = t/t_0 = 252$. All quoted CPU times are for an IBM RS6000 workstation (Model 43P).

In our analysis, we define two measures of energy error: global energy drift, ϵ_g and local energy fluctuations ϵ_l. Both are determined from a plot of the instantaneous total energy (per particle) as function of time over the entire length of the simulation. Global energy drift is defined as the slope of this curve (determined by a least-squares fit) multiplied by the duration of the simulation. Local fluctuation error is defined as the standard deviation energy about the above linear fit. In Figure 1 we show energy-versus-time plots for a variety of time steps (h^* = 0.002, 0.004, 0.006, 0.008, 0.010) for a DSS system of 500 particles. Results for both quaternions and RSHAKE are shown. (The plots for the different h^* values have been shifted for clarity.) From this Figure it is clear that global energy drift for the quaternion method increases rapidly with time step and always dominates the local energy fluctuation for the method. In constrast, the global error for RSHAKE is negligible - being always smaller than the local fluctuations for the method. The local fluctuations in RSHAKE are always larger than that for the quaternion method, due largely to the use of a higher-order (Gear) integrator.

In Table I is shown the derived global energy drift and local energy fluctuations for both methods for two systems - one small (N = 256) and one large (N = 864) - for a variety of time steps. The CPU time (in seconds) for each run is also reported. Although, for a given time step, the quaternion method is always faster than RSHAKE, the global energy drift of the non-symplectic quaternion integrator is so large that, for fixed energy error tolerence and simulation length, RSHAKE allows for a much greater time step to be used. To illustrate this, we show in Figure 2 a log-log plot of ϵ_l and ϵ_g for both RSHAKE and our implementation of quaternions. This shows that, for example, for a fixed total error tolerence of 1 part in 10^{-4}, in our runs a time step of just above $0.002t_0$ would be required for quaternions where as $0.007t_0$ would be sufficient for RSHAKE - for larger tolerences the difference is even greater! The slopes of the curves in Figure 2 give the order of the error measurement for each method. The slopes for the local fluctuations of each method are equal to about 2, indicating an $O(h^2)$ method - this makes since despite the fact that the integrator used for the quaternions is fourth order, since the integrator for the translational degrees of freedom is the same in both methods and is second-order in the time step. The slopes of the global energy-drift curves for RSHAKE and quaternions are about 2 and 3, respectively - the origin of this difference is unclear. To view the same data in a slighly different way for the 500 particle simulations, we plot in Figure 3 the dominant error for each method (global for quaternions and local for RSHAKE) as a function of CPU time (in seconds) for a fixed length run of $252t_0$ - the different points correspond to different time steps. Again, it can be clearly seen that, for fixed energy tolerence, RSHAKE is superior to the standard implementation of quaternions.

To examine the size dependence of our results, we show in Figure 4 a plot of CPU time versus particle number N for our two methods. The time steps chosen (0.005 and $0.002t_0$ for RSHAKE and quaternions, respectively) give comparable total energy errors for our $252t_0$ length runs of the 500 particle system (see Figure 2). In addition to total CPU time , the amount of time spent in force (and torque) evaluations is plotted for both methods, along with the amount of time required in to RSHAKE to solve for the Lagrange multiplier matrix. For all N, the RSHAKE method remains the faster algorithm. Also note that, since the Lagrange multiplier determination step scales only linearly with N - as opposed to the N^2 scaling of the force evaluations, this step will become much less impertant for large system sizes, further increasing the effiency of RSHAKE. It should also be noted that the local energy fluctuations in the energy per particle will scale as $N^{-1/2}$, thereby decreasing with system size, whereas the global energy drift appears to be relatively size independent. Therefore the total error of RSHAKE, being dominated by local fluctuations, will decrease as larger systems are considered, but that for quaternions will remain nearly constant. This is illustrated for our system in Figure 5 which shows the dominant error for RSHAKE ($h^* = 0.006$) and quaternions ($h^* = 0.003$) over our $252t_0$ runs.

