Molecular dynamics algorithm for multiple time scales: Systems with long range forces

Mark E. Tuckerman[a] and Bruce J. Berne
Department of Chemistry, Columbia University, New York, New York 10027

Glenn J. Martyna
Department of Chemistry, University of Pennsylvania, Philadelphia, Pennsylvania 19104

(Received 16 October 1990; accepted 23 January 1991)

A frequently encountered problem in molecular dynamics is how to treat the long times that are required to simulate condensed systems consisting of particles interacting through long range forces. Standard methods require the calculation of the forces at every time step. Because each particle interacts with all particles within the interaction range of the potential the longer the range of the potential the larger the number forces that must be calculated at each time step. In this note we present a variant of the RESPA (reference system propagator algorithm), which we developed for handling systems with multiple time scales like disparate mass mixtures. This version of RESPA greatly reduces the number of forces that must be computed at each time step and thereby leads to a dramatic acceleration of such simulations. The RESPA method uses ideas similar to NAPA, an algorithm we invented to treat high frequency oscillators interacting with low frequency bath. The method is based on a choice of a reference system in which the particles interact through short range forces. The reference system is numerically integrated for n time steps δt and the error incurred by using short range forces is corrected by solving a rigorous set of equations once every $\Delta t = n\delta t$. This method reduces the cpu time dramatically. It is shown that this approach and suitable generalizations should be very useful for future simulations of quantum and classical condensed matter systems.

I. INTRODUCTION

Consider a system consisting of N particles in which the particles interact through forces with a cutoff distance R_c. Each particle then feels the forces from $N_c \propto \rho R_c^3$ neighbors and the cpu time required to advance the system one time step δt is proportional to the number of forces, $NN_c/2$, that must be calculated. Clearly the simulation time grows as the cube of the cutoff distance.

In this short note a method for accelerating the simulation of such systems is presented. This method, which is related to the NAPA algorithm that we invented for treating the problem of high frequency oscillators coupled to low frequency oscillators,[1] is a variant of the RESPA algorithm that we invented for treating systems with multiple time scales like disparate mass mixtures.[2] The gist of the method is to define a dynamical reference system involving only short range forces and to derive equations of motion for the deviation, $\delta(t)$, of the coordinates of the system from those of the system containing the full forces. The reference system equations of motion are integrated for n time steps $n\delta t$. The time dependence of the reference system is then fed into the coupled equations for $\delta(t)$ and the resulting equations are numerically integrated for one large time step $\Delta t = n\delta t$. The initial conditions for each large time step are chosen so that this deviation $\delta(t)$ is zero at the start of each new time step with the consequence that the deviation is always kept small. The only approximation in this algorithm springs from the numerical integrator used to integrate the equations of motion of the reference system and the coupled equations. Otherwise the method is self correcting and exact.

It should be noted that as the range of the interaction increases, the ratio of the number of long range forces to the number of short range forces which must be computed will increase. We thus expect that RESPA will grow more efficient as the range of the interaction grows. By studying two systems with different ranges, we verify that this expectation is in accordance with our results.

II. METHOD

To illustrate the method, consider a set of differential equations of the form

$$\ddot{x} = \frac{1}{m} F(x), \tag{2.1}$$

where m and x are, respectively, the mass and positions of the particles in the system. We must solve this set of equations subject to the initial conditions $\{x(0), \dot{x}(0)\}$. To proceed we subdivide the forces $F(x)$ into short and long range components F_s and F_l, respectively. We define a "reference system" equation of motion

$$\ddot{x}_0 = \frac{1}{m} F_s(x_0) \tag{2.2}$$

which must be solved subject to the initial conditions

$$x_0(0) = x(0),$$
$$\dot{x}_0(0) = \dot{x}(0). \tag{2.3}$$

The solution of the reference system equations of motion Eq. (2.2) denoted

[a] Ph.D. student in the Department of Physics, Columbia University.

