JOURNAL OF COMPUTATIONAL PHYSICS 52, 24–34 (1983)

# Rattle: A "Velocity" Version of the Shake Algorithm for Molecular Dynamics Calculations

HANS C. ANDERSEN

*Department of Chemistry, Stanford University, Stanford, California 94305*

Received September 14, 1982

An algorithm, called RATTLE, for integrating the equations of motion in molecular dynamics calculations for molecular models with internal constraints is presented. The algorithm is similar to SHAKE, which is one of the standard methods for performing such calculations. RATTLE calculates the positions and velocities at the next time from the positions and velocities at the present time step, without requiring information about the earlier history. Like SHAKE, it is based on the Verlet algorithm and retains the simplicity of using Cartesian coordinates for each of the atoms to describe the configuration of a molecule with internal constraints. RATTLE guarantees that the coordinates and velocities of the atoms in a molecule satisfy the internal constraints at each time step. RATTLE has two advantages over SHAKE. On computers of fixed precision, it is of higher precision than SHAKE. Since it deals directly with the velocities, it is easier to modify RATTLE for use with the recently developed constant temperature and constant pressure molecular dynamics methods and with the nonequilibrium molecular dynamics methods that make use of rescaling of the atomic velocities.

## I. INTRODUCTION

Various types of models have been used to perform molecular dynamics calculations for molecular fluids, for example, vibrating models [1, 2], rigid models [3], and flexible models with internal constraints [4]. Calculations for vibrating models are straightforward using Cartesian coordinates for each of the nuclei. For rigid models, either Euler angles [3] or quaternions [5] can be used to represent the rotational degrees of freedom. An alternative for rigid molecules is to use the methods devised by Ryckaert *et al.* [4], the most familiar of which is called SHAKE. SHAKE retains the simplicity of Cartesian coordinates and avoids many of the complications of Euler angles and quaternions, while incorporating the effects of the constrained geometry of the molecule. SHAKE can also be used for flexible molecules with internal constraints.

SHAKE is based on the Verlet algorithm [6], which is also known as the explicit central difference method [7]. The Verlet algorithm as usually described and implemented has several drawbacks [7, 8]. The velocities of the atoms are not among the variables used in integrating the equations of motion, and they can be obtained only with extra effort or storage. It is difficult to start the algorithm with previously



0021-9991/83 $3.00

chosen values of the coordinates and velocities. It is difficult to change the time step and continue the calculation. It is easy to lose precision because quantities of very different magnitude are added in this algorithm [7].

Since the velocities are not among the variables in the algorithm, it is difficult to implement some recently developed molecular dynamics methods using the Verlet algorithm. It is a complicated matter to implement the stochastic collisions required for the constant temperature molecular dynamics method [9]. In the constant pressure molecular dynamics method [9], the equations of motion contain the velocities, and the Verlet algorithm is unable to solve such equations. Some of the techniques used in nonequilibrium molecular dynamics calculations [10] require continuous or periodic rescaling of the velocities, and this is clumsy to implement within the Verlet algorithm.

It is possible to incorporate the effect of impulsive stochastic collisions into the Verlet algorithm using the method of Stratt *et al.* [11]. This requires in principle some loss of accuracy for those time steps in which the impulsive collisions take place, but in practice there is no problem with this. Van Gunsteren and Berendsen [12] have devised an algorithm for constrained dynamics in which the velocities appear explicitly and satisfy the constraints, but it is of the predictor–corrector type rather than the Verlet type.

For atomic fluids and vibrating molecular models, the drawbacks discussed above can be very easily overcome by use of the "velocity" version of the Verlet algorithm [8]. This is merely a different formulation of the same approximate solution of the equations of motion, in which the velocities are explicitly calculated as part of solving the equations of motion. As far as we can tell, it is not possible to express SHAKE in an analogous way. In this paper we present a generalization of SHAKE that has the advantages of the velocity version of the Verlet algorithm. We call this new algorithm RATTLE. It can be described as a generalization of the Verlet algorithm to the case of molecules with internal constraints in which the velocities and positions at one time $t$ are used to calculate the velocities and positions at the time $t + h$. At each time $t$, the values of the positions satisfy the internal constraints to within any desired accuracy (limited only by the precision of the calculations). In addition, the velocities satisfy the constraints to within any desired accuracy, unlike in SHAKE.

