Molecular Dynamics in Systems with Multiple Time Scales: Reference System Propagator Algorithms

Bruce J. Berne

Department of Chemistry, Columbia University,
New York, NY 10027

Abstract. Systems with multiple time scales, and with forces which can be subdivided into long and short range components are frequently encountered in computational chemistry. In recent years, new, powerful and efficient methods have been developed to reduce the computational overhead in treating these problems in molecular dynamics simulations. Numerical reversible integrators for dealing with these problems called r-RESPA (Reversible Reference System Propagator Algorithms) are reviewed in this article. r-RESPA leads to considerable speedups in generating molecular dynamics trajectories with no loss of accuracy. When combined with the Hybrid Monte Carlo (HMC) method and used in the Jump-Walking and the Smart-Walking algorithms, r-RESPA is very useful for the enhanced sampling of rough energy landscapes in biomolecules.

1 Introduction

Molecular Dynamics (MD) is one of the major tools in the arsenal of computational chemistry and physics. It grew out of attempts to understand the static and dynamical properties of hard sphere fluids and its first appearance[1, 2] was in a form applicable to impulsive forces (1957). Several years later (1964), Rahman extended MD to monoatomic liquids in which the atoms interact pairwise through the Lennard-Jones pair potential.[3] This major development was followed soon after (1968) by the first application of MD to fluids containing diatomic molecules[4, 5] interacting through continuous potentials and then to triatomic molecules[6] (1971). It was these applications of MD to molecules interacting through continuous force fields that set the stage for all subsequent applications of MD in computational chemistry. Several excellent monographs exist which treat the methodology in detail.[7, 8, 9, 10]

One of the problems encountered in applying molecular dynamics to the simulation of complex systems is the presence of both fast and slow degrees of freedom. One must choose a small time step to achieve stable integration of the equations of motion for the fast motion and must then generate a very large number of time steps to achieve sufficient sampling of the slow degrees of freedom. Another major bottleneck is the calculation of the long

Even with fast methods for generating molecular dynamics (MD) or Monte Carlo (MC) trajectories, the problems of sampling conformational states separated by large energy barriers remains an obstacle to progress. This problem raises another and more serious kind of multiple time scale problem – one due to the presence of a rugged energy landscape with the attendant separation of time scales arising from activated barrier crossing. This latter problem will be refered to as the "extrinsic" multiple time scale problem.

In recent years a variety of powerful new molecular dynamics and Monte Carlo methods have been developed to address the "intrinsic" and "extrinsic" multiple time scale respectively. Accurate numerical integrators are required for questions involving real dynamical problems such as transport and energy relaxation. Thus in Sec. **2** we discuss accurate numerical integrators for "intrinsic" multiple time scale problem. On the other hand, in the simulation of biomolecular systems, one is often interested in computing equilibrium averages and thermodynamic quantities. For this purpose, the exact time dependence is not required, since all that is needed is the correct and efficient sampling of the thermally accessible configurations of the system, a problem made difficult by the "extrinsic" multiple time scales connected with the omnipresent energy barriers in systems with rough energy landscapes. A variety of techniques, such as stochastic dynamics, Monte Carlo, Hybrid Monte Carlo, J-Walking etc can be used, some of which are discussed in Sec. **3.** First new methods for generating accurate dynamical trajectories are described and then methods based on inaccurate dynamics for sampling state space in systems with rough energy landscapes are treated.

2 Methods for Dealing with the Intrinsic Multiple Time Scale Problem in Molecular Dynamics

In complex systems the set of fast degrees of freedom arises both from vibrations of stiff bonds or particles with small mass. An example of the latter is the fast vibrational motions of the C-H and 0-H bonds in biomolecules and the 0-H bonds of water. In systems with multiple time scales it is necessary to choose a time step much smaller than the periods of the fastest motions and to recalculate the forces after each small time step. It then requires very long runs to sample the conformational space of the slower degrees of freedom. To bypass this problem some fast degrees of freedom can be eliminated by constraining the length of the stiff bonds.[11] Constrained molecular dynamics suffers from several problems: (a) bond constraints introduce additional angular correlations is torsion angle distribution functions that are not found in the flexible systems in nature; (b) constraints cannot be used to eliminate problems like the fast librational motion of water; (c) the integrators often

methods like Hybrid Monte Carlo[12] which require reversible integrators to insure detailed balance.

In conventional MD the forces are recomputed after each time step. The force calculations account for as much as 95% of the CPU time in an MD simulation. In systems with long range forces, the force computation becomes the major bottleneck to the computation. When using the direct pairwise evaluation, the computational effort required to compute the long-range Coulomb forces on N interacting particles is of order N^2. A variety of strategies, such as the fast multipole method and the particle-particle-mesh Ewald method, have been introduced to reduce the computational effort in calculating the forces. Building on earlier reference system propagator algorithm (RESPA) based integrators,[13, 14, 15, 16] a class of new reversible and symplectic integrators have been invented that greatly reduces the "intrinsic" multiple time scale problem. By using a reversible Trotter factorization of the classical propagator[17] one can generate simple, accurate, reversible and symplectic integrators that allow one to integrate the fast motions using small time steps and the slow degrees of freedom using large time steps.[17] This approach allows one to split the propagator up into a fast part, due to the high frequency vibrations, and slow parts, due to short range, intermediate range, and long range forces, in a variety of ways. These new integrators, called reversible reference system algorithms (r-RESPA), require for the treatment of all-atom force fields no more CPU time than constrained dynamics and often lead to even larger improvements in speed. Although r-RESPA is quite simple to implement, there are many ways to factorize the propagator. A recent paper shows how to avoid bad strategies.[18] Applications of these methods to Car-Parrinello ab *initio* molecular dynamics has resulted in speedups by a factor of approximately five in semiconductor materials.[19, 20, 21] There has been significant progress in recent years to apply these methods to systems of biological relevance.[22, 23, 24, 25, 26]

2.1 Background

As is well known, Molecular Dynamics is used to simulate the motions in many-body systems. In a typical MD simulation one first starts with an initial state of an N particle system $\Gamma = (x_1, \ldots, x_f, p_1, \ldots, p_f)$ where f = 3N is the number of degrees of freedom in the system. After sampling the initial state one numerically solves Hamilton's equations of motion:

$$\begin{aligned} \dot{x}_i &= \frac{\partial H}{\partial p_i} \\ \dot{p}_i &= -\frac{\partial H}{\partial x_i} \end{aligned} \tag{1}$$

we will focus on the Verlet integrator.[27] Over the years this integrator has undergone various extensions and modifications. For example, Andersen *et* al. have introduced the velocity Verlet integrator.[28] In this integrator, the positions $x_i(\Delta t)$ and velocities $\dot{x}_i(\Delta t)$ after one time step At are related to the positions $x_i(0)$ and velocities $v_i(0) = \dot{x}_i(0) = p_i(0)/m$ at the beginning of the time step by:

$$x_i(\Delta t) = x_i(0) + \Delta t \dot{x}_i(0) + \frac{(\Delta t)^2}{2} \frac{F(\{x_i(0)\})}{m}$$

$$\dot{x}_i(\Delta t) = \dot{x}_i(0) + \frac{\mathrm{At}}{2\mathrm{m}} [\mathrm{F}\{\{x_i(0)\}) + F_i(\{x_i(\mathrm{At})\})] \tag{2}$$

for $i = 1, \ldots, 3N$, where $F_i = -\partial U(\{x_i\})/\partial x_i$ is the force on the coordinate x_i. The forces at any time can be computed from the potential function $U(\{x_i\})$ and are functions of all of the position coordinates at that time.

