A Modified Shake Algorithm for Maintaining Rigid Bonds in Molecular Dynamics Simulations of Large Molecules

S. G. LAMBRAKOS, J. P. BORIS, AND E. S. ORAN

Laboratory for Computational Physics and Fluid Dynamics, Naval Research Laboratory, Washington, DC 20375-5000

AND

I. CHANDRASEKHAR* AND M. NAGUMO

Bio/Molecular Engineering Branch, Naval Research Laboratory, Washington, DC 20375-5000

Received September 8, 1987; revised February 24, 1989

We present a new modification of the SHAKE algorithm, MSHAKE, that maintains fixed distances in molecular dynamics simulations of polyatomic molecules. The MSHAKE algorithm, which is applied by modifying the leapfrog algorithm to include forces of constraint, computes an initial estimate of constraint forces, then iteratively corrects the constraint forces required to maintain the fixed distances. Thus MSHAKE should always converge more rapidly than SHAKE. Further, the explicit determination of the constraint forces at each timestep makes MSHAKE convenient for use in molecular dynamics simulations where bond stress is a significant dynamical quantity. © 1989 Academic Press, Inc.

I. INTRODUCTION

A fundamental difficulty with the simulation of dynamics of polyatomic molecules is the large range of time-scales associated with molecular motions. The duration of a simulation is limited by the magnitude of the smallest timestep, which must be small compared to the shortest characteristic time of interest. Often the slower modes of a molecule, for example, rotations about bonds, are of more interest than the bond vibrations. In such cases, much longer periods may be simulated by constraining the bonds to fixed lengths, thus allowing a larger timestep.

Various iterative and noniterative methods have been developed for maintaining rigid bonds in molecular dynamics simulations of large molecules [1–10]. Two commonly used algorithms are SHAKE [2, 3], which is iterative, and the matrix method [2], which is noniterative. The matrix method inverts a matrix to solve for

* Molecular Biophysics Unit, Indian Institute of Science, Bangalore 560-012 India.

0021-9991/89 $3.00

Lagrange multipliers that satisfy the constraint conditions, and so becomes computationally expensive for very large molecules. The SHAKE algorithm avoids explicit matrix inversion by iteratively adjusting particle coordinates until the system satisfies all the constraints to within a given tolerance. In addition to maintaining rigid bonds, constraint algorithms must correct for the increasing departure from the ideal lengths, called constraint decay, that results from the accumulation of numerical errors. Iterative algorithms correct for constraint decay implicitly by requiring convergence to within a specified tolerance at each timestep. Deviations in the constrained distances from their initial values are continually checked and corrected. Noniterative algorithms require an explicit scheme for counteracting constraint decay because there is no inherent feedback mechanism for monitoring changes in distance. Recently Edberg *et al.* [6] developed a noniterative constraint algorithm that defines penalty functions that monitor constraint deviations. The penalties are minimized by correcting the deviations according to Gauss' principle of least constraint [7]. By permitting the constraint to relax slightly, the computational cost is reduced because the accumulation of numerical errors is not corrected at every timestep.

In this paper, we present a modification of the SHAKE algorithm for enforcing holonomic constraints in polyatomic molecules. A constraint force function is derived using the leapfrog algorithm with constraint conditions applied pairwise to all restricted particles. Memon *et al.* [8] have also developed a constraint force algorithm based on leapfrog integration. However, where they solve for constraint forces by matrix inversion, the MSHAKE algorithm first computes an initial estimate for each constraint force using finite differences of previous timesteps and then performs functional interation to reach convergence. The finite differences used to compute the initial estimates of the constraint forces require little extra work because the constraint forces are calculated explicitly at each timestep. Tests using the MSHAKE algorithm show that it converges rapidly and that numerical errors do not accumulate, so there is no unstable "constraint decay" [6]; the constraint error fluctuates stably.

In principle, any two-particle constraint algorithm can serve as the basis of an iterative multiple-constraint algorithm since the evaluation of any two-particle constraint force can treat constraint forces from previous calculations as external forces. In practice, however, this is not always the case. For example, Singer *et al.* [9] have developed an efficient and widely used [10] two-particle constraint algorithm that treats the dynamics of the center-of-mass separately from the rotational motion. Because it treats the dynamics of two linked particles implicitly, it cannot be extended simply to a system with multiple constraints. The MSHAKE algorithm discussed below treats multiple constraints in a very natural manner, since constraint forces calculated at one iteration are treated as though they were external forces on the next iteration.

Our method of computing constraints is well suited to molecular dynamics simulations where bond stress is a significant dynamical quantity [11, 12]. We therefore present a description of constraints in terms of constraint forces, rather than bond distances, and give a detailed geometric interpretation of the constraint force (see Appendix) that should provide a basis for its use in monitoring various aspects of bond stress in molecular dynamics simulations. The finite-difference scheme used in MSHAKE requires the constraint force per bond at previous timesteps. In molecular dynamics simulations where bond stress is a significant quantity, computing and storing the constraint force per bond is a negligible additional cost.