## II. Derivation of the Algorithm RATTLE

In this section we will discuss the usual version of the Verlet algorithm and the difficulties associated with its use. Then we discuss the velocity version of the Verlet algorithm and its advantages. Next we discuss SHAKE and the reason why it can not be simply converted to the velocity form. Then we derive RATTLE.

Consider a differential equation of the form

$$\ddot{r}(t) = f[r],$$

where $r$ denotes the set of Cartesian coordinates necessary to specify the configuration of the system. The Verlet algorithm for solving the equation is

$$r(t+h) = 2r(t) - r(t-h) + h^2 f[r(t)]. \tag{2.1}$$

To solve the differential equation, one starts with a value of $r(0)$ and $r(h)$ and calculates each succeeding value of $r$ from the two preceding values using Eq. (2.1). Equation (2.1) is correct except for errors of order $h^4$. This is the so-called local error [7]. When used in an iterative procedure to integrate the equations of motion for a finite time interval, the error in the calculated solution at the end of the finite interval is of order $h^2$. This is the so-called global error.

One problem with this procedure is that the calculation involves adding a term of order $h^2$ to terms of order $h^0$. Thus only a few significant figures of the calculated force $f[r(t)]$ are actually utilized, and the cumulative effect can be a significant loss of precision [7]. The actual magnitude of the error introduced varies with the size of $h$ and the precision of the computer. Fortunately, there are simple ways to avoid this error that involve no additional computation. One way is discussed by Dahlquist and Björck [7]; the other is the velocity version of the Verlet algorithm to be discussed below [8].

A second problem with the usual version of the Verlet algorithm is that the velocities do not appear explicitly in the algorithm. This leads to difficulties in certain applications mentioned in the introduction. To evaluate the velocity in the Verlet algorithm, the following approximation can be used:

$$\dot{r}(t) = [r(t+h) - r(t-h)]/2h. \tag{2.2}$$

This makes an error of order $h^2$. Thus, the velocity can be obtained from the solution of the Verlet algorithm, but note that the velocity at time $t$ can be obtained only after the position at time $t+h$ has been obtained. This makes it difficult to implement stochastic collisions for the equilibration of the temperature and impossible to use this method to solve differential equations, such as those arising in the constant pressure method, in which the accelerations depend upon the velocities as well as the positions.

The velocity version of the Verlet algorithm [8] eliminates these two types of problems. It is

$$r(t+h) = r(t) + h\dot{r}(t) + h^2 f[r(t)]/2, \tag{2.3}$$

$$\dot{r}(t+h) = \dot{r}(t) + h[f[r(t)] + f[r(t+h)]]/2. \tag{2.4}$$

The equations are equivalent to the Verlet algorithm (2.1) plus Eq. (2.2), as can most easily be seen by deriving (2.3) and (2.4) directly from (2.1) and (2.2) by elementary algebraic methods. The local error in each of these equations is of order $h^3$. (One possible point of confusion arises in discussion of the local error of the Verlet algorithm. When the Verlet algorithm is expressed in its usual form, Eq. (2.1), as an equation for $r(t+h)$ in terms of $r(t)$ and $r(t-h)$, the local error is of order $h^4$.

However, when the velocity version of the Verlet algorithm is used, as in Eqs. (2.3) and (2.4), the local error is of order $h^3$. Nevertheless, both formulations are exactly equivalent, and their global errors are the same, namely, of order $h^2$.)