One property of the exact trajectory for a conservative system is that, the total energy is a constant of the motion.[12] Finite difference integrators provide approximate solutions to the equations of motion and for trajectories generated numerically the total energy is not strictly conserved. The exact trajectory will move on a constant energy surface in the 6N dimensional phase space of the system defined by,

$$H(\Gamma) = E. \tag{3}$$

The numerical trajectory will wander off this energy surface. If the trajectory is stable it will wander on an energy shell

$$|E - H(\Gamma)| \leq \Delta E. \tag{4}$$

The smaller the time step At used in the integrator, the more accurate will be the trajectory and the smaller will be the thickness of the energy shell on which the trajectory wanders. If the time step is too large the integrator will generate an unstable trajectory and the energy will diverge after a small number of time steps. This will happen if during a time step the errors in the new positions give rise to very large changes in the forces between particles. Then on the next time step the particles will speed up giving rise to still larger errors in the next positions and to even larger changes in the forces. This situation eventually results in disaster. In general, one must choose time steps sufficiently small that the forces do not change significantly. This means that the time step must be small enough for the fastest motions in the system.

One of the advantages of the Verlet integrator is that it is time reversible and symplectic[30, 31, 32]. Reversibility means that in the absence of numerical round off error, if the trajectory is run for many time steps, say $n\Delta t$, and the velocities are then reversed, the trajectory will retrace its path and after $n\Delta t$ more time steps it will land back where it started. An integrator

map these states to another measurable point set. If the mapping is *symplectic,* the measure of the initial point set will be equal to the measure of the final point set. The mapping then satisfies Liouville's Theorem[12] and conserves the measure in phase space. Thus, like the exact solution of Hamilton's equations of motion, symplectic integrators, such as the Verlet integrator (see Eq. 2) , are reversible and measure conserving. In recent years it has been understood that symplectic integrators are more stable than non-symplectic integrators.[31, 32] It can be shown that dynamics generated by a symplectic integrator will conserve not the true Hamiltonian, but rather a modified, time step dependent Hamiltonian, $\tilde{H}(\Delta t)$ in one dimension and is postulated to do so in many dimension [33]. This theorem guarantees that Eq. 4 will hold for all time, t and that the integrator will be stable.

2.2 Integrators Generated from Factorizing the Classical Propagator

Before discussing the method for handling the problem of multiple time step molecular dynamics, it is useful to show how simple operator algebra can be used to generate reversible integrators.[17] The starting point for this is the definition of the classical Liouvillian, a Hermitian operator on functions of the state variables. The Liouvillian is defined in terms of the Poisson Bracket, $\{, \mathrm{H})$, of whatever it operates on with the Hamiltonian H of the system. In Cartesian coordinates it has the form,

$$iL = \{, H\} = \dot{x}\frac{\partial}{\partial x} + F\frac{\partial}{\partial p_x}, \tag{5}$$

where F is the force ($\mathrm{F} = -\partial V/\partial x$), $V(x)$ is the potential function, $\dot{x} = p_x/m$ is the velocity and p is the momentum. For simplicity of notation we treat only a one dimensional system (one position coordinate and one conjugate momentum); nevertheless it should be recognized that for general systems the Liouvillian involves a sum over all degrees of freedom.

The operator,

$$G(t) \equiv e^{iLt}, \tag{6}$$

is the propagator of the classical motion. Thus the state of the system after one time step At is found by applying the propagator to the initial state so that

$$\begin{pmatrix} x(\Delta t) \\ p(\Delta t) \end{pmatrix} = e^{iL\Delta t} \begin{pmatrix} x(0) \\ p(0) \end{pmatrix}. \tag{7}$$

Now assuming any decomposition of the Liouvillian into two parts,

one can use the reversible Trotter factorization of the propagator to approximate the true propagator,

$$e^{(iL_1+iL_2)\Delta t} = e^{iL_1\Delta t/2} e^{iL_2\Delta t} e^{iL_1\Delta t/2} + O(\Delta t^3). \tag{9}$$

Applying this to an initial state of the system represented by the column vector,

$$\begin{pmatrix} x(0) \\ p(0) \end{pmatrix}, \tag{10}$$

gives the state after one time step At,

$$\begin{pmatrix} x(\Delta t) \\ p(\Delta t) \end{pmatrix} = e^{iL_1 \frac{\Delta t}{2}} e^{iL_2 \Delta t} e^{iL_1 \frac{\Delta t}{2}} \begin{pmatrix} x(0) \\ p(0) \end{pmatrix}. \tag{11}$$

If we subdivide the Liouvillian into the two parts by separating the force and velocity terms,

$$iL_1 = F\frac{d}{\partial p_x} \qquad \text{and} \qquad iL_2 = \dot{x}\frac{d}{dx}, \tag{12}$$

and apply this factorization to the propagator, we obtain:

$$\begin{pmatrix} x(\Delta t) \\ p(\Delta t) \end{pmatrix} = e^{\frac{\Delta t}{2} F \frac{\partial}{\partial p_x}} e^{\Delta t \dot{x} \frac{\partial}{\partial x}} e^{\frac{\Delta t}{2} F \frac{\partial}{\partial p_x}} \begin{pmatrix} x(0) \\ p(0) \end{pmatrix} \tag{13}$$

Each of the factorized operators are displacement operators and can thus be applied seriatim to the initial state vector to give the final solution,

$$\begin{pmatrix} x(\Delta t) \\ p(\Delta t) \end{pmatrix} = \begin{pmatrix} x(0) + \Delta t \dot{x}(0) + \frac{\Delta t^2}{2m} F(x(0)) \\ p(0) + \frac{\Delta t}{2} [F(x(\Delta t)) + F(x(0))] \end{pmatrix}. \tag{14}$$

This procedure is then repeated after each time step. Comparison with Eq. (2) shows that the result is the velocity Verlet integrator and we have thus derived it from a split-operator technique; which is not the way that it was originally derived. A simple interchange of the L_1 and L_2 operators yields an entirely equivalent integrator,

$$\begin{pmatrix} x(\Delta t) \\ p(\Delta t) \end{pmatrix} = \begin{pmatrix} x(0) + \frac{\Delta t}{2m} [p(0) + p(\Delta t)] \\ p(0) + \Delta t F(\left[x(0) + \frac{\Delta t}{2} p(0)\right]) \end{pmatrix}. \tag{15}$$

which by symmetry we call the position Verlet integrator, an integrator of the same accuracy as the velocity Verlet integrator.