 0021-9606/91/106811-05$03.00

$$x_0(t) = x_0(t;x(0),\dot{x}(0)). \tag{2.4}$$

The true position of the system deviates from the reference system position. This can be expressed as

$$x(t) = x_0(t) + \delta(t). \tag{2.5}$$

Substitution of this into Eq. (2.1) and elimination of $\ddot{x}_0$ using Eq. (2.2) results in the set of equations

$$\ddot{\delta} = \frac{1}{m}\left[F_s(x_0(t)+\delta) + F_l(x_0(t)+\delta) - F_s(x_0(t))\right]. \tag{2.6}$$

If F_l is derived from the long range part of the potential, it will be slowly varying compared to F_s.

To solve Eq. (2.6) subject to the initial conditions [which follow from Eq. (2.3)]

$$\delta(0) = 0,$$
$$\dot{\delta}(0) = 0 \tag{2.7}$$

we propose the following scheme.

(1) Numerically integrate Eq. (2.2) for a sequence of n small time steps δt ($\Delta t = n\delta t$) generating $x_0(t)$ for $0 \leqslant t \leqslant \Delta t$.

(2) Substitute the numerical solution for $x_0(t)$ into Eq. (2.6).

(3) Solve Eq. (2.6) subject to the initial conditions, Eq. (2.7), for one time step Δt using a suitable integrator to obtain $\delta(\Delta t)$ and $\dot{\delta}(\Delta t)$.

(4) Calculate

$$x(\Delta t) = x_0(\Delta t) + \delta(\Delta t),$$
$$\dot{x}(\Delta t) = \dot{x}_0(\Delta t) + \dot{\delta}(\Delta t). \tag{2.8}$$

This process is repeated using $x(\Delta t)$, $\dot{x}(\Delta t)$ as initial conditions. In general, at each step, the output is used as the initial conditions for the next step. The advantage of this method lies in the resetting of the initial conditions on $\delta(t)$ and $\dot{\delta}(t)$ to 0 at every step. Since $\delta(t)$ and $\dot{\delta}(t)$ never deviate much from 0 in a given step, the force term $F_s(x_0+\delta) - F_s(x_0)$ in Eq. (2.6) is prevented from becoming too large, thus allowing the use of a larger time step in the numerical integration than could be used in the standard integration schemes. It should be emphasized that the equations of motion Eqs. (2.2) and (2.6) for $x_0(t)$ and $\delta(t)$ are exact, i.e., no approximation has been made related to the disparity in range of the forces and that the physical properties of the system (e.g., pressure, internal energy, etc.) will be the same for RESPA as for straightforward integration of the equations of motion.

The integration of Eq. (2.6) can be done by any integrator suitable for equations with explicit as well as implicit time dependence.[3] The Runge–Kutta (or predictor–corrector) method are such an integrators. In particular, the velocity Verlet[3,4] integrator can be adapted straightforwardly for Eq. (2.6), as we will later demonstrate.

The major reason that this method saves so much time is that only at the end of a large time step Δt Eq. (2.2) does one have to compute all of the forces. Thus it is not necessary to recompute the long range forces after every short time step. These have to be updated only when Eq. (2.6) is integrated. This saving in force computations can be dramatic. Let z_s and z_l denote the average number of particles that interact with a particle through short and long range forces, respectively. The number of forces that must be calculated during the n time steps δt during which the reference system is evolving is $nN(z_s/2)$ and the number that must be calculated for the integration of Eq. (2.6) is $Nz_l/2$. Thus the total cpu time for one large time step using the new method, assuming that the calculation of the forces is the time consuming part of the calculation, is proportional to $[nNz_s + Nz_l]$. On the other hand if the calculation is done by standard integrators to compute n steps of length δt moving all of the particles during each cycle the total number of forces that must be calculated is $nN(z_s + z_l)/2$. This leads to the prediction that the ratio of cpu times for the standard method to that of the RESPA method will be

$$r = \frac{n(z_s+z_l)}{(nz_s+z_l)} \approx \frac{n(\rho R_l^3)}{(nz_s+\rho R_l^3)}, \tag{2.9}$$

where R_l is the cutoff distance for the long range force, and ρ is the number density.

It should be noted that

$$\lim_{R_l/R_s \to \infty} r = n. \tag{2.10}$$

Thus the new method will be at best n times faster than straightforward methods. The size of n will be determined by the magnitude of F_l compared to F_s. The longer the range of the potential the smaller F_l will be compared to F_s and the larger will be the number of time steps n for which the reference system trajectory will be close to the true trajectory.