Equations (2.3) and (2.4) allow one to start with values of the positions and velocities at time $t$ and calculate the positions and velocities at time $t + h$. Iteration of this procedure allows an entire trajectory to be calculated. (The global error of the resulting trajectory is of order $h^2$, the same as for the usual form of the Verlet algorithm, as must be the case since the two methods are equivalent.) This procedure explicitly involves the velocities, so stochastic collisions can easily be implemented. For example, a stochastic collision or a rescaling of the velocity at time $t$ is accomplished merely by changing the velocity at time $t$ before calculating any of the quantities for time $t + h$. The procedure also allows differential equations to be solved with greater accuracy on computers of fixed precision. The reasons for the increased precision is the fact that in both (2.3) and (2.4) quantities of order $h^0$ are combined with quantities of order $h^1$, unlike the procedure in (2.1). The memory storage requirements of the two algorithms are the same, namely, $3N$ locations are required for $N$ degrees of freedom. The velocity form of the Verlet algorithm can easily be generalized to the case of velocity dependent accelerations in a number of ways [13].

The SHAKE algorithm, introduced by Ryckaert *et al.* [4], is a procedure for integrating the equations of motion for molecules with internal constraints (e.g., fixed internuclear distances or fixed bond angles) while retaining the use of Cartesian coordinates as the dynamical variables. When internal constraints are present, the equations of motion can be expressed as

$$\ddot{r}(t) = f[r(t)] + g[r(t), \dot{r}(t)],$$

which $f$ includes the physical forces of intermolecular and intramolecular interaction and $g$ includes the forces associated with the constraints. The constraint forces $g$ in general depend on all the details of the mechanical state of the system. Their functional form depends on the nature of the constraints. They contain time-dependent Lagrange multipliers.

The Verlet algorithm for this differential equation is

$$r(t + h) = 2r(t) - r(t - h) + h^2[f[r(t)] + g[r(t), \dot{r}(t)]].$$

A major problem associated with using this equation is that even if the exact $g$ were known and used, the intramolecular constraints would eventually be violated due to the fact that the algorithm is not exact. Rychaert *et al.* [4] noticed that this problem could be solved by not using the exactly correct $g$ but by using an approximation for $g$ that requires that $r(t + h)$ satisfy the constraints exactly (or to within a desired accuracy). An important feature of this approximation is that it makes errors of the same order as the local error of the Verlet algorithm. The requirements can be satisfied by a proper choice of the time-dependent Lagrange multipliers. Ryckaert *et al.* derived an iterative procedure for finding these unknowns in such a way that the

new configuration at $t + h$ satisfies the constraints. Thus the SHAKE algorithm can be written as

$$r(t + h) = 2r(t) - r(t - h) + h^2[f[r(t)] + g_S(t)],$$

where $g_S$ is the SHAKE approximation for $g$.

SHAKE has the same disadvantages as those associated with the Verlet algorithm. It might be thought that a "velocity version" of SHAKE could eliminate these disadvantages. However, such a formulation of the equations of motion for constrained dynamics would give

$$r(t + h) = r(t) + h\dot{r}(t) + h^2[f[r(t)] + g[r(t), \dot{r}(t)]]/2, \tag{2.5}$$

$$\dot{r}(t + h) = \dot{r}(t) + h[f[r(t)] + g[r(t), \dot{r}(t)] + f[r(t + h)] + g[r(t + h), \dot{r}(t + h)]]/2. \tag{2.6}$$

Suppose we know $r(t)$ and $\dot{r}(t)$ and we want to solve these equations by iteration using the ideas of SHAKE to evaluate the unknown forces $g$. Then using (2.5) we could calculate $r(t + h)$ by replacing $g[r(t), \dot{r}(t)]$ by an approximation that made $r(t + h)$ satisfy the constraints. But then we cannot use (2.6) to get $\dot{r}(t + h)$ because we have no way of evaluating the second $g$ that appears in that equation. In other words, by (2.5), we need to know $\dot{r}(t)$ before beginning the calculation of $g(t)$, but by (2.6) we need to know $g(t + h)$ before calculating $\dot{r}(t + h)$. This is inconsistent with a simple iterative scheme.