We have also studied our system using the implicit method of Fincham for integrating the quaternion equations of motion (Section III). Our results show that this method has a significantly larger energy drift for all studied time steps than when the predictor-corrector is used. For example for a 500 particle system with time step $h^* = 0.002$, the global energy error in Fincham's method is $1.1d - 3$ in reduced units, compared to 6.3d-5 for predictor-corrector. In a recent paper [20], Fincham compared results for a linear molecule using his implicit method and several other methods including his LEN algorithm, which is shown in Appendix B to be equivalent to RSHAKE applied to a linear molecule. Fincham concluded that LEN (RSHAKE) was superior to his implicit quaternion integrator. Thus, concluding that Fincham's implicit method doesn't compare well, we have performed our detailed comparisons only between quaternions and RSHAKE.

VII. SUMMARY

We have developed a new algorithm, RSHAKE, for the numerical integration of the equations of motion for the orientational degrees of freedom of rigid molecules. Unlike other standard methods such as quaternions in which the rotation matrix is parameterized, the entire rotation matrix is treated in RSHAKE as a dynamical variable with corresponding conjugate momentum. The resulting equations of motion for the rotation matrix must then be solved subject to the contstraint that the matrix remain orthogonal. In analogy with the standard SHAKE algorithm for molecules with bond constraints, RSHAKE uses a Verlet (leap-frog) integration scheme while simultaneously solving the constraint equations exactly. Like SHAKE and true Hamiltonian dynamics, RSHAKE can be shown to be fully symplectic - a property that gives the method superior stability with respect to energy conservation. For a system of soft-sphere dipoles we have demonstrated that RSHAKE is superior to quaterion-based integration schemes using both a 4th-order (Gear) predictor-corrector method and an explicit leap-frog-like algorithm due to Fincham [13].

Acknowledgements

This research was supported by National Science Foundation Grants CHE-950281 (BBL and AK) and DMS-9303223 (BJL). The authors also thank the Kansas Insitute for Theoretical and Computational Science and the Department of Applied Mathematics and Theoretical Physics of Cambridge University for providing stimulating research environments during this project.

Appendix A: Solution of the Nonlinear Equations

We can simplify the equations (30)-(31) to

$$\mathcal{Q}_{n+1} = \hat{\mathcal{Q}}_{n+1} + 2h^2 \mathcal{Q}_n \lambda_n J^{-1}. \tag{39}$$

where $\hat{\mathcal{Q}}_{n+1}$ represents the unconstrained step in Q:

$$\hat{\mathcal{Q}}_{n+1} = \mathcal{Q}_n + h\Pi_{n-1/2} J^{-1} - h^2 \partial_{\mathcal{Q}_n} \Phi J^{-1}. \tag{40}$$

Substituting (39) into (32) and multiplying on either side by J results in a quadratic (matrix Ricatti) equation for the six independent elements of the (symmetric) λ matrix:

$$F(\lambda_n) = \alpha \lambda_n^2 + \lambda_n B + B^T \lambda_n + C = 0, \tag{41}$$

where

$$\alpha = 2h^2, \qquad B = \mathcal{Q}_n^T \hat{\mathcal{Q}}_{n+1} J, \qquad C = \frac{1}{2h^2} J(\hat{\mathcal{Q}}_{n+1}^T \hat{\mathcal{Q}}_{n+1} - I)J.$$

Observe that from (40), and the fact that λ is bounded as $h \to 0$, the matrix C also remains bounded as $h \to 0$.

Following the standard approach, we localize the solution of this nonlinear equation by assuming that we have some initial guess $\bar{\lambda}$ (such as λ_{n-1}) near to the solution λ_n. We compute an improved approximation by linearizing the equation

$$F(\bar{\lambda} + \Delta) = 0 \, . \tag{42}$$

and solving for Δ. This gives Δ as the solution of the matrix Lyapunov equation

$$K^T \Delta + \Delta K = -F(\bar{\lambda}), \qquad K = \alpha \bar{\lambda} + B.$$