The subdivision of the forces into slow and fast components can be made in a variety of ways. One possibility is to base the subdivision on the WCA approximation. If r_m is the position of the minimum of the pair potential, then the subdivision is

$$V(r) = V_s(r) + V_l(r), \tag{2.11}$$

where

$$V_s(r) = \begin{cases} V(r) - V(r_m), & r \leqslant r_m \\ 0, & r > r_m \end{cases} \tag{2.12}$$

and

$$V_l(r) = \begin{cases} V(r_m), & r < r_m \\ V(r), & r \geqslant r_m \end{cases} \tag{2.13}$$

so that the corresponding forces are

$$F_s(r) = \begin{cases} F(r), & r \leqslant r_m \\ 0, & r > r_m \end{cases} \tag{2.14}$$

and

$$F_l(r) = \begin{cases} 0, & r < r_m \\ F(r), & r \geqslant r_m \end{cases}. \tag{2.15}$$

As will be seen in the foregoing, the choice of subdivision leads to a considerable reduction in cpu time over standard methods, but may not be optimum because the long range force can still be large in the neighborhood of r_m. In this eventuality, n must be chosen small in order to insure accurate integration of the equations of motion.

A more flexible subdivision is to introduce a switching

function $S(r)$ which varies monotonically between 1 and 0 as r increases and to express the interparticle force as

$$F(r) = S(r)F(r) + (1 - S(r))F(r). \tag{2.16}$$

Then the short and long range components of the force are taken to be

$$F_s(r) = S(r)F(r) \tag{2.17}$$

and

$$F_l(r) = (1 - S(r))F(r). \tag{2.18}$$

One can vary the position of the inflection point and width of $S(r)$ so as to minimize the cpu time for the given model.

Let us now consider the implementation of this method using the velocity Verlet integrator. Although the discussion outlined before is perfectly acceptable it is possible to improve it. It would be useful to have the algorithm satisfy the following two requirements:

(1) For $F_s \to 0$ the algorithm should reduce to the standard velocity Verlet algorithm for F_l using a time step $\Delta t = n\delta t$.

(2) The method should work for long range forces even when they are large, but slowly varying.

The method as outlined does not satisfy condition (1) and becomes inefficient when the long range force is large. An algorithm that satisfies these two conditions is the following. Replace the reference system equation, Eq. (2.2), by

$$\ddot{x}_0(t) = \frac{1}{m} F_s(x_0(t)) + \frac{1}{m} F_l(x(0)), \tag{2.19}$$

where the last term is constant in time for the integration interval. If the long range force is strong but slowly varying this term will make an important contribution to the reference system's acceleration, thereby allowing a larger choice of n for the same energy conservation. Following the same steps leading to Eq. (2.6) leads to

$$\ddot{\delta} = \frac{1}{m} [F_s(x_0(t) + \delta) - F_s(x_0(t)) + F_l(x_0(t) + \delta(t)) - F_l(x(0))], \tag{2.20}$$

an equation which contains the constant $F_l(x(0))$. One then carries out the same proceedure outlined following Eq. (2.6). When the velocity Verlet integrator is used to solve this new set of equations it is a simple matter to show that for $F_s = 0$ condition 1 is satisfied. It is also a simple matter to show that the use of velocity Verlet on Eq. (2.20) gives $\delta(\Delta t) = 0$. In general, velocity Verlet applied to Eqs. (2.19) and (2.20) gives the following iterative equations for computing the trajectory $x(t)$ and $\dot{x}(t)$ from Eq. (2.8):

$$x(\Delta t) = x_0(\Delta t),$$

$$\dot{x}(\Delta t) = \dot{x}_0(\Delta t) + \frac{\Delta t}{2m} [F_s(x(\Delta t)) - F_s(x_0(\Delta t))] + \frac{\Delta t}{2m} [F_l(x(\Delta t)) - F_l(x(0))]. \tag{2.21}$$

III. RESULTS

In the following examples we require a measure of the accuracy of the algorithm. One good measure is the degree to which energy is conserved. In a run of length T we define

$$\Delta\hat{E} = \frac{1}{T} \int_0^T dt \left| \frac{E(t) - E(0)}{E(0)} \right|, \tag{3.1}$$

where $E(0)$ and $E(t)$ are the initial energy and the energy at time t. Clearly, the more accurate the algorithm the smaller will be $\Delta\hat{E}$. We shall compare cup times required for different algorithms for the same values of $\Delta\hat{E}$.