There is a straightforward way of eliminating this difficulty. There is no need that the same approximation for $g$ be used in the position equation as in the velocity equation. This suggests the following procedure. Suppose we know the positions, forces, and velocities at time $t$. Using (2.5) we calculate $r(t + h)$ by choosing the $g$ that appears in this equation so that $r(t + h)$ satisfies the constraints exactly (or to within a desired precision). Thus we replace (2.5) by

$$r(t + h) = r(t) + h\dot{r}(t) + h^2[f[r(t)] + g_{RR}(t)]/2. \tag{2.7}$$

Knowing $r(t + h)$, we can calculate $f[r(t + h)]$. Then using (2.6) we calculate $\dot{r}(t + h)$ by choosing the second $g$ that appears in this equation so that the resulting $\dot{r}(t + h)$ satisfies the time dericatives of the constraints exactly (or within a desired tolerance). Thus we replace (2.6) by

$$\dot{r}(t + h) = \dot{r}(t) + h[f[r(t)] + g_{RR}(t) + f[r(t + h)] + g_{RV}(t)]/2. \tag{2.8}$$

This algorithm, which will be called RATTLE, makes two separate approximations, $g_{RR}$ and $g_{RV}$, for the forces associated with the constraints. As a result, it is possible to require that both the positions and the velocities satisfy the constraints. The detailed equations for RATTLE are given in Appendix A.

## III. Properties of the Algorithm RATTLE

RATTLE, defined by Eqs. (2.7) and (2.8) and the conditions for the choice of $g_{RR}(t)$ and $g_{RV}(t)$, is an algorithm for integrating the equations of motion for molecules subject to intramolecular constraints. Its important characteristics are: (1) the positions and velocities at one time are used to calculate the positions and velocities at the next time, without using information about the positions or velocities at previous times, (2) the coordinates at each point in time satisfy the intramolecular constraints, (3) the velocities at each point in time satisfy the constraints, and (4) the precision with which the calculations are performed is comparable to that of the velocity version of the Verlet algorithm rather than the original version. The algorithm can be initiated by choosing the positions and the velocities at one point in time, and thus it is easy to implement stochastic collisions and a change in the magnitude of the time step.

The global error of RATTLE is of order $h^2$ for small $h$, as in the Verlet algorithm for unconstrained dynamics and in SHAKE. The proof of this is in Appendix B. Among the implications of this is that energy along a calculated trajectory is conserved only to within errors of order $h^2$.

We have performed some test calculations on the simple problem of a pendulum swinging in a gravitational potential in two dimensions, using both RATTLE and SHAKE. The numerical work verifies that the global errors in the energy, the coordinates, and the velocities are quadratic in the time step used in the numerical integration. The coefficients of the quadratic errors are similar in magnitude for the two algorithms. The method has also been programmed for molecular dynamics calculations for water [14], and it appears to work satisfactorily.

Since RATTLE requires calculation of two sets of constraint forces, $g_{RR}$ and $g_{RV}$, rather than one as in SHAKE, this aspect of the computation will take longer for RATTLE than for SHAKE. Typically, the calculation of intermolecular forces requires the overwhelming fraction of the computing time, and so in practice RATTLE will be as efficient an algorithm as SHAKE. The two methods require the same amount of storage, namely, approximately $3N$ floating point numbers for $N$ degrees of freedom. RATTLE has several advantages over SHAKE, as discussed in the Introduction, and thus it will be a useful alternative for performing molecular dynamics calculations on molecular models with internal constraints.