An interesting property of these integrators is that the Jacobian of the transformation from the state at time 0 to the time At is

$$\left| \partial(x(\Delta t), p_x(\Delta t)) \right|$$

Thus these integrators are measure preserving and give trajectories that satisfy the Liouville theorem.[12] This is an important property of symplectic integrators, and, as mentioned before, it is this property that makes these integrators more stable than non-symplectic integrators.[30, 33]

By now it should be clear that this kind of operator algebra can be a useful method for generating integrators. We show, in the following, how it can be applied to generate a wide variety of methods for treating the multiple time scale problem.

2.3 Reference System Propagator Algorithms

The aforementioned factorizations of the classical propagator can be used to generate efficient reversible and symplectic integrators for systems with long and short range forces and for systems in which the degrees of freedom can be subdivided into fast and slow subsets. All of the methods described below are called Reference System Propagator Algorithms (RESPA); a name that we gave to our initial attempts to use an underlying reference system propagator for the fast motion. This early effort resulted in non-reversible integrators.[13, 14, 15, 16] If the Liouville operator of the system is decomposed into a "reference system" part, iL_{ref}, and a "correction part", $i\delta L$, as

$$iL = iL_{ref} + i\delta L. \tag{17}$$

Trotter factorization of the propagator then leads to

$$e^{iL\Delta t} = e^{i\delta L\Delta t/2} e^{iL_{ref}\Delta t} e^{i\delta L\Delta t/2} \tag{18}$$

In this context the velocity Verlet integrator is equivalent to taking the reference system to be the dynamical system with all of the forces turned off; that is, the ideal gas system. In some cases the reference system can be solved analytically, and we refer to these methods as the Numerical Analytical Propagator Algorithm (NAPA). The development of symplectic, reversible RESPA (r-RESPA) integration methods grew out of our earlier attempts to devise multiple time scale integrators based on the generation of the dynamics of a reference system and, in principle, exact correction to it.[13, 14, 15, 16] The latter, being non-reversible, guided us in the direction of analyzing the structure of the classical propagator and the use of the symmetric Trotter factorization. In fact in the development of r-RESPA and r-NAPA we have adopted many of many of the strategies used in our earlier non-reversible RESPA (nr-RESPA).[17] All of these r-RESPA integrators are also symplectic. First we treat the problem where there are fast and slow degrees of freedom (or light and heavy particles). Then we treat the case where the forces can be subdivided into short and long range components. Finally, we show how the long and short range force factorizations can be combined with the fast and slow factorization yielding a speedup which is approximately the product of the

2.4 Fast and Slow Processes

In many cases the dynamical system consists of fast degrees of freedom, labeled x, and slow degrees of freedom, labeled y. An example is that of a fluid containing polyatomic molecules. The internal vibrations of the molecules arc often very fast compared to their translational and orientational motions. Although this and other systems, like proteins, have already been treated using RESPA,[17, 34, 22, 23, 24, 25, 26] another example, and the one we focus on here, is that of a system of very light particles (of mass m) dissolved in a bath of very heavy particles (mass M).[14] The positions of the heavy particles are denoted y and the positions of the light particles are denoted by x. In this case the total Liouvillian of the system is:

$$iL = iL_x + iL_y, \tag{19}$$

where

$$\begin{aligned} iL_x &= \dot{x}\frac{\partial}{\partial x} + F_x\frac{\partial}{\partial p_x} \\ iL_y &= \dot{y}\frac{\partial}{\partial y} + F_y\frac{\partial}{\partial p_y}. \end{aligned} \tag{20}$$

With this break up the reversible Trotter factorization of the propagator is

$$G_{xyx}(t) = e^{(iL_x+iL_y)\Delta t} = e^{iL_x\Delta t/2}e^{iL_y\Delta t}e^{iL_x\Delta t/2} + O(\Delta t^3). \tag{21}$$

For the slow (y) motion the time step $\boldsymbol{At}$ may be chosen large whereas for the fast motion this time will be too large. Thus this propagator can be expressed as;

$$G_{xyx}(\Delta t) = G_x(\frac{\Delta t}{2})G_y(\Delta t)G_x(\frac{\Delta t}{2}), \tag{22}$$

where

$$G_y(\Delta t) = e^{\frac{\Delta t}{2}F_y\frac{\partial}{\partial p_y}}e^{\Delta t\dot{y}\frac{\partial}{\partial y}}e^{\frac{\Delta t}{2}F_y\frac{\partial}{\partial p_y}}, \tag{23}$$

$$G_x(\Delta t/2) = \left[e^{\frac{\delta t}{4}F_x\frac{\partial}{\partial p_x}}e^{\frac{\delta t}{2}\dot{x}\frac{\partial}{\partial x}}e^{\frac{\delta t}{4}F_x\frac{\partial}{\partial p_x}}\right]^n, \tag{24}$$

and At $= n\delta t$, where n is a whole number. The r-RESPA integrator involves the following: the heavy particles are integrated for one half of one large time

large time step. A simple example of fortran pseudocode for this is:

$$
\begin{aligned}
&v_h \leftarrow v_h + F_h \tfrac{\Delta t}{4m} \\
&x_h \leftarrow x_h + v_h \tfrac{\Delta t}{2} \\
&v_h \leftarrow v_h + F_h \tfrac{\Delta t}{4m} \\
&\text{do } i = 1, n \\
&\qquad v_l \leftarrow v_l + F_l \tfrac{\delta t}{2m} \\
&\qquad x_l \leftarrow x_l + v_l \delta t \\
&\qquad v_l \leftarrow v_l + F_l \tfrac{\delta t}{2m} \\
&\text{end do} \\
&v_h \leftarrow v_h + F_h \tfrac{\Delta t}{4m} \\
&x_h \leftarrow x_h + v_h \tfrac{\Delta t}{2} \\
&v_h \leftarrow v_h + F_h \tfrac{\Delta t}{4m}
\end{aligned}
$$

where we designate the fast motion by subscript l (standing for light particles) and the slow motion by subscript h (standing for heavy particles).