We then obtain a corrected value

$$\hat{\lambda} = \bar{\lambda} + \Delta.$$

The iteration of this process is just the Newton method, which, started from an initial guess $\lambda^{(0)}$ computes a sequence of approximations $\lambda^{(1)}, \lambda^{(2)}, \ldots$ to λ_n recursively from

$$\lambda^{(m)} = \lambda^{(m-1)} + \Delta^{(m)},$$

where

$$K_m^T \Delta^{(m)} + \Delta^{(m)} K_m = -F(\lambda^{(m-1)}), \qquad K_m = \alpha\lambda^{(m-1)} + B. \tag{43}$$

Although there is a substantial body of literature on the efficient solution of matrix Riccati and Lyapunov equations, this research is primarily oriented to the treatment of large-dimensional problems. In our case, we need to solve many small-dimensional decoupled problems. For such small dimensional problems, the straightforward methods are probably optimal. The simplest approach is to write (43) as a 6-dimensional vector equation, requiring the factorization of a 6x6 matrix at each Newton iteration step.

There are several possible ways to reduce the complexity of the Newton step, all obtained by different simplifications of the Lyapunov equation (43):

(i) Observe that K is an $O(h)$ perturbation of J, and replace K by J in (43). This results in a modified Newton iteration which is linearly, not quadratically, convergent, with rate of convergence $\rho = O(h)$.

(ii) Write the equation (43) for the components of $\Delta^{(m)}$ in standard linear algebraic form as a system,

$$W_m \hat{\Delta}^{(m)} = b,$$

where $\hat{\Delta}^{(m)} \in \mathbf{R}^6$ is a vector composed of the entries in the upper triangular part of Δ, b is the corresponding part of the right hand side of (43), and W_m is a 6×6 matrix determined from the components of K. Instead of inverting (i.e. factoring) W_m at each iteration, compute and factor W once, at the beginning of the timestep, and use these factors to solve the problem at each step of the iteration. This method is also only linearly convergent, with rate $\rho = O(h)$, but we expect this to be faster than variant (i).

(iii) In case $J = \beta 1$, with scalar β, observe that the matrix B is also nearly a scalar multiple of the identity and that K_m and $\Delta^{(m)}$ will commute to second order,

$$\Delta^{(m)} K_m = K_m \Delta^{(m)} + O(h^2).$$

Replacing $\Delta^{(m)} K_m$ by $K_m \Delta^{(m)}$ we obtain an equation

$$(K_m^T + K_m)\Delta^{(m)} = -F(\lambda_n^{(m-1)}).$$

This equation generally does not have a solution with symmetric $\Delta^{(m)}$, but we enforce the symmetry directly by solving for only the upper triangular part of $\Delta^{(m)}$ and then reflecting this about the diagonal to obtain the update. The latter operation can be viewed as a projection from the class of 3×3 matrices onto the symmetric matrices. The combination of these operations can be viewed as equivalent to an $O(h^2)$ alteration of the matrix W_m in the Newton iteration, hence this scheme would be expected to yield an algorithm that gains *two* orders of the stepsize at each iteration.

The modified iterations will typically require several more iterations per timestep than the Newton iteration to achieve the same tolerence, although they have the advantage of reduced complexity. In our implementation the iteration is considered converged if the norm of Δ is less than 10^{-15} or (42) is satisfied within 10^{-20}. We set the maximum number of iterations to 50 which was necessery for the largest time steps that we have studied.

In our examples, J was a scalar multiple of the identity. Initially we took the simplest initial guess for the lagrange multipliers: the value from the previous step. Even with such a simple minded guess, variant (iii) of the Newton method performed better than the standard Newton. For the soft-sphere dipole system we consider in section IV, this is faster than standard Newton by 60% for $h = 0.005$. One can improve the speed by improving the initial guess for example by keeping the previous three lagrange multipliers and extrapolating from these points. For large time steps this might be beneficial, although it did not have much of an effect in our simulations. Also note that, as the number of particles is increased, the relative time spent in solving λ gets smaller because the total time consumed goes linearly with number of particles - compared to the time required to evaluate forces and torques, which is increases quadratically with particle number.

Variant (i) of the modified Newton iteration was not competitive with either (iii) or the standard Newton. We did not implement (ii).