II. The Modified SHAKE Algorithm

A. *Background*

The mathematical basis of the MSHAKE algorithm presented here is the same as that for all other iterative constraint algorithms. We assume in this analysis that all constraints may be expressed as constraints between pairs of particles. For each linked particle in the system, a force $\Delta\mathbf{F}_i$, the sum of all constraint forces acting on the particle i, is added to the total force $\mathbf{F}_i$ on particle i. The quantity $\Delta\mathbf{F}_i$ may be given by

$$\Delta\mathbf{F}_i = \sum_{j=1}^{K_i} \alpha_{ij}\boldsymbol{l}_{ij}, \tag{1}$$

where the α_{ij} are constants that specify the magnitude of each constraint force and the $\boldsymbol{l}_{ij}$ are the K_i vectors from i to its constrained partners j. If there are N constraints, the constrained system has N unknowns α_{ij}, and the system is solvable. Memon *et al.* [8] have discussed efficient matrix methods for determining these unknowns using constraint equations obtained from the leapfrog integration scheme. Ryckaert *et al.* [2] pointed out that because the solution to the constraint system of equations is unique, any convergent procedure that satisfies all geometric constraint conditions by displacements ΔX_i of the form

$$\Delta\mathbf{X}_i = \sum_{j=1}^{K_i} \frac{\alpha_{ij}(\delta t)^2}{\mu_{ij}}\boldsymbol{l}_{ij}, \tag{2}$$

where δt is the magnitude of the timestep and μ_{ij} is the reduced mass of particles i and j, is equivalent to results obtained by solving Eq. (1). Some iterative constraint algorithms adjust Eq. (1), e.g., Memon *et al.*, while other adjust Eq. (2), e.g., SHAKE, to satisfy the constraints.

At each time step, the MSHAKE algorithm computes an approximation of the constraint constants $\{\alpha_{ij}\}$ using the constraint constants from previous timesteps. Corrections are then applied to $\{\alpha_{ij}\}$ using rapidly converging procedure. Because the initial estimates of $\{\alpha_{ij}\}$ are usually close to the actual solutions, the number of iterations required in MSHAKE is significantly less than in other iterative schemes. Further, MSHAKE calculates the constraint force explicitly at each timestep, rather than implicitly, as is the case in SHAKE. The explicit calculation of the constraint force on each bond is important in molecular dynamics simulations where the force along molecular bonds is required to determine significant dynamical attributes of the molecules [11, 12], and thus MSHAKE is better suited than SHAKE for such problems.

B. *Derivation*

In this section, we develop a procedure to calculate the constraint force $\Delta\mathbf{F}_i$ for maintaining fixed separations between particles that are subject to more than one distance constraint. Consider a system of particles in which each particle moves in the force field of all the others. A constraint on any given particle, such as a fixed separation or fixed angular orientation, imposes a condition on that particle's trajectory. The constraint force can be defined by the equations of motion describing the trajectory of a given particle i,

$$m_i \frac{d^2\mathbf{X}_i}{dt^2} = \mathbf{F}_i + \Delta\mathbf{F}_i(\mathbf{X}_i, \{\mathbf{X}_j\}, \mathbf{V}_i, \{\mathbf{V}_j\}, \{\mathbf{F}_i\}, \{\mathbf{F}_j\}). \tag{3}$$

In Eq. (3), $\mathbf{F}_i$ is the total force on particle i due to the other particles in the system, $\Delta\mathbf{F}_i$ is the total constraint force on particle i; m_i, $\mathbf{X}_i$, and $\mathbf{V}_i$ are the mass, position, and velocity, respectively, of particle i; $\{\mathbf{X}_j\}$ and $\{\mathbf{V}_j\}$ are the sets of positions and velocities of the particles that are constrained with particle i; and $\{\mathbf{F}_i\}$ and $\{\mathbf{F}_j\}$ are the sets of forces, both external and constraint, that act on particles i and j.

First consider the two-particle, single constraint case. We define the constraint force along the bond between particles i and j, $\Delta\mathbf{F}_{ij}$, to be the force required to maintain a fixed separation between particle i and particle j at each discrete timestep of the leapfrog integration of Eq. (3). The leapfrog procedure [13, 14] is modified to include the forces of constraint, $\Delta\mathbf{F}_{ij}$, so that the position $\mathbf{X}$ and the velocity $\mathbf{V}$ of each particle are given by

$$\mathbf{X}_i^{n+1} = \mathbf{X}_i^n + \mathbf{V}_i^{n+1/2}\,\delta t \tag{4}$$

and

$$\mathbf{V}_i^{n+1/2} = \mathbf{V}_i^{n-1/2} + (\mathbf{F}_i^n + \Delta\mathbf{F}_{ij}^n)\frac{\delta t}{m_i}. \tag{5}$$

The superscripts in Eqs. (4) and (5) indicate that $\mathbf{X}$ and $\mathbf{V}$ are centrally differenced. After n timesteps of size δt, $\mathbf{V}$ and $\mathbf{X}$ are computed at times $(n+\frac{1}{2})\,\delta t$ and $(n+1)\,\delta t$, respectively.