This procedure is very cost efficient when the fast (or light) particles are the dilute component because then one only has to update the forces on the heavy particles (the expensive part of the computation) every large time step instead of every small time step as would be the case in the straightforward application of the Verlet integrator. For example when applied to a system containing 64 particles of mass 1 dissolved in 800 solvent atoms of mass 100, the CPU time for the full simulation took only slightly longer than it would if the complete system was made up of heavy particles.[14] In contrast, application of the usual Verlet integrator using the small time step required for the light particles but evaluating all the forces after each one of these small time steps required approximately ten times the CPU time used in the RESPA integrator. The same accuracy was achieved in these two different treatments.

Another important application of this strategy was to the vibrational relaxation of a stiff diatomic molecule dissolved in a Lennard-Jones solvent. As is typical of such problems, the frequency of the oscillator can be an order of magnitude or more larger than the typical frequencies found in the spectral density of the solvent. Thus very small time steps are required to to integrate the equations of motion, but because there are very few accepting solvent modes at the frequency of the oscillator, its vibrational relaxation time will be very long, largely occurring by a multiphonon mechanism. In the past it was not practicable to simulate these processes directly. Using a form of r-RESPA modified for the specific case of an oscillator dissolved in a slow solvent, we have been able to reduce the CPU time required for these calculations by factors of ten in many cases making possible the direct simulation

fullerene, speedups of as much as a factor of forty have been obtained.[35] It is important to note that the strategy outlined here is a direct generalization of the strategy we introduced in our original RESPA papers,[13] and is distinct from other attempts to deal with multiple time scales.

2.5 Long and Short Range Forces

Another immediate application of r-RESPA is to the case when the force can be subdivided into a short range part and a long range part. One way for effectuating this break up is to introduce a switching function, $s(x)$ that is unity at short inter-particle separations and 0 at large inter-particle separations. We introduced this strategy in our earlier non-reversible RESPA paper[15] where we expressed the total force as,

$$F(x) = s(x)F(x) + (1 - s(x))F(x) = F_s(x) + F_l(x). \tag{25}$$

The switching function $s(x)$ was taken to be a sigmoidal function (usually a cubic spline) whose inflection point (switching point) and skin-depth can be optimized. The short range force $F_s(x) = s(x)F(x)$ defines the time step to be used in a molecular dynamics calculation. In the velocity Verlet integrator one must compute the full force after each time step. If only the short range force were present, the CPU cost would be small because each particle would only interact with its nearest neighbors. It is the long range force $F_l(x) = (1 - s(x))F(x)$ which is costly to calculate. We introduced this strategy into the r-RESPA propagator factorization,[17] and as with the non-reversible RESPA, we showed that this can significantly reduce the CPU cost of the simulation.

Introducing the above force breakup into the Liouvillian gives,

$$iL = \dot{x}\frac{\partial}{\partial x} + F_s\frac{\partial}{\partial p_x} + F_l\frac{\partial}{\partial p_x} = iL_s + F_l\frac{\partial}{\partial p_x}. \tag{26}$$

The system defined by the Liouvillian L, is called the reference system. Now applying the Trotter factorization to the propagator $\exp(iL_s + F_1\frac{\partial}{\partial p_x})\Delta t$ arising from this subdivision gives the new propagator,[17]

$$G_{lsl}(\Delta t) = e^{\frac{\Delta t}{2}F_l\frac{\partial}{\partial p_x}}e^{iL_s\Delta t}e^{\frac{\Delta t}{2}F_l\frac{\partial}{\partial p_x}}, \tag{27}$$

where with $\mathrm{At} = n\delta t$

Thus the propagator in Eq. (27) produces the following dynamics algorithm:

$$
\begin{aligned}
& v \leftarrow v + F_l \tfrac{\Delta t}{2m} \\
& \text{do } i = 1, n \\
& \qquad v \leftarrow v + F_s \tfrac{\delta t}{2m} \\
& \qquad x \leftarrow x + v\,\delta t \\
& \qquad v \leftarrow v + F_s \tfrac{\delta t}{2m} \\
& \text{end do} \\
& v \leftarrow v + F_l \tfrac{\Delta t}{2m}.
\end{aligned}
$$

Note that while the velocities will be updated on two different time-scales, the positions will be updated using only the smallest time-step. This procedure[17] allows one to update the expensive long range force much less frequently than updating the cheap short range forces and thus saves CPU time without sacrificing accuracy. Even for simple systems like a liquid consisting of atoms interacting through a Lennard-Jones potential this procedure leads to a speedup of as much as 400%. It is important to note that if one takes the switching function to be a Heaviside function, an approximation not recommended, the factorization of the propagator introduced reduces to the so called Verlet I integrator introduced by Grubmuller et *al.*.[36] However, factorizations like the one in Sections 2.4 and 2.6 are distinct from the Verlet I integrator and are not treated in ref. [36]. This should dispell some confusion with respect to these issues.

It is worth calling to attention one difference between the force subdivision used in r-RESPA[17] and the one used in the original non-reversible RESPA.[15] In the non-reversible RESPA paper we included the value of the long range force at the beginning of the time interval into the reference system equation of motion which was then integrated for n small time steps. We then solved the correction equation involving the difference between the true force and the reference system force for one large time step. This was shown to lead to a more stable integration scheme with much smaller long time drift than when the long range force was not introduced into the reference set of equations. Unfortunately, in the r-RESPA factorization there is no way to introduce the long range force at the beginning of the interval into the reference system propagator because that would remove reversibility. Strategies are being developed to implement such effects in new reversible integrators [37].

2.6 Combining Force Subdivision and Dynamic Subdivision

The preceding breakup for light and heavy particles can be combined with breaking the forces up into short and long range forces in r-RESPA[17] in a similar manner to what was done in non-reversible RESPA.[16] We can then

divided into short and long range parts. This yields the propagator for one large time step:[17]

$$G_{xyx}(\Delta t) = G_{lsl}^{(x)}\left(\frac{\Delta t}{2}\right) G_{lsl}^{(y)}(\Delta t) G_{lsl}^{(x)}\left(\frac{\Delta t}{2}\right), \tag{29}$$

where

$$G_{lsl}^{(x)}(\Delta t) = e^{\frac{\Delta t}{2} F_{xl}(x,y)\frac{\partial}{\partial p_x}} e^{iL_{xs}\Delta t} e^{\frac{\Delta t}{2} F_{xl}(x,y)\frac{\partial}{\partial p_x}}, \tag{30}$$

$$G_{lsl}^{(y)}(\Delta t) = e^{\frac{\Delta t}{2} F_{yl}(x,y)\frac{\partial}{\partial p_y}} e^{iL_{ys}\Delta t} e^{\frac{\Delta t}{2} F_{yl}(x,y)\frac{\partial}{\partial p_y}}, \tag{31}$$

and the middle propagator for x is integrated as:

$$e^{iL_{xs}\Delta t} = \left[e^{\frac{\delta t}{2} F_{xs}\frac{\partial}{\partial p_x}} e^{\delta t \dot{x}\frac{\partial}{\partial x}} e^{\frac{\delta t}{2} F_{xs}\frac{\partial}{\partial p_x}}\right]^{n_1}. \tag{32}$$

Likewise, the middle propagator for y is integrated n_2 times with a time step of $\Delta t_1 = n_1 \delta t$

$$e^{iL_{ys}\Delta t} = \left[e^{\frac{n_1\delta t}{2} F_{xs}\frac{\partial}{\partial p_y}} e^{n_1 \delta t \dot{y}\frac{\partial}{\partial y}} e^{\frac{n_1\delta t}{2} F_{ys}\frac{\partial}{\partial p_y}}\right]^{n_2}. \tag{33}$$

Thus At $= n_2 \Delta t_1 = n_1 n_2 \delta t$. It is simple matter to write down the Fortran pseudocode for this breakup.