Appendix B: RSHAKE for Linear and Planar Molecules

For a linear molecule, assumed to be aligned in the reference x-direction, only the J_{11} element of the J matrix is nonzero. Therefore J is not invertable, preventing the straightforward implementation of (27)-(29) and modifications must be made. As before, the Lagrangian for a single rigid body is

$$\mathcal{L} = \frac{M}{2}\dot{\mathbf{r}}_{cm}\dot{\mathbf{r}}_{cm}^T + \frac{1}{2}Tr[\dot{\mathcal{Q}}J\dot{\mathcal{Q}}^T] - \Phi(\mathcal{Q}, \mathbf{r}_{cm}) + Tr[\lambda(\mathcal{Q}^T\mathcal{Q} - 1)] \; . \tag{44}$$

In the linear molecule case, the trace operation in the rotational kinetic energy term (second term) involves only $\dot{\mathcal{Q}}_{\alpha 1}$, where $\alpha = 1, 2, 3$. Also, the potential energy depends only on $\mathcal{Q}_{\alpha 1}$, because the position of any atom in the laboratory frame will be the rotation of a reference coordinate that lies on the x axis of the body frame; that is, the position of the i^{th} atom is given by

$$r^i_\alpha(t) = \mathcal{Q}_{\alpha 1}(t)R^i_1 \tag{45}$$

As a result, the potential energy depends only on $\mathcal{Q}_{\alpha 1}$ and the center-of-mass coordinate $\mathbf{r}_{cm}$. The behavior of $\mathcal{Q}_{\alpha\beta}$ for $\beta = 2, 3$ is irrelevant to the dynamics of the system. In other words, a description of the orientation of a linear molecule requires only the specification of a single vector, which in this case is the first column of the rotation matrix. Thus, the only relevant constraint is that this vector has unit magnitude:

$$\mathcal{Q}_{\alpha 1}\mathcal{Q}_{\alpha 1} - 1 = 0 \tag{46}$$

In summary, the Lagrangian and the corresponding rotational equations of motion for a linear molecule are

$$\mathcal{L} = \frac{M}{2}\dot{\mathbf{r}}_{cm} \cdot \dot{\mathbf{r}}_{cm} + \frac{1}{2}[\dot{\mathcal{Q}}_{\alpha 1}J_{11}\dot{\mathcal{Q}}_{\alpha 1}] - \Phi(\mathcal{Q}_{\alpha 1}, \mathbf{r}_{cm}) + \lambda_{11}(\mathcal{Q}^T_{1\alpha}\mathcal{Q}_{\alpha 1} - 1) \tag{47}$$

and

$$\Pi_{\alpha 1} = \dot{\mathcal{Q}}_{\alpha 1} J_{11} \tag{48}$$
$$\dot{\Pi}_{\alpha 1} = -\partial_{\mathcal{Q}_{\alpha 1}} \Phi + 2\mathcal{Q}_{\alpha 1}\lambda_{11} \tag{49}$$
$$g(\mathcal{Q}) = \mathcal{Q}^T_{1\alpha}\mathcal{Q}_{\alpha 1} - 1 = 0 \;, \tag{50}$$

with the RSHAKE algorithm becomes

$$\Pi^{n+1/2}_{\alpha 1} = \Pi^{n-1/2}_{\alpha 1} + h\left[2\mathcal{Q}^n_{\alpha 1}\lambda^n_{11} - \partial_{\mathcal{Q}^n_{\alpha 1}}\Phi\right] \tag{51}$$
$$\mathcal{Q}^{n+1}_{\alpha 1} = \mathcal{Q}^n_{\alpha 1} + h\Pi^{n+1/2}_{\alpha 1}/J_{11} \tag{52}$$
$$g(\mathcal{Q}^{n+1}) = 0 \tag{53}$$

In this case the constraint equation (53) is simply a quadratic equation for λ^n_{11} which can be solved exactly. The above algorithm for a linear molecule has appeared in the literature [20] and is referred to as the LEN method. Fincham showed that LEN is remarkably stable and superior to his implicit, explicit and ORT methods.