The constraint force $\Delta\mathbf{F}_{ij}^n$ must satisfy the condition

$$|\mathbf{X}_j^{n+1} - \mathbf{X}_i^{n+1}|^2 = l_0^2, \tag{6}$$

where $l_0 = |\mathbf{l}_0|$ is the fixed distance between particles i and j. However, in any real simulation, the errors accumulate at each timestep so that

$$|\mathbf{X}_j^n - \mathbf{X}_i^n|^2 = l_{ij}^2, \tag{7}$$

where $l_{ij} = |\boldsymbol{l}_{ij}| \approx l_0$, but in general $\boldsymbol{l}_{ij} \neq \boldsymbol{l}_0$. Equation (6) may be written, using Eqs. (4) and (5), as

$$l_0^2 = \left| (\mathbf{X}_j^n - \mathbf{X}_i^n) + (\mathbf{V}_j^{n-1/2} - \mathbf{V}_i^{n-1/2})\,\delta t + \left(\frac{\mathbf{F}_j^n}{m_j} - \frac{\mathbf{F}_i^n}{m_i}\right)(\delta t)^2 - \frac{\Delta \mathbf{F}_{ij}^n (\delta t)^2}{\mu_{ij}} \right|^2, \tag{8}$$

where μ_{ij} is the reduced mass.

The constraint force can be found directly from Eq. (8);

$$\Delta \mathbf{F}_{ij} = \frac{\mu_{ij} a_{ij}}{(\delta t)^2} \boldsymbol{l}_{ij}, \tag{9}$$

where

$$a_{ij} = \left[1 + \frac{\boldsymbol{l}_{ij} \cdot \Delta \boldsymbol{l}_{ij}}{l_{ij}^2} - \sqrt{\frac{l_0^2}{l_{ij}^2} + \frac{(\boldsymbol{l}_{ij} \cdot \Delta \boldsymbol{l}_{ij})^2}{l_{ij}^4} - \frac{(\Delta l_{ij})^2}{l_{ij}^2}} \right] \tag{10}$$

and

$$\Delta \boldsymbol{l}_{ij} = (\mathbf{V}_j^{n-1/2} - \mathbf{V}_i^{n-1/2})\,\delta t + \left(\frac{\mathbf{F}_j^n}{m_j} - \frac{\mathbf{F}_i^n}{m_i}\right)(\delta t)^2, \tag{11}$$

the change in $\boldsymbol{l}_{ij}$ due to the forces and velocities at timestep $(n+1)\,\delta t$ without the constraint forces applied. A geometric interpretation of Eq. (9) is given in Appendix 1, where we show that the negative root gives the physically reasonable value for the constraint force.

At this stage, the algorithm is exact for the special case of single-constraint systems, e.g., diatomic molecules. To extend the method to polyatomic molecules, we employ simple functional iteration solving the system of equations using Gauss–Jacobi iteration. We begin the first iteration by calculating the force displacement for the first pair of constrained atoms using Eq. (9). The constraint force is then added to the force acting on each particle, i.e., in Eq. (11), $\mathbf{F}_i \rightarrow \mathbf{F}_i + \Delta \mathbf{F}_{ij}$ and $\mathbf{F}_j \rightarrow \mathbf{F}_j - \Delta \mathbf{F}_{ij}$. The next successive displacement is calculated using Eq. (9) with the updated forces. This continues until force displacements have been calculated for all constrained pairs. Iteration is continued for all unconverged pairs. Finally, the equations of motion are integrated and the next time step begins.

The MSHAKE method, although defined in terms of constraint forces (Eqs. (10) and (11)), would be equivalent to the SHAKE method if used according to the procedure outlined in the previous paragraph. However, by introducing a finite-difference scheme, the MSHAKE method defines a procedure that is *essentially* linear and thus significantly different in character from SHAKE.

The convergence of the calculation can be greatly improved by starting with an estimate of the constraint force derived from preceding timesteps:

$$\Delta \mathbf{F}_{ij}^n \approx \frac{\mu_{ij} \mathscr{A}_{ij}}{(\delta t)^2} \boldsymbol{l}_{ij}, \tag{12}$$

The constraint force estimates are then added, with proper signs, to the forces on particles i and j, and processing by MSHAKE proceeds as in the previous paragraph, but with many fewer iterations. If the converged estimates A_{ij} are retained for several timesteps, finite difference approximations for the nth time step, $\mathscr{A}_{ij}^{m,n}$, can be evaluated as

$$\mathscr{A}_{ij}^{m,n} = \begin{cases} 0, & \text{if } m=0; \\ A_{ij}^{n-1}, & \text{if } m=1; \\ 2A_{ij}^{n-1} - A_{ij}^{n-2}, & \text{if } m=2, \end{cases} \tag{13}$$

where

$$A_{ij}^{n} = \mathscr{A}_{ij}^{m,n-1} + \sum_{\text{iterations}} a_{ij}. \tag{14}$$

The implementation of the MSHAKE algorithm is described in the next section.