2.7 The Applications of RESPA to Proteins and Chemical Systems

In order to apply the techniques discussed above to the MD simulation of biomolecules, one takes the Liouville operator for a macromolecule in *vacuo* containing N atoms to be

$$iL = \sum_{i}^{3N}[\dot{x}_i \frac{\partial}{\partial x_i} + F_i(x)\frac{\partial}{\partial p_i}\ ], \tag{34}$$

where

$$F(x) = F_{\text{stret}}(x) + F_{\text{bend}}(x) + F_{\text{tors}}(x) + F_{\text{Hbond}}(x) + F_{\text{vdW}}(x) + F_{\text{elec}}(x)\ , \tag{35}$$

$F_{\text{stret}}, F_{\text{bend}}, F_{\text{tors}}, F_{\text{Hbond}}, F_{\text{vdW}}$, and F_{elec} represent the forces for stretch, bending, torsion (including improper torsion), hydrogen-bonding, van der Waals, and electrostatic interactions, respectively. Their functional forms can be found elsewhere [38, 39]. The databases of parameters for these functional forms are generally called force fields. There are several force fields avail-

In an atomic level simulation, the bond stretch vibrations are usually the fastest motions in the molecular dynamics of biomolecules, so the evolution of the stretch vibration is taken as the "reference" propagator with the smallest time step. The nonbonded interactions, including van der Waals and electrostatic forces, are the slowest varying interactions, and a much larger time-step may be used. The bending, torsion and hydrogen-bonding forces are treated as intermediate time-scale interactions.

In addition, the non-bonded forces can be divided into several regions according to pair distances. The near region is normally more important than the distant region because the non-bonded forces decay with distance. Since most of the CPU time in a MD simulation is spent in the calculation of these non-bonded interactions, the separation in pair distance results in valuable speedups. Using a 3-fold distance split, the non-bonded forces are separated in 3 regions: near, medium, and far distance zones. Thus, the Liouville operator can be express as a sum of five terms

$$L = L_1 + L_2 + L_3 + L_4 + L_5 \ , \tag{36}$$

where

$$iL_1 \equiv \dot{x}\frac{\partial}{\partial x} + F_1(x)\frac{\partial}{\partial p} \tag{37}$$

$$iL_i \equiv F_i(x)\frac{\partial}{\partial p}, \qquad i = 2,3,4,5 \tag{38}$$

and

$$F_{1(\mathrm{X})} \equiv F_{\mathrm{stret(X)}} \tag{39}$$

$$F_2(x) \equiv F_{\mathrm{bend}}(x) + F_{\mathrm{tors}}(x) + F_{\mathrm{Hbond}}(x) \tag{40}$$

$$F_3(x) \equiv F_{\mathrm{vdW}}^{\mathrm{near}}(x) + F_{\mathrm{elec}}^{\mathrm{near}}(x) \tag{41}$$

$$F_4(x) \equiv F_{\mathrm{vdW}}^{\mathrm{med}}(x) + F_{\mathrm{elec}}^{\mathrm{med}} \tag{42}$$

$$F_5(x) \equiv F_{\mathrm{vdW}}^{\mathrm{far}}(x) + F_{\mathrm{elec}}^{\mathrm{far}} \ . \tag{43}$$

To separate the non-bonded forces into near, medium, and far zones, pair distance separations are used for the van der Waals forces, and box separations are used for the electrostatic forces in the Fast Multipole Method,[24] since the box separation is a more convenient breakup in the Fast Multipole Method (FMM). Using these subdivisions of the force, the propagator can be factorized according to the different intrinsic time scales of the various components of the force. This approach can be used for other complex systems involving long range forces.

2.8 Efficient Integrators for Systems with Coulomb Potentials

calculation of these forces scales as $O(N^2)$, where N is the number of force centers in the system, direct calculation of these forces in large systems makes molecular dynamics (or Monte Carlo) infeasible for large protein-water systems. The standard approach has been to truncate the long range forces so that their calculation scales as $O(N)$ for large enough systems. Unfortunately, truncation introduces significant non physical effects. To eliminate surface effects and to avoid the errors caused by truncation it is now becoming common to use periodic boundary conditions and to invoke Ewald summation. Optimal application of Ewald summation also scales as $O(N^{3/2})$ and thus becomes prohibitively expensive for large systems. Procacci and Marchi have combined Ewald with RESPA for protein solutions by including the total Fourier sum in the intermediate time loop.[42] A better strategy for applying r-RESPA to Ewald boundary conditions involves subdividing the Fourier space sum in such a way that the short time contribution is placed in the inner short time loop of RESPA and the "true" long range and slow part of the sum is put in the outer loop.[18]

There are three different algorithms for the calculation of the electrostatic forces in systems with periodic boundary conditions: (a) the (optimized) Ewald method, which scales like $O(N^{3/2})$; (b) the Particle Mesh Ewald (PME) method, which scales like $O(N \log N)$; and (c) the periodic Fast Multipole Method (PFMM), which scales like $O(N)$. For very large systems ($N \gtrsim 10^5$) it is expected that the PFMM will be the best choice, given its linear algorithmic complexity. It is of interest to determine the break-even point for these two methods. Because PME scales as $O(N \ln N)$ and periodic-FMM scales as $O(N)$, PFMM will be faster than PME for N greater than some N_0. The break-even point for these two methods combined with r-RESPA will be different because the implementation of r-RESPA will be different in these two cases. This break-even point has not yet been determined systematically. Figueirido et al. estimated that the break-even point for protein-water systems is $N_0 \approx 20{,}000$. Despite the significant progress in this field the optimal strategy has yet to be found.