## APPENDIX A: Details of RATTLE

We restrict attention to the case in which all the constraints are of the type that require certain pairs of mass points to remain a fixed distance apart. If $i$ and $j$ are such a pair of constrained atoms, we define

$$\sigma_{ij}(\{\mathbf{r}(t)\}) \equiv [\mathbf{r}_i(t) - \mathbf{r}_j(t)]^2 - d_{ij}^2,$$

where $\mathbf{r}_i$ is the position of atom $i$, $m_i$ is the mass of atom $i$, and $d_{ij}$ is the fixed distance between atoms $i$ and $j$. Then the constraint is expressed as

$$\sigma_{ij}(\{\mathbf{r}(t)\}) = 0. \tag{A1}$$

The time derivatives of the constraint equations give constraints on the velocities, namely,

$$[\dot{\mathbf{r}}_i(t) - \dot{\mathbf{r}}_j(t)] \cdot [\mathbf{r}_i(t) - \mathbf{r}_j(t)] = 0. \tag{A2}$$

The equations for constrained dynamics are

$$m_i \ddot{\mathbf{r}}_i = \mathbf{F}_i + \mathbf{G}_i,$$

where $\mathbf{F}_i$ is the force due to intermolecular interactions and the intramolecular interactions not associated with constraints and $\mathbf{G}_i$ is the force on atom $i$ due to the constraints. The latter is given by [4]

$$\mathbf{G}_i = -\sum_j{}' \lambda_{ij}(t) \nabla_i \sigma_{ij},$$

where the prime denotes a summation over only those atoms $j$ that are connected with atom $i$ by a constraint and the $\lambda_{ij}$ are time-dependent Lagrange multipliers associated with the intramolecular forces of the constraints. (Note that $\sigma_{ij} = \sigma_{ji}$ and $\lambda_{ij} = \lambda_{ji}$.)

The RATTLE algorithm involves making approximations for the $\lambda_{ij}$. The first equation of RATTLE, Eq. (2.7), in the present notation is

$$\mathbf{r}_i(t+h) = \mathbf{r}_i(t) + h\dot{\mathbf{r}}_i(t) + (h^2/2m_i)\Big[\mathbf{F}_i(t) - 2\sum_j \lambda_{RRij}(t)\,\mathbf{r}_{ij}(t)\Big], \tag{A3}$$

where $\mathbf{r}_{ij}(t) = \mathbf{r}_i(t) - \mathbf{r}_j(t)$. The quantities $\lambda_{RRij}(t)$ are to be chosen so that the constraint equations (A1) are satisfied at time $t+h$. The second equation of RATTLE, Eq. (2.8), in the present notation is

$$\dot{\mathbf{r}}_i(t+h) = \dot{\mathbf{r}}_i(t) + (h/2m_i)\Big[\mathbf{F}_i(t) - 2\sum_j \lambda_{RRij}(t)\,\mathbf{r}_{ij}(t) \times \mathbf{F}_i(t+h) - 2\sum_j \lambda_{RVij}(t+h)\,\mathbf{r}_{ij}(t+h)\Big]. \tag{A4}$$

The quantities $\lambda_{RVij}(t+h)$ are to be chosen to satisfy the time derivatives of the constraint equations, Eqs. (A2), at time $t+h$. The iterative method of Ryckaert *et al.* [4] is applicable to solving the equations for $\lambda_{RRij}$ and $\lambda_{RVij}$. See Appendix C.

## APPENDIX B: GLOBAL ERROR OF RATTLE

We will prove that the local error of RATTLE, i.e., the error made in each time step, is the same as in the velocity form of the Verlet algorithm for unconstrained dynamics. It will then follow that the global errors of the two methods are the same, namely, of order $h^2$.

First consider Eq. (A3), the first equation of RATTLE. If $\lambda_{RRij}(t)$ were replaced by the exactly correct (but unknown) $\lambda_{ij}(t)$, then this equation would be a second order Taylor series expansion for $\mathbf{r}_i(t+h)$. The resulting $\mathbf{r}(t+h)$ would have errors of order $h^3$, and the constraint equations would be violated by amounts of order $h^3$. Therefore, if the $\lambda_{RRij}(t)$ are chosen to satisfy the constraints exactly, they need differ from the correct $\lambda_{ij}(t)$ by amounts of order $h$,

$$\lambda_{RRij}(t) = \lambda_{ij}(t) + O(h), \tag{B1}$$

since $\lambda_{RRij}$ is multipled by a factor of $h^2$ in (A3). It follows that (A3) predicts positions at $t+h$ that differ from the Taylor series predictions by an amount of order $h^3$, and hence they differ from the correct positions at $t+h$ by an amount of order $h^3$. Therefore, the local error in $r(t+h)$ as calculated by RATTLE is the same as that of the velocity form of the Verlet algorithm, namely, of order $h^3$.