C. *Procedure for Calculating Multiple-Constraint Forces*

The procedure is illustrated by the flowchart in Fig. 1. For each particle i in the system:

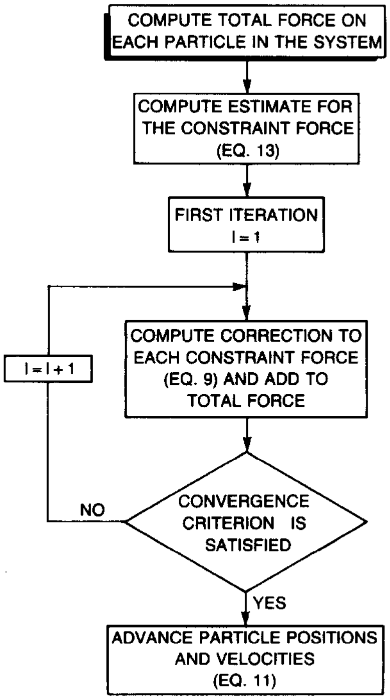
COMPUTE TOTAL FORCE ON EACH PARTICLE IN THE SYSTEM
COMPUTE ESTIMATE FOR THE CONSTRAINT FORCE (EQ. 13)
FIRST ITERATION I = 1
COMPUTE CORRECTION TO EACH CONSTRAINT FORCE (EQ. 9) AND ADD TO TOTAL FORCE
CONVERGENCE CRITERION IS SATISFIED
NO
I = I + 1
YES
ADVANCE PARTICLE POSITIONS AND VELOCITIES (EQ. 11)

FIG. 1. Flow chart illustrating the MSHAKE algorithm.

(1) Compute $\mathbf{F}_i$, the sum of all the nonconstraint forces acting on each particle.

(2) Compute an estimate for the constraint force on each particle based on the values at previous time steps, and add to $\mathbf{F}_i$. During initialization, the estimate is set equal to zero.

(3) Compute the constraint force, $\Delta\mathbf{F}_{ij}$ defined by Eq. (9) for each bond ij in the system, and add the constraint force to the force on each particle.

(4) If the estimate of total force has converged, then move the particles, otherwise continue looping at step (3).

D. *Features of the MSHAKE Algorithm*

(1) The use of simple functional iteration on the constraint forces allows MSHAKE to be essentially linear in character. In order that the numerical solution of the constrained equations-of-motion be well behaved, a timestep must be chosen so that the change in the external force on each particle is small over the time interval defined by the timestep. As a result, since the constraint force cannot exceed the external force on each pair, the constraint force for each bond is slowly varying and well behaved over several timesteps. Thus initial estimates of the constraint force magnitude, a_{ij}, based on converged estimates from one or two preceeding timesteps, have the property that they lie within a small neighborhood, i.e., in the linear region, of the solution for the current timestep. Therefore, the use of final values as initial estimates for the current iteration yields rapid convergence. Further, since this property is independent of system size, it follows from the theory of nonlinear equations [15] that large systems should converge as rapidly as small ones.

(2) Although forces and positions are simply related, using final estimates of position displacements may not be as efficacious as force displacements. The timestep is chosen such that forces will not vary greatly with timestep, but positions may not be as well behaved. Since position is a function of velocity as well as acceleration, a constant large acceleration will yield no variation in the force but a large variation in the velocity and position with time. Further, velocity and position are affected by forces from the two previous timesteps, cf., Eqs. (4) and (5). Therefore, retaining estimates of the magnitude of the change in particle position may not guarantee initial estimates in the near neighborhood of the current solution.

(3) The development of the constraint forces during the first two iterations of MSHAKE, with $m = 0$, i.e., no initial estimate of the constraint force, is illustrated in Fig. 2. For this example, we have chosen to perform Gauss–Jacobi iteration from top to bottom. During the first iteration, the first constraint force calculated, $\Delta\mathbf{F}_{12}$, depends only on the distance between particles 1 and 2, their velocities, and the external forces acting on them. The second constraint force calculated, $\Delta\mathbf{F}_{23}$, depends on the corresponding quantities, but the forces that act on particle 2 now include $\Delta\mathbf{F}_{12}$, a force of constraint. During all subsequent iterations, the constraint force $\Delta\mathbf{F}_{ij}$ depends on all the constraint forces previously calculated for particles i

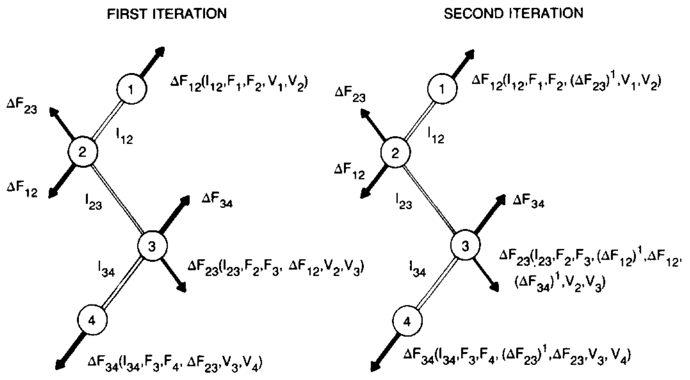
FIRST ITERATION
SECOND ITERATION
ΔF12(l12,F1,F2,V1,V2)
ΔF23
l12
ΔF12
l23
ΔF34
l34
ΔF23(l23,F2,F3, ΔF12,V2,V3)
ΔF34(l34,F3,F4, ΔF23,V3,V4)
ΔF12(l12,F1,F2,(ΔF23)^1,V1,V2)
ΔF23(l23,F2,F3,(ΔF12)^1,ΔF12,(ΔF34)^1,V2,V3)
ΔF34(l34,F3,F4,(ΔF23)^1,ΔF23,V3,V4)