Fast Multipole Methods To manage the calculation of all of the electrostatic interactions, several groups have experimented with approximate schemes, of which the most widely used is the Fast Multipole Method (FMM) of Greengard and Rokhlin[43, 44] and its variants.[45, 46, 47, 48, 49] This algorithm decreases the computational burden to $O(N)$ by cleverly exploiting a hierarchy of clusters and using multipolar expansions to approximate the potential produced by these clusters. The basic principle of FMM is rather elegant. It interpolates the potential and force on a particular charge due to distant charges not by direct calculation, but by using the local expansion of fields produced by the multipoles generated from those distant charges.

expansions. Each particle then interacts with the local field of distant multipoles. Meanwhile, the near range interactions are calculated directly by pairwise evaluation. Thus, the potential (and force) consists of two parts:

$$\Phi(\boldsymbol{x}) = \Phi^{\text{near}}_{\text{direct}}(\boldsymbol{x}) + \Phi^{\text{far}}_{\text{multipole}}(\boldsymbol{x}) \tag{44}$$

where $\Phi^{\text{near}}_{\text{direct}}$ contains the near range inter-particle interactions, and $\Phi^{\text{far}}_{\text{multipole}}$ contains the contribution from distant particles. A top-down FMM recursive method was proposed for multipole generation by Zhou and Berne[24]. Their method is based on White and Head-Gordon's simplified derivation [45]. Zhou and Berne have incorporated r-RESPA in this top-down FMM algorithm and applied it to isolated all-atom proteins.[24] They were able to achieve speed-ups on the order of fifteen-fold for the photo-synthetic reaction center over the direct UN-truncated calculation of the forces using the standard velocity verlet integrator. Fig 1 shows a comparison between the cpu times required by ordinary velocity verlet and r-RESPA for different size proteins. The figure also shows improvements that can be achieved by combining efficient algorithms such as the fast-multipole method (FMM) with r-RESPA.

The fast multipole method was first extended to periodic systems by Schmidt and Lee.[50] Figueirido *et al.* also designed a periodic FMM with a full derivation of the local field expansion which scales as $O(N)$.[26] These authors combined PFMM with r-RESPA producing in a very powerful algorithm (r-RESPA/PFMM) that is expected to be the optimum strategy for dealing with very large systems (see below).

Particle Mesh Ewald Methods Recently the particle mesh Ewald method (PME), and a smooth variant of it (SPME), developed by Darden *et al.*., have been described in the literature [51, 52, 53,461. These algorithms are based on Hockney and Eastwood's [54] idea of assigning charges to a mesh according to their real space positions; the CPU time savings come from applying the Fast Fourier Transform (FFT) to the particle mesh to accelerate the reciprocal-space calculations of the Ewald sum and to use a small cutoff in real space. The algorithms are found to be of order $O(N \log N)$. This method has been combined with r-RESPA by Procacci, Darden and Marchi.[55]

3 New Sampling Methods for the Extrinsic Multiple Time Scale Problem

Biomolecular systems often have rough energy landscapes. The sampling of rugged energy landscape poses special problems for molecular dynamics and Monte Carlo. As the system moves from one potential energy basin to another it must cross barriers that are large compared to kT. The crossing of

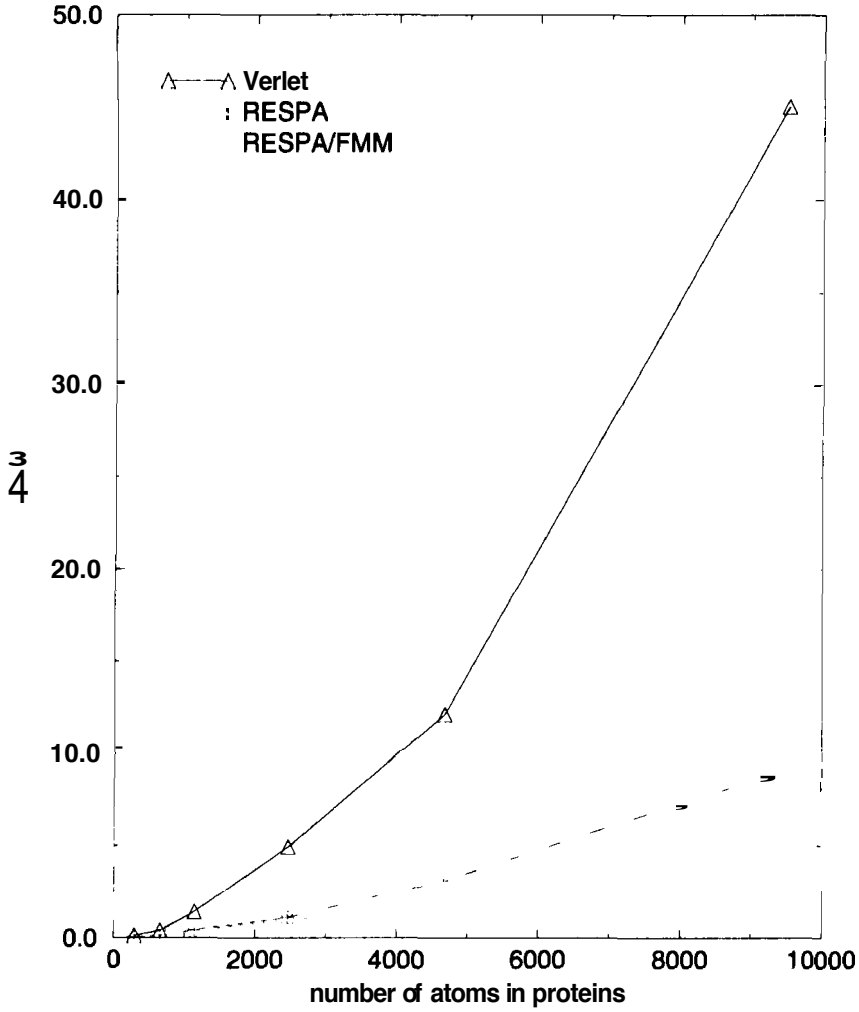

Fig. 1. CPU times (in hours) for 1 ps MD runs for various proteins using three different methods, direct velocity Verlet with a time-step 0.5 fs, r-RESPA with direct evaluation of electrostatic forces and an overall time-step of 4.0 fs, and r-RESPA/TFMM with an overall time-step 4.0 fs (combination of (2,2,2,2) in force breakup).The energy conservation parameter log ΔE for the three methods are comparable. The CPU time (hours) is for RISC6000 /MODEL 590 computer.

classes of interactions. First there are the local barriers that separate stable states of the torsion angles. Then there are barriers arising from close encounters of atoms on side chains as well as on the primary chain which result from very repulsive (r^{-12}) non-bonded interactions. There is a long history of using fictitious dynamics to sample the configuration space of complex systems. These schemes fall into three classes: Brownian or Langevin dynamics; BGK (Bhatnanger, Gross, Krook) dynamics; and Monte Carlo methods. One can accelerate all of these methods by a clever break-up of the forces, as we have done for molecular dynamics with RESPA and r-RESPA, but this will not solve the problem of sampling the rare barrier crossing events frequently

and final conformational states of large molecules. New methods are required to deal with rugged energy surfaces.