Although we study fluids with different potential models here, the general approach will be to choose a standard integrator (e.g., velocity Verlet) with a given time step δt chosen to give a specified accuracy in energy conservation $\Delta\hat{E}$. For comparison, the new algorithm RESPA is used to integrate the equations of motion for the same system, first using a WCA subdivision and then using a switching function

$$S(r) = \begin{cases} 1, & r < r_c - \lambda \\ 1 + R^2(2R - 3), & r_c - \lambda \leqslant r \leqslant r_c \\ 0, & r_c < r \end{cases}, \tag{3.2}$$

where $R \equiv [r - (r_c - \lambda)]/\lambda$ and where r_c is the cutoff distance and λ is the healing length (the effective width of $S'(r)$). Of course, the choice of switching function is arbitrary, the above having been used in a different context by Watanabe and Reinhardt.[5] For the WCA subdivision n is chosen to give the same energy conservation used in the full velocity Verlet integration. For the switching function subdivision, r_c, λ, and n are chosen to give the same energy conservation and to optimize the cpu time. In each of these simulations, the small time step δt is taken to be the time step used in the velocity Verlet algorithm.

A. Lennard-Jones (12–6) fluids

First we study the well worn problem of the LJ (12–6) potential. Clearly, this is not a long range force problem. Nevertheless, even for this model, we find an impressive reduction in cpu time when RESPA is used.

For this study, the LJ fluid consists of 864 particles interacting with a pairwise additive LJ (12–6) potential

$$\phi(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] \tag{3.3}$$

in a box of size L. The system was equilibrated for each of four thermodynamic states discussed below. Using the standard velocity Verlet integrator with periodic boundary conditions and a cutoff of $\min(3\sigma, L/2)$, the cpu time required to run a total length of real time $T = 300\delta t$ with energy conservation tolerance $\Delta\hat{E} = 10^{-5}$ is determined. The four thermodynamic states studied here are $(\hat{T}, \rho\sigma^3) = (1.0, 0.8)$, $(1.5, 0.9)$, $(2.0, 1.0)$, $(2.5, 1.1)$, where $\hat{T} = k_B T/\epsilon$ is the reduced temperature and $\rho\sigma^3$ is the reduced number density. RESPA, using both the WCA and the switching function subdivisions defined above is compared with the standard velocity Verlet integrator for all of these states. In this comparison the same short time step δt is used in the standard velocity Verlet algorithm, RESPA(WCA) and RESPA(SWITCH) for a given thermodynamic state. The parameter n is chosen to give the specified energy conservation $\Delta\hat{E} = 10^{-5}$ and the time required by

RESPA to run the same real time is determined for each of the two force subdivisions. In addition a search is made of the parameter space of the switching function (namely, r_c and λ) to obtain the smallest cpu time. We find that $r_c = 1.7\sigma$ and $\lambda = 0.1\sigma$ is a very good choice. Although a detailed optimization might lead to a different set of parameters for each thermodynamic state, we find that the results are quite insensitive to the value of λ. The results of this comparison are shown in Table I. It is clear from this study that RESPA can accelerate the simulation of even short range forces by as much as a factor of 3.7. This is a significant improvement. It can be seen from the table that the WCA subdivision improves as the fluid density is increased whereas the switching function subdivision is not strongly density dependent at high densities and gives superior performance at all densities studied.