FIG. 2. Constraint forces contributing to the total force on each bound particle at different iterations. The quantity $\mathbf{F}_i$ is the total force on particle i not including constraint forces. The quantity $(\Delta\mathbf{F}_{ij})^k$ is the constraint force for maintaining bond ij at the kth iteration.

and j. As the timestep δt decreases, the order of evaluation has less of an effect on the increment to the total force eventually applied to each particle, and the number of iterations required for computing the constraint forces decreases. Note that the forces between all mutually constrained particles are set to zero before evaluating the constraint force. In principle, this is not necessary because the constraint force counteracts any mutual interaction. However, interactions at bond-length separations are relatively large and result in large values for $\mathbf{l}_{ij}\cdot\Delta\mathbf{l}_{ij}$ in Eq. (10). Furthermore, the force gradient is large at short range, necessitating a very small timestep.

(4) As the number of iterations k increases, $(\Delta\mathbf{F}^n_{ij})^k \rightarrow 0$, independent of the order in which the $(\Delta\mathbf{F}^n_{ij})^k$ are computed. This property provides flexibility for computing the constraint forces because the calculation may be partitioned for optimal processing for a given problem or computer.

(5) The MSHAKE algorithm, when used with its least-optimal mode, $m=0$, converges as rapidly as SHAKE. However, when estimates of a_{ij} are included, MSHAKE is more efficient than other iterative schemes. If the constraint constants $\{A_{ij}\}$ are kept for at least two previous steps, an efficient finite-difference scheme can be used to compute a good approximation to changes in the values of the constraint force. Since the constraint forces are usually smooth and slowly varying functions of time, the linear approximation in Eq. (12) is usually adequate. Thus it is possible to reduce the number of iterations appreciably and maintain the same level of accuracy as other iterative schemes at little additional computer cost.

There are two advantages of using the constraint constants $\{A_{ij}\}$, rather than the constraint forces themselves, as estimates for the constraint forces in the following timestep. First, at the next timestep, the geometry and orientation of the molecule will have changed somewhat, hence the vector constraint force must also change, even though its magnitude may not. When the $\{A_{ij}\}$ are used, the new force estimate is found simply by multiplying $\{A_{ij}\}$ by the direction cosines of bond ij.

Second, only scalars need be stored for each constrained distance rather than the components of the corresponding vector forces.

(6) Small errors in the computed value of $\Delta \mathbf{F}_{ij}^{n}$ combined with errors resulting from discretization produce two types of constraint decay: deviations of the bond lengths from their assigned values and nonzero relative velocity between constrained pairs of particles along the direction of the bond. The multiple-constraint force $\Delta \mathbf{F}_{ij}^{n}$ has two features that counteract constraint decay. These are the condition imposed by Eq. (6) that causes $\Delta \mathbf{F}_{ij}^{n}$ to maintain an assigned bond length, and the contribution of the first term in $\Delta \mathbf{l}_{ij}$ in Eq. (1)) to $\Delta \mathbf{F}_{ij}^{n}$ that minimizes any relative velocity components along bonds. These two factors contribute to the stability of the algorithm and give the algorithm the ability to correct for the accumulation of small errors resulting from discretization or any approximation of a_{ij}.

(7) The constraint force along a bond can be used as a measure of the stress on the bond induced by other particles and fields in the system. For example, in constrained molecular dynamics simulations of shocks and detonations in solids [11, 12], when the constraint force along a fissile bond reaches some pre-established value, the bond may be broken, and the bond energy distributed as kinetic energy between the fragments. The MSHAKE algorithm is well suited for such simulations, since the constraint force along each bond is calculated explicitly at each timestep.

III. Demonstration of General Features

We conducted a series of molecular dynamics simulations using a system of particles with constraints. Our goal was *not* to model any particular real system but to demonstrate the general features of the MSHAKE algorithm with a system of highly constrained molecules whose interactions are typical of real molecules. We compared the performance of the MSHAKE algorithm as a function of the order of approximation. When $m = 0$, the lowest order, MSHAKE solves the same system of equations as SHAKE and therefore provides a good general estimate of the characteristics of iterative constraint algorithms. We used this general estimate as a basis for investigating the accuracy and efficiency of the MSHAKE method. Our results demonstrate that MSHAKE is stable, that it has accuracy comparable to SHAKE, and that it converges rapidly relative to other iterative constraint algorithms.