New Monte Carlo methods have been devised to deal with the intrinsic multiple time scale problem.[56, 10] The Hybrid Monte Carlo (HMC) method, combined with r-RESPA, as outlined in Sec. 2, can be used to speedup the intrinsic multiple time scale problem in MC sampling. In addition, new methods for speeding up barrier crossings in systems with rough energy landscapes like the Jump-Walking (J-Walking) method[57] and the Smart-Walking (S-Walking) method[58] can be combined with HMC and thereby r-RESPA. These methods are particularly useful for sampling the conformation space of many-body systems, such as proteins.[24, 26]Lastly, it is worth mentioning methods that allow the Lennard-Jones diameters a to fluctuate or that allow the barriers in the torsion angle potential to fluctuate. These methods very rapidly explore the configuration space.[59]

Hybrid Monte Carlo In standard MC only single particle moves are tried and accepted or rejected. Attempts to make many particle moves of the system before applying the Metropolis acceptance criterion leads to such small acceptance probabilities that this method is not efficient. Moreover it requires the recalculation of the whole potential after each attempted move, a costly computation especially when the move is likely to be rejected. One efficient method for generating collective moves is the Hybrid Monte Carlo method invented by Duane and Kennedy.[12] In this method one starts with a configuration of the system and samples momenta of the particles from a Maxwell distribution. Molecular dynamics is used to move the whole system for a time At and, because this time may be sufficiently large as to cause a reasonable energy change due the lack of strict energy conservation, one then accepts or rejects the move using the Metropolis criterion based on $\exp(-\beta H)$ where H is the hamiltonian of the system. This step is repeated over and over. In I-IMC, bad MD is used to generate efficient MC. It is important that the integrator used for generating the solution to the equations of motion be reversible because only then will this method satisfy detailed balance and only then will the method generate the canonical distribution and the Boltzrnann distribution. A number of authors have further elaborated the HMC method.[60, 61, 62, 63]

Since many systems of interest in chemistry have intrinsic multiple time scales it is important to use integrators that deal efficiently with the multiple time scale problem. Since our multiple time step algorithm, the so-called reversible Reference System Propagator Algorithm (r-RESPA) [17, 24, 18, 26] is time reversible and symplectic, they are very useful in combination with HMC for constant temperature simulations of large protein systems.

In HMC the momenta are constantly being refreshed with the consequence that the accompanying dynamics will generate a spatial diffusion process su-

diffusion can lead to smaller rates for barrier crossing. Thus the HMC or BGK methods may suppress barrier crossing. Parenthetically, it is important to note that stochastic methods such as Langevin dynamics or BGK dynamics will behave similarly. We have found[64] that in some systems the Nose' thermostat[65] may have a more beneficial sampling of different basins. Reversible integration schemes for these methods have been developed.[66] One way to improve these methods is to couple them to new methods for accelerating the dynamics on rugged energy landscapes such as the J-Walking method[57, 67, 68] or the S-Walking method.[58]

Jump Walking In the J-walking method, the MC or HMC sampling at the desired low temperature is infrequently punctuated by sampling from a higher temperature distribution for the same system. Since a higher temperature MC simulation can involve larger attempted moves and more frequent barrier crossings, this allows the system to access more conformational states according to the high temperature Boltzmann distribution. Then, the lower temperature walker attempts occasional jumps to the conformation states of the high temperature walker, thus enhancing the barrier crossing. The trial sampling distribution for these occasional jumps is the Boltzmann distribution at the higher temperature. The method is so constructed that one generates the correct low temperature Boltzmann distribution. Since the energy landscape of biomolecules contains very high barriers, it is often necessary to use many high temperature walks spaced at intervals of approximately 50 K and the CPU time required by this method will scale as the number of high temperature walks.

Smart Walking Jumping directly into a high temperature structure is not the only way to use the conformational space information from the J-Walker. Instead, the structure can be first relaxed before being jumped into.[58] Approximate minimization with a steepest descent method (or conjugate gradient method) will generate structures close to the local minimum. These relaxed configurations will significantly decrease the potential energy, and thus increase the jump success ratio dramatically. Each minimized structure is then regarded as one of the possible trial moves at low temperature and are accepted or rejected with acceptance probability function, that generates a Boltzmann distribution at the low temperature. Unlike the J-Walking acceptance probability, this scheme, which is called Smart Walking[58] (or S-Walking), will dramatically increase the jump success ratio from one basin to another. It also enables the system to explore more phase space and undergo more efficient barrier-crossings. This S-walking method avoids the linear increase of CPU time and memory usage required by the multiple-stage J-Walking method, because it is not necessary to use multiple stages for

provided the time between S-jumps is much longer than the time required by the low temperature walker to explore its local basin effectively (more discussion follows in the section on results). This new S-Walking algorithm only requires a simple modification of the J-Walking algorithm.

4 Summary

Integrators based on r-RESPA, when combined with enhanced methods for calculating long-range electrostatic forces, such as the FMM or SPME schemes have led to a considerable speed-up in the CPU time for large scale simulations of biomacromolecular solutions. Since r-RESPA is symplectic such integrators are very stable. Moreover since r-RESPA is time reversible it can be used in Hybrid Monte Carlo and satisfies the condition of detailed balance. This HMC method can be used in enhanced sampling methods such as J-Walking and S-Walking methods which lead to a more rapid exploration of rugged energy landscapes and thus to enhanced conformational searches.

Acknowledgements

This work was supported by grants from the National Science Foundation and the National Institutes of Health.

References

1. B. J. Alder and T. E. Wainwright. J. Chem. Phys., 27:1208–9, 1957.
2. B. J. Alder and T. E. Wainwright. J. Chem. Phys., 31:459–56, 1959.
3. A. Rahman. Phys. Rev., 136A:405–11, 1964.
4. G. D. Harp and B. J. Berne. J. Chem. Phys., 49:1249–54, 1968.
5. G. D. Harp and B. J. Berne. Phys. Rev., A2:975–96, 1970.
6. A. Rahman and F. H. Stillinger. J. Chem. Phys., 55:3336–59, 1971.
7. M. P. Allen and D. J. Tildesley. Computer Simulation of Liquids. Oxford University Press, Oxford, 1987.
8. J. A. McCammon and S. C. Harvey. Dynamics of Proteins and Nucleic Acids. Cambridge University Press, Cambridge, 1987.
9. J. M. Haille. Molecular Dynamics Simulation: Elementary Methods. John Wiley & Sons, USA, 1992.
10. D. Frenkel and B. Smit. Understanding Molecular Simulation. Academic Press, 1996.
11. J. P. Ryckaert and G. Ciccotti and H. J. C. Berendsen. J. Comp. Phys., 23:327, 1977.
12. S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Phys. Rev. Lett. B, 195:216–222, 1987.