For a planar molecule, assumed to lie entirely in the $x-y$ plane of the body frame, the J matrix will have J_{11} and J_{22} as the only nonzero elements. In analogy with the linear case, the potential energy will only depend upon $\mathcal{Q}_{\alpha 1}$ and $\mathcal{Q}_{\alpha 2}$ together with $\mathbf{r}_{cm}$. In this case the evolution of $\mathcal{Q}_{\alpha 3}$ is irrelavent - only two non-colinear vectors are necessary to describe a planar molecule. In this representation these two vectors are the first and second columns of the rotation matrix. The rotational equations of motion for a planar molecule are then

$$\Pi_{\alpha i} = \dot{\mathcal{Q}}_{\alpha j} J_{ji} \tag{54}$$
$$\dot{\Pi}_{\alpha i} = -\partial_{\mathcal{Q}_{\alpha i}} \Phi + 2\mathcal{Q}_{\alpha j}\lambda_{ji} \tag{55}$$
$$g(\mathcal{Q}) = \mathcal{Q}^T_{i\alpha}\mathcal{Q}_{\alpha j} - \delta_{ij} = 0 \;, \tag{56}$$

where $\alpha = 1, 2, 3$ and $i, j = 1, 2$ and repeated indices are summed over. The RSHAKE algorithm for a planar molecule reduces to

$$\Pi^{n+1/2}_{\alpha i} = \Pi^{n-1/2}_{\alpha i} + h\left[2\mathcal{Q}^n_{\alpha j}\lambda^n_{ji} - \partial_{\mathcal{Q}^n_{\alpha i}}\Phi\right] \tag{57}$$
$$\mathcal{Q}^{n+1}_{\alpha i} = \mathcal{Q}^n_{\alpha i} + h\Pi^{n+1/2}_{\alpha j}(J^{-1})_{ji} \tag{58}$$
$$g(\mathcal{Q}^{n+1}) = 0 \tag{59}$$

In summary, whether the molecule is linear, planar or three dimensional the equations of motion and the corresponding leap-frog algorithm are described by (27)-(29) and (31)-(32) with the understanding that the dimensions of the variables are given as in the Table II.

REFERENCES

[1] Email: ayla@stout.chem.ukans.edu
[2] Email: laird@pilsner.chem.ukans.edu
[3] Email: leimkuhl@math.ukans.edu
[4] H. Goldstein, *Classical Mechanics*, (Addison-Wesley, Reading, Massachussetts, 1980).
[5] D.J. Evans, Mol. Phys. **34**, 317 (1977).
[6] M.A. Allen and D.J. Tildesley, *Computer Simulation of Liquids*, (Oxford Science Press, Oxford, 1987).
[7] V.I. Arnold, *Mathematical Methods of Classical Mechanics*, (Springer-Verlag,, New York, 1978).
[8] J.M. Sanz-Serna and M.P. Calvo, *Numerical Hamiltonian Problems* (Chapman and Hall, New York ,1995).
[9] J. Frank, W. Huang, and B. Leimkuhler, J. Comput. Phys., to appear.
[10] Andersen, H.C., J. Comput. Phys., **52**, 24 (1983).
[11] B. Leimkuhler and R. D Skeel, J. Comput. Phys., **112**, 117 (1994).
[12] J. P. Ryckaert, G. Ciccotti, and H. J. C. Berendsen, J. Comput. Phys. **23**, 327 (1977).
[13] D. Fincham, Mol. Simul. **8**, 165 (1992).
[14] R.I. McLachlan and C. Scovel, J. Nonlinear Sci. **5**, 233 (1995).
[15] S. Reich, Physica D, **76**, 375-383 (1994).
[16] R. D. Ruth, IEEE Trans. Nucl. Sci., **30**, 2669 (1983).
[17] R. Ahlrichs and S. Brode, Comput. Phys. Commun. **42**, 59 (1986).
[18] P.G. Kusalik, J. Chem. Phys. **93**, 3520 (1990).
[19] D. Wei and G.N. Patey, Phys. Rev. A **46**, 2043 (1992).
[20] D. Fincham, Mol. Simul. **11**, 79 (1993).

TABLES

TABLE I. Comparison of RSHAKE with quaternion method.