Next consider Eq. (A4). If we replace

$$2 \sum_j{}' \lambda_{RRij}(t)\, \mathbf{r}_{ij}(t)$$

in this equation by the following equivalent quantity

$$2 \sum_j{}' \lambda_{ij}(t)\, \mathbf{r}_{ij}(t) + 2 \sum_j{}' [\lambda_{RRij}(t) - \lambda_{ij}(t)]\, \mathbf{r}_{ij}(t+h)$$

$$+ 2 \sum_j{}' [\lambda_{RRij}(t) - \lambda_{ij}(t)][\mathbf{r}_{ij}(t) - r_{ij}(t+h)]$$

(A4) becomes

$$\dot{\mathbf{r}}_i(t+h) = \dot{\mathbf{r}}_i(t) + (h/2m_i) \times \Bigg[ \mathbf{F}_i(t) - 2 \sum_j{}' \lambda_{ij}(t)\, \mathbf{r}_{ij}(t)$$

$$+ \mathbf{F}_i(t+h) - 2 \sum_j{}' (\lambda_{RVij}(t+h) + \lambda_{RRij}(t) - \lambda_{ij}(t))\, \mathbf{r}_{ij}(t+h)$$

$$- 2 \sum_j{}' [\lambda_{RRij}(t) - \lambda_{ij}(t)][\mathbf{r}_{ij}(t) - \mathbf{r}_{ij}(t+h)]. \tag{B2}$$

Note that the last sum in this equation contributes an amount of order $h^3$ to $\dot{\mathbf{r}}_i$,

because of (B1). If this last term were neglected and if $\lambda_{RVij}(t+h)$ were chosen so that

$$\lambda_{RVij}(t+h)+\lambda_{RRij}(t)-\lambda_{ij}(t)=\lambda_{ij}(t+h),$$

then (B2) would be a Taylor series for $\dot{\mathbf{r}}_i$ that would be correct to order $h^2$. The resulting $\dot{\mathbf{r}}_i$ would contain errors of order $h^3$ and would fail to satisfy the time derivatives of the constraints, namely (A2), by an amount of order $h^3$. Restoration of the last sum in (B2) introduces additional errors of order $h^3$. By making changes of order $h^2$ to $\lambda_{RVij}(t+h)$, the constraints could be satisfied exactly, Therefore the $\lambda_{RVij}(t+h)$ that make the velocities satisfy the constraints exactly are given by

$$\lambda_{RVij}(t+h)=\lambda_{ij}(t+h)+\lambda_{ij}(t)-\lambda_{RRij}(t)+O(h^2).$$

When this is substituted into (A3) it is seen that the RATTLE velocities differ from the exact Taylor series expression for the velocities by an amount of order $h^3$. Therefore, the local error in $\dot{\mathbf{r}}(t+h)$ generated by the RATTLE algorithm is of order $h^3$, which is the same as that of the velocity form of the Verlet algorithm for unconstrained dynamics.

## APPENDIX C: Iterative Procedure for RATTLE Calculations

Suppose that the positions, velocities, and intermolecular forces are known at time $t$, and we wish to calculate the corresponding quantities for the time $t+h$. This can be done by a straightforward modification of the iterative procedure of Ryckaert *et al.* [4] for SHAKE.