The model used for these tests consisted of a 12×12 array of rigid tetra-atomic molecules in three-dimensional space. The bond lengths and angles were fixed at the normal carbon–carbon single bond length of 1.54 Å and at the tetrahedral angle, 109°28′, respectively. Non-bonded interactions were given by a Lennard–Jones potential with parameters taken from atomic nitrogen [16], $\varepsilon = 0.5143 \times 10^{-14}$ erg and $\sigma = 3.310$ Å. The boundary conditions were periodic in the plane of the initial array. The system was bounded in the third dimension by a pair of walls with the same LJ parameters as the particles themselves. The MLG algorithm [17, 18] was used to compute interactions and track particle positions. Each molecule in the system was given a random initial velocity, and the motion was calculated with timesteps varying between 2×10^{-16} and 2×10^{-14} s until the kinetic and potential energies reached steady values. Test simulations were conducted starting from these equilibrated systems, during which we monitored the deviation of actual distances l_{ij} from their proper values l_0, the energy and momentum of the particles in the system, and the constraint force per bond per iteration.

Using the system described above, a simulation was conducted to test the stability of the MSHAKE algorithm. The energy and geometric constraint deviations were monitored for 42000 steps of 5×10^{-16} s each. Five iterations were used to evaluate the constraint forces at each timestep. The maximum deviation from exact lengths is about 2%. The average number of deviations greater than 1% per timestep is about 3, and the maximum recorded is 12 out of 720 bonds in the whole system. The identity of the constrained particles responsible for these deviations does not stay the same for more than about 100 timesteps. Both the small number and the fluctuating identity of deviations demonstrate the ability of the MSHAKE algorithm to counteract constraint decay. Under these conditions, we prefer to speak of "constraint relaxation," since there is no tendency for the number or magnitude of the deviations to grow. We conducted these same tests with a similar system that did not have fixed bond angles. In these simulations the average magnitude of constraint deviation was negligible after five iterations.

Simulations were also conducted to test the extent to which the constraint forces calculated in our model depend upon the sequences in which the constraint-force corrections are evaluated at each iteration k. The trajectories of particles were compared for simulations starting from the same state but having the constraint-force corrections calculated in a different sequence. The number of constraint force iterations per timestep for all these simulations was kept fixed at 5. The final states of two simulations run over 3×10^{-12} s for $\delta t = 1 \times 10^{-15}$ s, one corresponding to a forward evaluation and the other to a backward evaluation, differ by one part in 10^5 in the mean particle displacements. These tests show that the stability of the algorithm does not depend on the order of evaluating the constraint-force corrections. For larger timesteps, the sequence in which the constraint-force corrections are evaluated has a larger effect on the trajectory due to the constraint relaxation associated with the highly constrained system.

The constraint-force correction, $(\Delta \mathbf{F}_{ij}^n)^k$, for a typical bond is shown in Fig. 3 as a function of iteration k for A_{ij} calculated using three different estimates for a_{ij}^n. For all values of m, the magnitude of the correction to the constraint force falls off exponentially with increasing iterations through the constraint force algorithm. The convergence rate and associated accuracy are comparable to those of the SHAKE algorithm for a similar system [2]. This is to be expected since, for $m = 0$, the constraint-force corrections (defined by Eq. (1)) are proportional to the spatial adjustments (defined by Eq. (2)) that would be made by the SHAKE algorithm in

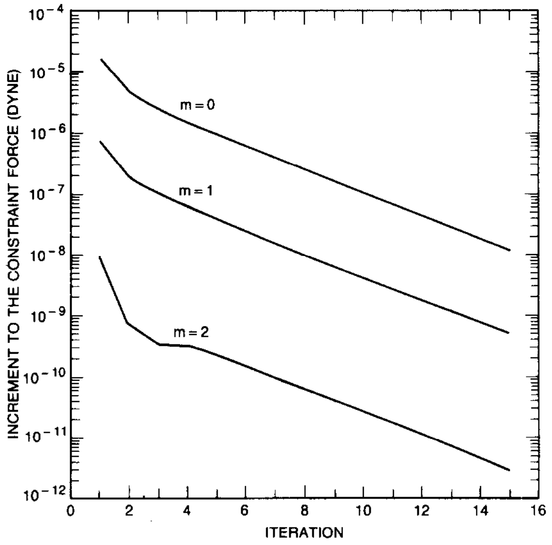

FIG. 3. Correction to the constraint force along one bond as a function of iteration through the MSHAKE algorithm for increasing accuracy of approximation.

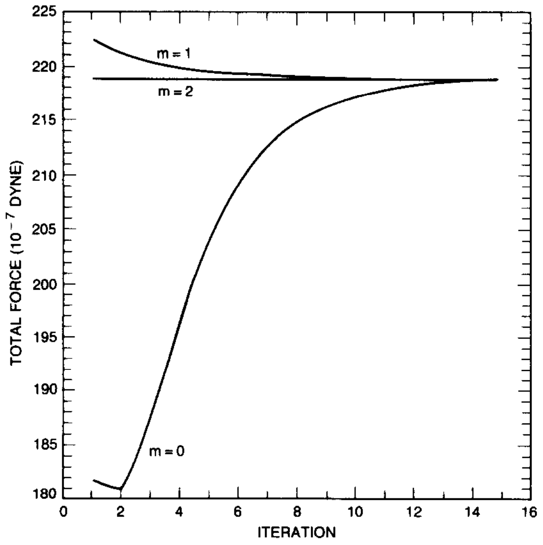

FIG. 4. Convergence of the total force on one particle as a function of iteration through the MSHAKE algorithm for increasing accuracy of approximation.

order to satisfy the same constraint conditions. For $m \neq 0$, there is a significant decrease in the number of iterations required for convergence. Note that there is no loss of accuracy associated with using the estimations ($m \neq 0$), since at each iteration any residual associated with A_{ij} is uniquely defined in terms of the constraint conditions. Thus, it follows that zero is a unique limit point of the sequence $\{(\Delta \mathbf{F}_{ij}^n)^k\}$. The convergence of the total force (external + constraint) acting on one of the particles of the bond involved in Fig. 3 is illustrated in Fig. 4. The negligible change in the total force for $m = 2$ upon iteration through the constraint force algorithm suggests that the maximum number of iterations might be reduced significantly in this case, resulting in a corresponding reduction in computation time.