N	h^*	ϵ_l^*	ϵ_g^*	CPU (s)
		RSHAKE		
256	0.002	1.1d-5	1.2d-6	5022
	0.004	4.3d-5	4.7d-6	2853
	0.006	1.0d-4	1.5d-5	2054
	0.008	1.8d-4	4.8d-5	1632
	0.010	2.7d-4	3.0d-5	1381
864	0.002	5.8d-6	6.4d-7	21391
	0.004	2.3d-5	6.6d-6	12167
	0.006	5.2d-5	8.5d-6	9421
	0.008	9.5d-5	1.0d-5	8069
	0.010	1.5d-4	1.6d-5	7619
		QUATERNION		
256	0.002	7.4d-6	6.0d-5	3306
	0.004	3.0d-5	5.9d-4	1762
	0.006	6.9d-5	2.4d-3	1246
	0.008	1.2d-4	6.4d-3	992
864	0.002	4.1d-6	6.1d-5	15643
	0.004	1.5d-5	5.8d-4	9160
	0.006	3.5d-5	2.4d-3	7026
	0.008	6.5d-5	6.4d-3	5815

TABLE II. Dimensions of parameters for different molecular geometries

MOLECULE	$\mathcal{Q}$	Π	J	λ
3-D	3x3	3x3	3x3	3x3
PLANAR	3x2	3x2	2x2	2x2
LINEAR	3x1	3x1	1x1	1x1

FIGURES

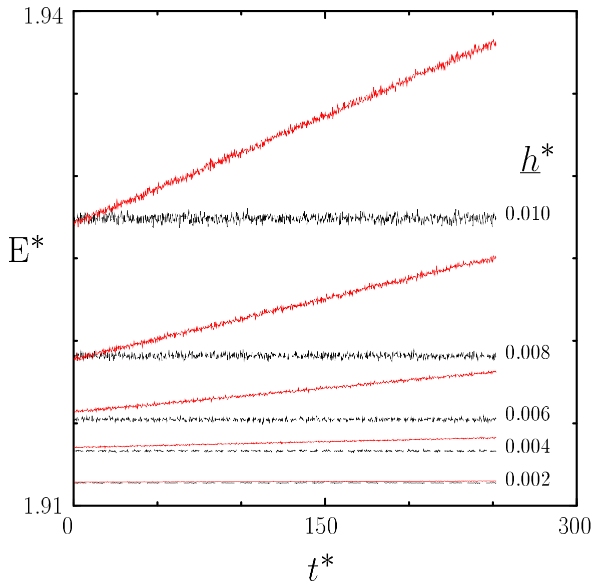

FIG. 1. Total energy as a function of time for a 500 particle DSS system using both RSHAKE (solid-staight lines) and standard quaternions (dashed-sloped lines). (The results for a variety of time steps have been shifted relative to each other along the energy axis for clarity.)

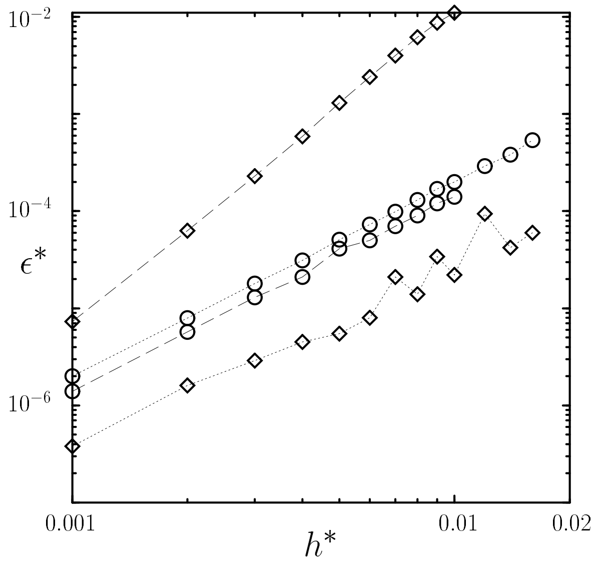

FIG. 2. Global (diamonds) and local (circles) energy errors as functions of time step, h, (as a log-log plot) for a 500 particle DSS system using both RSHAKE (dotted lines) and the 4th-order (Gear) quaternion integrator (dashed lines).