Let us define

$$g_{ij}=h\lambda_{RRij}(t),$$

$$k_{ij}=h\lambda_{RVij}(t+h),$$

$$\mathbf{q}_i=\dot{\mathbf{r}}_i(t)+(h/2m_i)\,\mathbf{F}_i(t)-(1/m_i)\sum_j{}' g_{ij}\mathbf{r}_{ij}(t).$$

Then Eqs. (A3) and (A4) can be expressed as

$$\mathbf{r}_i(t+h)=\mathbf{r}_i(t)+h\mathbf{q}_i,$$

$$\dot{\mathbf{r}}_i(t+h)=\mathbf{q}_i+(h/2m_i)\,\mathbf{F}_i(t+h)-(1/m_i)\sum_j{}' k_{ij}\mathbf{r}_{ij}(t+h).$$

First we solve for the $\mathbf{q}_i$ by interation. To start, we let

$$\mathbf{q}_i=\dot{\mathbf{r}}(t)+(h/2m_i)\,\mathbf{F}_i(t),\qquad i=1,\ldots,N.$$

At this point the iterative loop begins. Pick a constraint. Suppose it involves atoms $i$ and $j$. Let

$$\mathbf{s} = \mathbf{r}_i(t) + h\mathbf{q}_i(t) - \mathbf{r}_j(t) - h\mathbf{q}_j(t).$$

Then $\mathbf{s}$ is the current approximation for the vector displacement of atoms $i$ and $j$. If $|\mathbf{s}|^2 - d_{ij}^2$ differs from zero by an amount less than an acceptable tolerance, go to the beginning of the iterative loop and pick a new constraint. If not, then we want to find corrections for $\mathbf{q}_i$ and $\mathbf{q}_j$ to make the constraint be satisfied more closely. Let

$$\mathbf{r}_i^T = \mathbf{r}_i(t) + h[\mathbf{q}_i - g\mathbf{r}_{ij}(t)/m_i]$$

and

$$\mathbf{r}_j^T = \mathbf{r}_j(t) + h[\mathbf{q}_j + g\mathbf{r}_{ij}(t)/m_j].$$

These are the new values for $\mathbf{r}_i(t+h)$ and $\mathbf{r}_j(t+h)$, when the corrections proportional to $g$ are made to $\mathbf{q}_i$ and $\mathbf{q}_j$. We want to choose $g$ so that

$$|\mathbf{r}_i^T - \mathbf{r}_j^T|^2 = d_{ij}^2.$$

Solving for $g$ we find

$$g = (s^2 - d_{ij}^2)/2h[\mathbf{s} \cdot \mathbf{r}_{ij}(t)](m_i^{-1} + m_j^{-1})$$

where we have neglected quantities of order $g^2$. Then we replace $\mathbf{q}_i$ by the old value of $\mathbf{q}_i$ minus $g\mathbf{r}_{ij}(t)/m_i$ and replace $\mathbf{q}_j$ by the old value of $\mathbf{q}_j$ plus $g\mathbf{r}_{ij}(t)/m_j$, go to the beginning of the iterative loop, and choose a new constraint. This iterative procedure is continued until all the constraints are satisfied to within the acceptable tolerance.

The procedure converges to the correct result. At each stage of the iterative procedure, the $\mathbf{q}$'s are corrected by an amount of the proper form, and the procedure terminates when all the constraints on the interatomic distances are satisfied to within the desired accuracy.

Now that $\mathbf{q}_i$ and $\mathbf{r}_i(t+h)$ are known for all $i$, the forces at time $t+h$ can be calculated. Just before doing this, the positions at time $t+h$ should be placed in the memory locations that previously held the positions at time $t$, and the $\mathbf{q}_i$, $i = i,..., N$, should be placed in the memory locations that previously held the velocities at time $t$. This allows the algorithm to be implemented using just $3N$ memory locations for $N$ degrees of freedom.