In the switching function algorithm there is a trade off between two effects. Because the short range force is switched off at greater distances than in the WCA subdivision more forces must be computed for the short time steps therebye increasing the time required to integrate the short cycles over the WCA method. On the other hand the long range forces will be weaker when they are switched on than in the WCA truncation making it possible to achieve the same energy conservation for larger values of n, thereby reducing the number of cpu cycles to achieve a given amount of real time. From Eq. (2.9)

$$\frac{r_{\text{SWITCH}}}{r_{\text{WCA}}} = \frac{n_{\text{SWITCH}}}{n_{\text{WCA}}}\left[\frac{n_{\text{WCA}} z_{s,\text{WCA}} + z_{l,\text{WCA}}}{n_{\text{SWITCH}} z_{s,\text{SWITCH}} + z_{l,\text{SWITCH}}}\right]. \quad (3.4)$$

If the numerator of the bracketed expression $z_{l,\text{WCA}} \gg n_{\text{WCA}} z_{s,\text{WCA}}$ and in the denominator $z_{l,\text{SWITCH}} \gg n_{\text{SWITCH}} z_{s,\text{SWITCH}}$,

$$\frac{r_{\text{SWITCH}}}{r_{\text{WCA}}} \approx \frac{n_{\text{SWITCH}} z_{l,\text{WCA}}}{n_{\text{WCA}} z_{l,\text{SWITCH}}}. \quad (3.5)$$

Since the short range cutoff for the switching function is typically chosen to be larger than the WCA cutoff, $z_{l,\text{WCA}} \geqslant z_{l,\text{SWITCH}}$ and $n_{\text{SWITCH}} \geqslant n_{\text{WCA}}$, and it follows that $r_{\text{SWITCH}}/r_{\text{WCA}} \geqslant 1$ so that the switching function method will be superior to the WCA method.

B. Lennard-Jones (12–3) fluids

To investigate the power of RESPA we study the Lennard-Jones (12–3) potential

TABLE I. Comparison of RESPA with velocity Verlet algorithms for Lennard-Jones (12–6) fluids.

$\hat{T}$	$\rho\sigma^3$	n_{WCA}	r_{WCA}	n_{SWITCH}	r_{SWITCH}
1.0	0.8	3	2.4	6	3.4
1.5	0.9	3	2.5	6	3.3
2.0	1.0	4	3.0	8	3.7
2.5	1.1	5	3.5	8	3.7

TABLE II. Comparison of RESPA with velocity Verlet algorithms algorithms for Lennard-Jones (12–3) fluids.

$\hat{T}$	$\rho\sigma^3$	n_{WCA}	r_{WCA}	n_{SWITCH}	r_{SWITCH}
1.0	0.8	3	2.6	5	3.4
1.5	0.9	3	2.7	5	3.5
2.0	1.0	4	3.4	6	4.1
2.5	1.1	5	4.5	7	4.6

$$\phi(r) = 4\epsilon a_3\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{3}\right], \quad (3.6)$$

a long range potential. The constant a_3 is chosen to give the same well depth ϵ as the (12–6) potential.

As before the system was equilibrated for each of the four same thermodynamic state using the standard velocity Verlet integrator with periodic boundary conditions, only for this system a cutoff of $\min(5\sigma, L/2)$ is used, and the energy conservation tolerance is set at $\Delta\hat{E} = 10^{-6}$. Otherwise, the same comparison is carried out as for the LJ (12–6) system, and the same type of search through the switching function parameter space is done. For this system we find that $r_c = 1.9\sigma$ and $\lambda = 0.17\sigma$ is a very good choice. The results of this comparison are shown in Table II. It is clear from this study that RESPA can accelerate the simulation of this long range system by as much as a factor of 4.3. This is a remarkable saving in cpu time and demonstrates the efficiency of RESPA in handling long range forces.

C. Lennard-Jones (12–1) fluids: Ewald summation

To study the implementation of RESPA with the Ewald summation technique, we consider the Lennard-Jones (12–1) potential given by

$$\phi(r) = 4\epsilon a_1\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)\right]. \quad (3.7)$$

As before, the constant a_1 is chosen so that this potential has the same well depth ϵ as the (12–6) potential. Only the $1/r$ part of this potential is written as an Ewald sum. This gives a force which has the structure

$$F(r) = F^{(\text{real})}(r;\alpha) + F^{(\text{recip})}(r;\alpha,k_{\max}). \quad (3.8)$$