IV. Conclusion

We have derived a new algorithm, MSHAKE, for constraint dynamics that represents a significant modification of the SHAKE algorithm. Our method consists of two steps: (1) calculation of an estimate using a finite-difference scheme; and (2) calculation of corrections using iterative relaxation. We have presented a comparison of our method to SHAKE (which uses only iterative relaxation) that is independent of the mode of implementation. The number of iterations required in the SHAKE method depends on the particular type of constrained system. Thus for SHAKE, maintaining a large number of constraints should require a large number of iterations. With our method, regardless of the number of constraints to be maintained in a system, the initial estimates are always within a very small neighborhood of the solution. Thus, iterates are always small, if not negligible (see $m = 2$ in Fig. 4), and any relaxation procedure will converge rapidly. Thus, it should not be expected that an increase in the number of constraints will not require an increase in the number of iterations for calculating corrections to the initial estimates.

We have also described the calculation of multiple constraints from the standpoint of constraint-force functions. Our purpose in doing this was not simply to recast the formalism of SHAKE using a different representation, but to provide some physical insight for the use of constraints in molecular dynamics simulations where bond stress is as important as position and velocity [11, 12]. It is interesting to note that this class of simulations motivated the idea of using forces along bonds from previous timesteps to calculate a very good estimate of each constraint force.

APPENDIX 1: Geometric Interpretation of Constraint–Force Corrections at each Iteration

The corrections $(\Delta \mathbf{F}_{ij}^n)^k$, $k = 1, ..., N$ to the estimated constraint force can be given a geometric interpretation by considering the relative motion of two particles associated with a particular bond of length l_0. The motion of a particle of mass m_i

relative to a particle of mass m_j is equivalent to the motion of a particle of mass $\mu = m_i m_j/(m_i + m_j)$ moving with respect to a fixed point in space in the center-of-mass coordinate system. We therefore consider the force required to constrain the motion of a particle of mass μ to a fixed orbit about a point in space. In the following discussion, we have dropped the superscripts i and n for clarity.

We define $\boldsymbol{l}_\varepsilon(t)$ to be the displacement of the particle from the fixed point at time t, $\boldsymbol{l}_{nc}(t+\delta t)$ to be the displacement of the particle at time $t+\delta t$ in the absence of the constraint, and $\boldsymbol{l}_0(t+\delta t)$ to be the displacement at time $t+\delta t$ in the presence of the constraint. See Fig. (A1). Note that in principle $|\boldsymbol{l}_\varepsilon| = |\boldsymbol{l}_0|$, but in actual simulations numerical errors destroy this equality. The force we seek is that required to correct for the displacement $\Delta\boldsymbol{l}_c$, where

$$\Delta\boldsymbol{l}_c = \boldsymbol{l}_{nc} - \boldsymbol{l}_0. \tag{A1}$$

The displacement $\Delta\boldsymbol{l}_c$ breaks naturally into two parts,

$$\Delta\boldsymbol{l}_c = \Delta\boldsymbol{l}_{c1} + \Delta\boldsymbol{l}_{c2}. \tag{A2}$$

First, the relative displacement in the absence of the constraint is given by

$$\Delta\boldsymbol{l}_\varepsilon = \boldsymbol{l}_{nc} - \boldsymbol{l}_\varepsilon. \tag{A3}$$

The projection of $\Delta\boldsymbol{l}_\varepsilon$ on $\Delta\boldsymbol{l}_c$ is

$$\Delta\boldsymbol{l}_{c1} \doteq \frac{\Delta\boldsymbol{l}_\varepsilon \cdot \boldsymbol{l}_\varepsilon}{l_\varepsilon^2}\,\boldsymbol{l}_\varepsilon. \tag{A4}$$

The remaining distance to be corrected, $\Delta\boldsymbol{l}_{c2}$, is the difference of $\boldsymbol{l}_\varepsilon(\delta t)$ and the projection of $\boldsymbol{l}_0(t+\delta t)$ onto $\boldsymbol{l}_\varepsilon(t)$, i.e.,

$$\Delta\boldsymbol{l}_{c2} = \boldsymbol{l}_\varepsilon - \boldsymbol{l}_c. \tag{A5}$$

Note that a distance-constraining force can only correct radial displacements, not tangential displacements such as $\Delta\boldsymbol{l}_\perp$. Substituting the relation $\boldsymbol{l}_c = \boldsymbol{l}_0 - \Delta\boldsymbol{l}_\perp$ into Eq. (A5) and summing Eqs. (A4) and (A5), we obtain