Next we solve for the $\dot{\mathbf{r}}(t+h)$ by iteration. To start, we let

$$\dot{\mathbf{r}}_i(t+h) = \mathbf{q}_i + h\mathbf{F}_i(t+h)/2m_i, \qquad i = 1,..., N.$$

At this point the iterative loop begins. Pick a constraint. Suppose it involves atoms $i$ and $j$. Calculate the dot product of $\mathbf{r}_{ij}(t+h)$ and $\dot{\mathbf{r}}_{ij}(t+h)$. If it differs from zero by less than an acceptable tolerance, then go to the beginning of the iterative loop and

pick another constraint. If it differs from zero by more than the acceptable tolerance, then we want to correct the two velocities, $\dot{\mathbf{r}}_i$ and $\dot{\mathbf{r}}_j$. Let

$$\dot{\mathbf{r}}_i^T = \dot{\mathbf{r}}_i(t+h) - k\mathbf{r}_{ij}(t+h)/m_i$$

and

$$\dot{\mathbf{r}}_j^T = \dot{\mathbf{r}}_j(t+h) + k\mathbf{r}_{ij}(t+h)/m_j.$$

These are the new values of $\dot{\mathbf{r}}_i(t+h)$ and $\dot{\mathbf{r}}_j(t+h)$ when corrections proportional to $k$ are made. We want to choose $k$ so that $\dot{\mathbf{r}}_i^T - \dot{\mathbf{r}}_j^T$ is perpendicular to $\mathbf{r}_{ij}(t+h)$. This leads to the following choice:

$$k = \mathbf{r}_{ij}(t+h) \cdot [\dot{\mathbf{r}}_i(t+h) - \dot{\mathbf{r}}_j(t+h)]/d_{ij}^2(m_i^{-1} + m_j^{-1}).$$

Then we replace $\dot{\mathbf{r}}_i(t+h)$ by $\dot{\mathbf{r}}_i^T$ and $\dot{\mathbf{r}}_j(t+h)$ by $\dot{\mathbf{r}}_j^T$, go to the beginning of the iterative loop, and pick another constraint.

This procedure converges to the correct result. At each stage of the iterative procedure, the $\dot{\mathbf{r}}_i(t+h)$ are corrected by an amount of the proper form, and the procedure terminates when all the constraints on the velocities are satisfied to within the desired accuracy.

## ACKNOWLEDGMENTS

The author would like to thank Professor Bruce Berne and the Chemistry Department of Columbia University for their hospitality during 1981–1982. This work was supported by National Institutes of Health Grant GM-26626.

## REFERENCES

1. G. D. HARP AND B. J. BERNE, *Phys. Rev.* **A2**, (1970), 975.
2. A. RAHMAN, F. H. STILLINGER, AND H. L. LEMBERG, *J. Chem. Phys.* **63** (1975), 5223.
3. A. RAHMAN AND F. H. STILLINGER, *J. Chem. Phys.* **55** (1971), 3336.
4. J.-P. RYCKAERT, G. CICCOTTI, AND H. J. C. BERENDSEN, *J. Comput. Phys.* **23** (1977), 327.
5. D. J. EVANS AND S. MURAD, *Mol. Phys.* **34** (1977), 327.
6. L. VERLET, *Phys. Rev.* **159** (1967), 98.
7. G. DAHLQUIST AND A. BJÖRCK, "Numerical Methods," p. 353, Prentice–Hall, Englewood Cliffs, N.J., 1974.
8. W. C. SWOPE, H. C. ANDERSEN, P. H. BERENS, AND K. R. WILSON, *J. Chem. Phys.* **76** (1982), 637.
9. H. C. ANDERSEN, *J. Chem. Phys.* **72** (1980), 2384.
10. W. G. HOOVER, D. J. EVANS, R. B. HICKMAN, A. J. C. LADD, W. T. ASHURST, AND B. MORAN, *Phys. Rev.* **A22** (1980), 1690, and references cited therein.
11. R. M. STRATT, S. L. HOLMGREN, AND D. CHANDLER, *Mol. Phys.* **42** (1981), 1233.
12. W. F. VAN GUNSTEREN AND H. J. C. BERENDSEN, *Mol. Phys.* **34** (1977), 1311.
13. J. R. FOX AND H. C. ANDERSEN, submitted for publication.
14. RANDALL HALL AND BRUCE J. BERNE, private communication.