That is, the force consists of a real space part and a reciprocal space sum. We indicate the explicit dependence of these components on the convergence parameter α and the cutoff in reciprocal space denoted $k_{\max}$. There are several possibilities for implementing RESPA with the force in Eq. (3.8). One is to use the switching function on the real space part of the force. This would give a reference system force

$$F_s(r) = S(r)F^{(\text{real})}(r;\alpha^{(s)}) \quad (3.9)$$

and a long range residual force

$$F_l(r) = (1 - S(r))(F^{(\text{real})}(r;\alpha^{(l)}) + F^{(\text{recip})}(r;\alpha^{(l)},k_{\max}^{(l)})). \quad (3.10)$$

Given the different ranges of the reference system force and the residual, we expect that a different convergence parameter and reciprocal space cutoff should be used for each piece, and we indicate this dependence by the s and l super-

TABLE III. Comparison of RESPA with velocity Verlet algorithms for Lennard-Jones (12–1) fluids with Ewald sum.

$\hat{T}$	$\rho\sigma^3$	n_{SWITCH}	r_{SWITCH}
1.0	0.8	5	3.7
1.5	0.9	6	4.0
2.0	1.0	7	4.4
2.5	1.1	8	4.8

scripts on α and $k_{\max}$. The choice of $\alpha^{(s)}$, $\alpha^{(l)}$, $k_{\max}^{(s)}$, and $k_{\max}^{(l)}$ will depend on the system size. Given the large system studied here, we choose $\alpha^{(s)} = \alpha^{(l)} \equiv \alpha$ and $k_{\max}^{(s)} = k_{\max}^{(l)} \equiv k_{\max}$ in Eqs. (3.9) and (3.10). Since the reference system force is purely real space, the use of RESPA becomes extremely efficient.

To illustrate the improvement given by RESPA with Ewald summation, we study the same four thermodynamic states as in the previous two cases using an energy conservation tolerance of $\Delta\hat{E} = 10^{-5}$. The same values of r_c and λ were used as in the (12–3) case, and in addition to these, we choose $k_{\max} = 6$ and $\alpha L = 6$. A cutoff of $\min(5\sigma, L/2)$ was used as in the (12–3) case. The results are summarized in Table III. We see that the timing ratios are more dramatic than for the (12–3) case even though the same cutoff was used. This improvement is clearly due to the fact that in the ordinary Verlet simulations, the reciprocal space sum must be evaluated at every step in addition to the real space part, while in the RESPA simulations, this sum must only be evaluated every n steps when doing the long range part of the force. The larger $k_{\max}$ must be, the more these timing ratios will improve.

IV. CONCLUSION

In this paper we have presented a method which accelerates molecular dynamics simulations of systems with long range forces. This method, which is a variant of the RESPA method, is based on a set of exact equations which greatly reduces the number of force calculations that are required for the simulation. Since it is the computation of the forces that dominates the cpu time required to simulate systems, this reduction in the number of pair forces that have to be evaluated leads to a saving in cpu time. The ratio of the time required for the simulation using this new method to that for a standard method is given by r in Eq. (2.9). For systems with very long range forces $r = n$, where n is the number of time steps for which the short range forces are integrated. The approach taken here is similar in spirit to the method that we presented for solving the problem with a very stiff oscillator buried in a slow fluid, namely, the NAPA algorithm.

We have introduced two of many possible schemes for subdividing the forces into short anfd long range components. We find that an optimized switching function is to be preferred over the WCA subdivision but it would not be surprising to find that a different subdivision scheme works even better than the above. By studying the Lennard-Jones (12–6) and (12–3) potentials, we saw that the cpu saving gained by RESPA increases as the range of the interaction increases.

ACKNOWLEDGMENTS

We wish to thank Dr. Joseph Hautman and Jim Bareman for useful discussions about Ewald summation techniques.

[1] M. Tuckerman, G. Martyna, and B. J. Berne, J. Chem. Phys. **93**, 1287 (1990).
[2] M. Tuckerman, B. J. Berne, and A. Rossi, J. Chem. Phys. **94**, 1465 (1991).
[3] M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids* (Oxford University, Oxford, 1989).
[4] W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson, J. Chem. Phys. **76**, 637 (1982).
[5] M. Watanabe and W. P. Reinhardt, Phys. Rev. Lett. **65**, 3301 (1990).