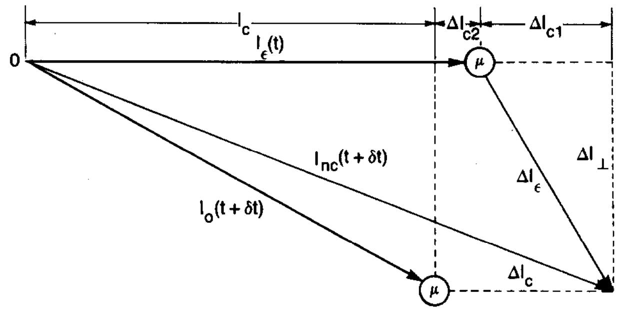

FIG. A1. Geometric description of the various terms of the constraint force function Eq. (9).

$$\Delta \mathbf{l}_c = \mathbf{l}_\varepsilon \left[1 + \frac{\Delta \mathbf{l}_\varepsilon \cdot \mathbf{l}_\varepsilon}{l_\varepsilon^2} - \sqrt{\frac{l_0^2}{l_\varepsilon^2} - \frac{(\Delta l)_\varepsilon^2}{l_\varepsilon^2} + \frac{(\mathbf{l}_\varepsilon \cdot \Delta \mathbf{l}_\varepsilon)^2}{l_\varepsilon^4}} \right]. \tag{A6}$$

The expression in brackets is identical to that for a_{ij} given in Eq. (10).

Since the constraint forces are directly proportional to the distances to be corrected,

$$\Delta \mathbf{F}_{ij} = \Delta \mathbf{F}_{c1} + \Delta \mathbf{F}_{c2} = \frac{\mu}{(\delta t)^2} \Delta \mathbf{l}_{c1} + \frac{\mu}{(\delta t)^2} \Delta \mathbf{l}_{c2}. \tag{A7}$$

From the discussion and Fig. (A1), we see that the force $\Delta \mathbf{F}_{c1}$ counteracts the external forces that cause deviations from the fixed bond length. This term may be isolated to monitor the stress on the bond at each timestep. The force $\Delta \mathbf{F}_{c2}$ gives the particle a trajectory consistent with the rigid bond constraint and also corrects deviations due to accumulated numerical errors and incomplete satisfaction of constraints at the previous timestep.

Acknowledgments

This work was supported by the Physics Division and the Biological Sciences Division of the Office of Naval Research and by the Naval Research Laboratory. MN was supported in in part by ONR Contract N0001487WX24087. The authors thank Dr. Robert Junker for his continued support and Drs. Robert Wyatt and Herschel Rabitz for their encouragement. Critical readings by Dr. J. L. Dote and Mr. J. H. Dunn are gratefully acknowledged.

References

1. C. Ciccotti and J. P. Ryckaert, *Comput. Phys. Rep.* **3**, 345 (1986).
2. J. P. Ryckaert, G. Ciccotti, and H. J. C. Berendsen, *J. Comput. Phys.* **23**, 327 (1977).
3. W. F. van Gunsteren and H. J. C. Berendsen, *Mol. Phys.* **34**, 1311 (1977).
4. G. Ciccotti, M. Ferrario, and J. P. Ryckaert, *Mol. Phys.* **47**, 1253 (1982).
5. J. P. Ryckaert, *Mol. Phys.* **55**, 549 (1985).
6. R. Edberg, D. J. Evans, and C. P. Morris, *J. Chem. Phys.* **84**, No. 12, 6933 (1986).
7. D. J. Evans, W. G. Hoover, B. H. Failor, B. Moran, and A. J. C. Ladd, *Phys. Rev. A* **28**, 1016 (1983).
8. M. K. Memon, R. W. Hockney, and S. K. Mitra, *J. Comput. Phys.* **43**, 345 (1981).
9. K. Singer, A. J. Taylor, and J. V. L. Singer, *Mol. Phys.* **33**, 1757 (1977).
10. C. Hoheisel, R. Vogelsang, and M. Schoen, *Comput. Phys. Commun.* **43**, 217 (1987).
11. S. G. Lambrakos, E. S. Oran, J. P. Boris, and R. H. Guirguis, in *Proceedings, 1987 Topical Conference: Shock Waves in Condensed Matter, Monterey, California, June 20, 1987.*
12. S. G. Lambrakos, M. Peyrard, E. S. Oran, and J. P. Boris, *Phys. Rev. B*, in press.
13. R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles* (McGraw-Hill, New York, 1981).
14. M. Amini, J. W. Eastwood, and R. W. Hockney, *Comput. Phys. Commun.* **44**, 83 (1987).
15. E. Isaacson and H. B. Keller, *Analysis of Numerical Methods* (Wiley, New York, 1966), Chap. 3.
16. P. S. Y. Cheung and J. G. Powles, *Mol. Phys.* **30**, 921 (1975).
17. J. Boris, *J. Comput. Phys.* **66**, 1 (1986).
18. S. G. Lambrakos and J. P. Boris, *J. Comput. Phys.* **73**, 183 (1987).