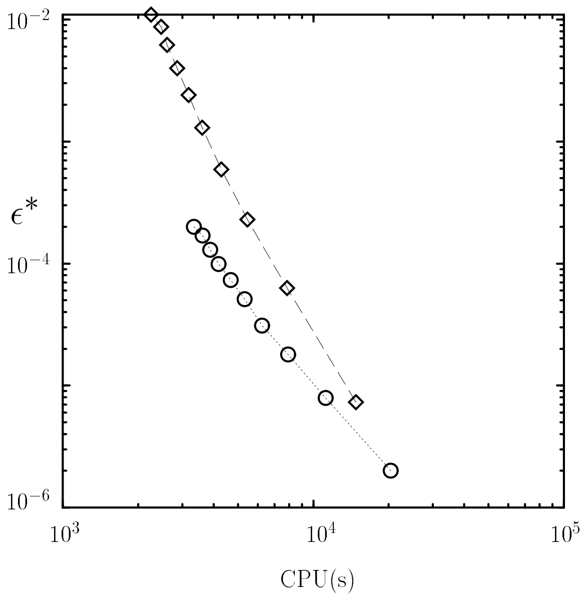

FIG. 3. Dominant energy error as a function of CPU time (in seconds) for a 500 particle DSS system using RSHAKE (circles) and the quaternion integrator (diamonds).

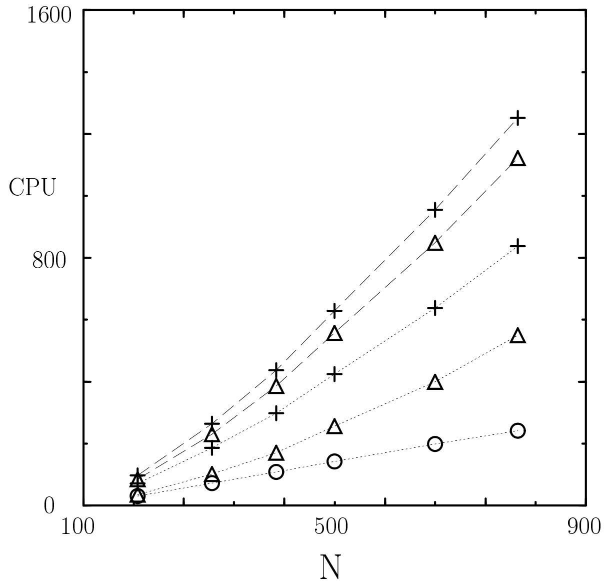

FIG. 4. Total CPU (crosses) as a function of system size N for RSHAKE (dotted line, $h = 0.005t_0$) and quaternions (dashed line, $h = 0.002t_0$). Also shown is the CPU time for the force evaluations in both methods (triangles) as well as the CPU time required to solve for the Lagrange multiplier matrix in RSHAKE (circles).

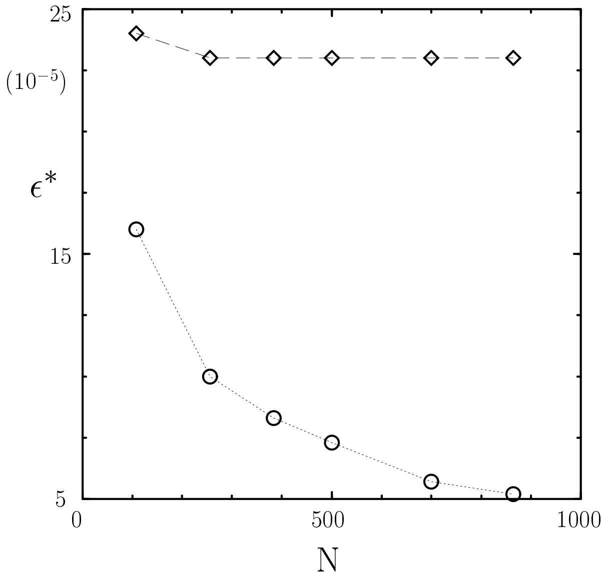

FIG. 5. Dominant energy error as a function of N for RSHAKE (circles, $h = 0.006t_0$, local energy error) and the quaternion integrator (diamonds, $h = 0.003t_0$, global energy error).