14. M. Tuckerman, B. J. Berne, and A. Rossi. J. Chem. Phys., 94:1465–1469, (1991).
15. B. J. Berne M. Tuckerman and G. Martyna. J. Chem. Phys., 94:6811–6815, (1991).
16. M. Tuckerman and B. J. Berne. J. Chem. Phys., 95:8362–8364, (1991).
17. M. E. Tuckerman, B. J. Berne, and G.J. Martyna. J. Chem. Phys., 97:1990–2001, (1992).
18. S. J. Stuart, R. Zhou, and B. J. Berne. J. Chem. Phys., 105:1426–1436, (1996).
19. M.E. Tuckerman and M. Parrinello. J. Chem. Phys., 101:1302–1315, (1994).
20. M. E. Tuckerman and G. J. Martyna and B. J. Berne. J. Chem. Phys., 93:1287, 1990.
21. J. Hutter, M.E. Tuckerman, and M. Parrinello. J. Chem. Phys., (1995).
22. D. Humphreys, R. A. Friesner, and B.J. Berne. J. Phys. Chem., 98:6885–6892, (1994).
23. D. Humphreys, R. A. F'riesner, and B. J. Berne. J. Phys. Chem., 99:10674–10685, (1995).
24. R. Zhou and B.J. Berne. J. Chem. Phys., 103:9444–9458, (1995).
25. M. Watanabe and M. Karplus. J. Chem. Phys., 99:8063–8074, 1993.
26. F. Figueirido, R. Zhou, B. J. Berne, and R. M. Levy. J. Chem. Phys., 106:9835–9849, (1997).
27. L. Verlet. Phys. Rev., 159:98–103, 1967.
28. T. A. Andrea, W. C. Swope, and H. C. Andersen. J. Chem. Phys., 79:4576–4584, 1983.
29. H. Goldstein. Classical Mechanics. Addison Wesley Publishing Company, Inc., 1980.
30. Haruo Yoshida. Phys. Lett. *A.*, 150:262–268, (1990).
31. J. J. Biesiadecki and R. D. Skeel. J. Comp. Phys., 109:318–328, 1993.
32. S. K. Gray, D. W. Noid, and B. G. Sumpter. J. Chem. Phys., 101:4062–4072, 1994.
33. S. Auerbach and A. F'riedman. J. Comp. Phys., 93:189, 1991.
34. M.E. Tuckerman and B. J. Berne. J. Chem. Phys., 98:7301–7318, (1993).
35. P. Procacci and B.J. Berne. J. Chem. Phys., 101:2421, (1994).
36. H. Grubmuller, H. Heller, A. Windemuth, and K. Schulten. Molecular Simulation, 6:121–142, (1991).
37. G.J. Martyna and M.E. Tuckerman. J. Chem. Phys., 102:8071–8077, (1996).
38. F. Mohamadi and N. G. J. Richards and W. C. Guida and R. Liskamp and M. Lipton and C. Caufield and G. Chang and T. Hendrickson and W. C. Still. J. Comp. Chem., 11:440, 1990.
39. S. J. Weiner and P. A. Kollman and D. T. Nguyen and D. A. Case. J. *Comp.* Chem., 7:230, 1986.
40. W. L. Jorgensen and J. Tirado-Rives. J. Am. Chem. Soc., 110:1657, 1988.
41. B. R. Brooks and R. E. Bruccoeri and B. D. Olafson and D. J. States and S. Swaminathan and M. Karplus. J. Comp. Chem., 4:187–217, 1983.
42. P. Procacci and M. Marchi. J. Chem. Phys., 104:3003–3012, 1996.
43. L. Greengard. The Rapid Evaluation of Potential Fields in Particle Systems. The MIT Press, Cambridge, Massachusetts, 1988.
44. L. Greengard and V. Rokhlin. J. Comp. Phys., 73:325, 1987.

47. H.-Q. Ding and N. Karasawa and W. A. Goddard III. J. Chem. Phys., 97:4309, 1992.
48. M. Saito. Molecular Simulations, 8:321–333, 1992.
49. C. Niedermeier and P. Tavan. Molecular Simulation., 17:57–66, (1996).
50. K. E. Schmidt and M. A. Lee. J. Stat. Phys., 63:1223–1235, 1991.
51. T. Darden, D. M. York, and L. G. Pedersen. Particle mesh Ewald: An N log(N) method for Ewald sums in large systems. J. Chem. Phys., 98:10089–10092, 1993.
52. H. G. Petersen. J. Chem. Phys., 103:3668–3679, 1995.
53. U. Essman, L. Perera, M. L. Berkowitz, T. Darden, H. Lee, and L. G. Pedersen. J. Chem. Phys., 103:8577–8593, 1995.
54. R. W. Hockney and J. W. Eastwood. Computer Simulation Using Particles. Adam Hilger, Bristol-New York, 1989.
55. P. Procacci and T. Darden and M. Marchi. J. Phys. Chem., 100:10464–10468, 1996.
56. B. J. Berne and John E. Straub. Current Topics in *Structural* Biology, 7 No 2:181–189, (1997).
57. D. D. Frantz, D. L. Freeman, and J. D. Doll. J. Chem. Phys., 93:2769–2784, 1990.
58. R. Zhou and B. J. Berne. *J.* Chem. Phys., 107:9185–9196, (1997).
59. Z. Liu and B. J. Berne. J. Chem. Phys., 99:6071, (1993).
60. U. H. E. Hansmann, Y. Okamoto, and F. Eisenmenger. Chem. Phys. Lett., 259:321–330, 1996.
61. R. M. Neal. J. Comp. Phys., 111:194–203, 1994.
62. S. Gupta, A. Irback, F. Karsch, and B. Petersson. Phys. Lett. B, 242:437–443, 1990.
63. P. B. Markenzie. Phys. Lett. B, 226:369–371, 1989.
64. M. Tuckerman, B. J. Berne, G. Martyna, and M. Klein. J. Chem. Phys., 99:2796–2784, (1993).
65. S. Nosé. J. Chem. Phys., 81:511–519, 1984.
66. G.J. Martyna, M.E. Tuckerman, D.J. Tobias, and M.L. Klein. Mol. Phys., 87:1117–1157, (1996).
67. D. L. Freeman, D. D. Frantz, and J. D. Doll. J. Chem. Phys., 97:5713, 1992.
68. A. Matro, D. L. Freeman, and R. Q. Topper. J. Chem. Phys., 104:8690, 1996.